BEA WebLogic Adapter for ClarifyCRM®

User Guide

Release 8.1.1
Document Date: October 2003

Copyright

Restricted Rights Legend

Trademarks or Service Marks

Contents

About This Document

This document describes how to use the BEA WebLogic Adapter for ClarifyCRM. This document is organized as follows:

- Chapter 1, "Introducing the BEA WebLogic Adapter for ClarifyCRM," describes the adapter, how it relates to both ClarifyCRM business objects and WebLogic Integration.
- Chapter 2, "Creating Schemas and Repositories," describes how to generate schemas for your ClarifyCRM business objects. This section also describes how to generate schema repositories and manifests.
- Chapter 3, "Creating Application Views for the CBO Interface," describes application views and how to use them to configure events and services with the CBO interface of ClarifyCRM.
- Chapter 4, "Creating Application Views for the ClearBasic Interface," describes application views and how to use them to configure events and services with the ClearBasic interface of ClarifyCRM.
- Appendix A, "Customizing Tuxedo Field Tables," describes how to customize the Tuxedo field definition file.

Who Should Read This Documentation

This document is intended for the following members of an integration team:

- Integration Specialists—Lead the integration design effort. Integration specialists have expertise in defining the business and technical requirements of integration projects, and in designing integration solutions that implement specific features of WebLogic Integration. The skills of integration specialists include business and technical analysis, architecture design, project management, and WebLogic Integration product knowledge.

- Technical Analysts—Provide expertise in an organization's information technology infrastructure, including telecommunications, operating systems, applications, data repositories, future technologies, and IT organizations. The skills of technical analysts include technical analysis, application design, and information systems knowledge.
- Enterprise Information System (EIS) Specialists—Provide domain expertise in the systems that are being integrated using WebLogic Integration adapters. The skills of EIS specialists include technical analysis and application integration design.
- System Administrators—Provide in-depth technical and operational knowledge about databases and applications deployed in an organization. The skills of system administrators include capacity and load analysis, performance analysis and tuning, deployment topologies, and support planning.

Additional Information

To learn more about the software components associated with the adapter, see the following documents:

- *BEA WebLogic Adapter for ClarifyCRM Release Notes*

  `http://edocs.bea.com/wladapters/clarify/docs811/pdf/relnotes.pdf`
- *BEA WebLogic Adapter for ClarifyCRM Installation and Configuration Guide*

  `http://edocs.bea.com/wladapters/clarify/docs811/pdf/install.pdf`
- *Introduction to the BEA WebLogic Adapters*

  `http://edocs.bea.com/wladapters/docs81/pdf/intro.pdf`
- BEA WebLogic Adapters 8.1 Dev2Dev Product Documentation

  `http://dev2dev.bea.com/products/wladapters/index.jsp`
- Application Integration documentation

  `http://edocs.bea.com/wli/docs81/aiover/index.html`

  `http://edocs.bea.com/wli/docs81/aiuser/index.html`
- BEA WebLogic Platform documentation

  `http://edocs.bea.com/platform/docs81/index.html`

- ClarifyCRM documentation

  www.amdocs.com

How to Use This Document

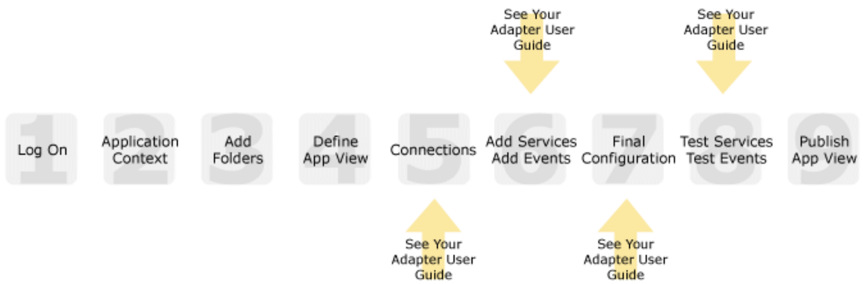
This document is designed to be used in conjunction with *Using the Application Integration Design Console*, available at the following URL:

http://edocs.bea.com/wli/docs81/aiuser/index.html

Using the Application Integration Design Console descibes, in detail, the process of defining an application view, which is a key part of making an adapter available to process designers and other users. What *Using the Application Integration Design Console* does *not* cover is the specific information about Adapter for ClarifyCRM that you need to supply to complete the application view definition. You will find that information in this document.

At each point in *Using the Application Integration Design Console* where you need to refer to this document, you will see a note that directs you to a section in your adapter user guide, with a link to the edocs page for adapters. The following roadmap illustration shows where you need to refer from *Using the Application Integration Design Console* to this document.

Figure 1 Information Interlock with ***Using the Application Integration Design Console***


See Your Adapter User Guide
See Your Adapter User Guide
Log On
Application Context
Add Folders
Define App View
Connections
Add Services Add Events
Final Configuration
Test Services Test Events
Publish App View
See Your Adapter User Guide
See Your Adapter User Guide

Contact Us!

Your feedback on the BEA WebLogic Adapter for ClarifyCRM 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 ClarifyCRM documentation.

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

If you have any questions about this version of BEA WebLogic Adapter for ClarifyCRM, or if you have problems using the BEA WebLogic Adapter for ClarifyCRM, 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.
	Examples:
	`#include <iostream.h> void main ( ) the pointer psz`
	`chmod u+w *`
	`\tux\data\ap`
	`.doc`
	`tux.doc`
	`BITMAP`
	`float`

Convention	Item
`monospace boldface text`	Identifies significant words in code. *Example*: `void` **`commit`** `( )`
`monospace italic text`	Identifies variables in code. *Example*: `String` *`expr`*
UPPERCASE TEXT	Indicates device names, environment variables, and logical operators. *Examples*: LPT1 SIGNON 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.
	Example:
	`buildobjclient [-v] [-o name ] [-f` *`file-list`*`]...`
	`[-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:
	• That an argument can be repeated several times in a command line
	• That the statement omits additional optional arguments
	• That you can enter additional parameters, values, or other information
	The ellipsis itself should never be typed.
	Example:
	`buildobjclient [-v] [-o name ] [-f` *`file-list`*`]...`
	`[-l` *`file-list`*`]...`

. . .	Indicates the omission of items from a code example or from a syntax line. The vertical ellipsis itself should never be typed.

Introducing the BEA WebLogic Adapter for ClarifyCRM

This section introduces the BEA WebLogic Adapter for ClarifyCRM and describes how the adapter enables integration with ClarifyCRM business objects and WebLogic Integration.

It includes the following topics:

- About the BEA WebLogic Adapter for ClarifyCRM
- Getting Started With the Adapter for ClarifyCRM

About the BEA WebLogic Adapter for ClarifyCRM

The BEA WebLogic Adapter for ClarifyCRM connects to your ClarifyCRM system so that you can easily use your ClarifyCRM data and functions within your business processes. The adapter provides scalable, reliable, and secure access to your ClarifyCRM system.

This section includes the following topics:

- Supported ClarifyCRM Operations for Application Integration
- About ClarifyCRM, Tuxedo, and the WebLogic Tuxedo Connector
- About ClarifyCRM Business Objects
- How ClarifyCRM Business Objects Access the Database
- Executing ClarifyCRM Business Functions
- Benefits of the Adapter for ClarifyCRM

Supported ClarifyCRM Operations for Application Integration

The Adapter for ClarifyCRM supports synchronous and asynchronous, bi-directional message interactions for ClarifyCRM.

It provides integration with the following ClarifyCRM interfaces:

- Access to ClarifyCRM business objects through the CBO interface
- Access to ClarifyCRM through the ClearBasic interface

About ClarifyCRM, Tuxedo, and the WebLogic Tuxedo Connector

This section applies only to the ClearBasic interface of ClarifyCRM.

When you are using the ClearBasic interface, ClarifyCRM runs under the control of the WebLogic Tuxedo Transaction Manager. and makes use of all the features that Tuxedo provides, such as transaction control, application clustering, or distributed invocation. The WebLogic Server itself provides support for Tuxedo services via the WebLogic Tuxedo Connector (WTC). With WTC, WebLogic Java-based modules (Enterprise Java Beans, or EJBs) may access the Application to Monitor Interface (ATMI) of Tuxedo from the WebLogic platform and make service calls of Tuxedo modules. Conversely, Tuxedo modules may make service calls of WTC-hosted EJB modules. The Java ATMI (JATMI) capability and the domain-to-domain communication capability of Tuxedo and the WTC are the foundation of the BEA WebLogic Adapter for ClarifyCRM.

Data passed between two Tuxedo modules, whether they be EJBs hosted on WebLogic or C-based programs hosted on a Tuxedo application server, is done in a typed buffer format called `FML` or `FML32` (a 32-bit addressed typed buffer). The typed buffer is described in a field definition file and each Tuxedo application system may have its own unique list of fields applicable to the functions performed by the ATMI / JATMI modules installed.

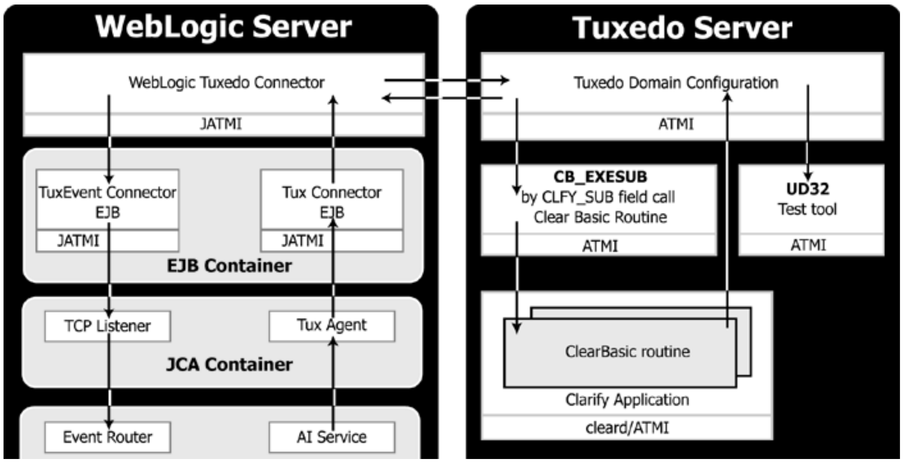
The BEA WebLogic Adapter for ClarifyCRM allows WebLogic Integration workflows to make service calls of Clarify through JATMI service calls of remote Tuxedo services. The standard Clarify `CB_EXESUB` Tuxedo module routes these requests into any of the registered ClarifyCRM ClearBasic routines, marshalling parameters from the FML typed buffer and unmarshalling returned ClearBasic results.

ClarifyCRM ClearBasic modules may use the standard ClearBasic call method to invoke a remote Tuxedo service. The adapter provides a WTC-hosted service, `IWAYWTC_GEN_SVC`, which marshals its event data from the FML typed buffer, creates a corresponding XML representation, and then posts the event XML document into the WebLogic Integration EventRouter via a standard TCP event listener. Workflows registered to the event posted in by the IWAYWTC_GEN_SVC EJB JATMI module will be invoked.

Each Tuxedo and each WTC installation may define its own list of fields that are relevant to them. However, the fields passed between the two must correspond with matching field names, IDs, and data types. See Appendix A, "Customizing Tuxedo Field Tables," for instructions about how to customize the WebLogic WTC environment for your custom Clarify application modules. Additionally, the BEA WebLogic Adapter for ClarifyCRM makes use of two specific fields in the FML typed buffers shared between the two platforms:

- The CLFY_SUB field is mandated by the Clarify CB_EXESUB module. The value contained in this field buffer is used to determine which ClearBasic routine to invoke.
- The second field, whose name defaults to HI_WORKFLOW_NAME, is configurable by the adapter, and controls the root element of the event schema being created. The contents of this field buffer will be used as the root element of the created XML document.

Figure 1-1 BEA WebLogic Adapter for ClarifyCRM Architecture



About ClarifyCRM Business Objects

This section applies only to the CBO interface of ClarifyCRM.

ClarifyCRM Business Objects (CBO) are part of the Clarify eBusiness Framework for developing applications for the Clarify eFrontOffice (CeFO) database.

These business objects are C++ objects with Java layer that provide access to data in the CeFO database. Each type of business object implements part of the Clarify data model and encapsulates the application logic for working with that part of the model.

How ClarifyCRM Business Objects Access the Database

A business object has methods and properties that can be used to query and update data in a CeFO database table.

Each business object contains a rowset, which is an in-memory copy of a set of rows from a CeFO database table. When database table is queried, the business object holds the results of the query in its rowset. A row in the rowset can be selected and any of the values of fields in that row can be obtained.

The rowset can be used to make changes to data in the database table. The values of fields in rows can be modified and the changed rows committed back to the database. If a new row must be added to the database table, add the rows to the rowset in the business object and commit the new rows to the database.

Executing ClarifyCRM Business Functions

This section applies only to the CBO interface of the adapter for ClarifyCRM.

You can use the BEA WebLogic Adapter for ClarifyCRM to perform inserts, updates, and deletes against ClarifyCRM objects stored in the ClarifyCRM database. The adapter supports the following objects:

- Contact
- PartMaster
- ItemPricesQty
- ItemPrice
- Quote
- QuoteSchedule
- QuoteLine
- QuotePONumber
- QuoteScheduleBilltoCust

- QuoteScheduleShiptoCust

You can also use the adapter to integrate ClarifyCRM with non-ClarifyCRM systems.

Benefits of the Adapter for ClarifyCRM

The combination of the adapter and WebLogic Integration supplies everything you need to integrate your workflows and enterprise applications with your ClarifyCRM system. The Adapter for ClarifyCRM provides these benefits:

- Integration can be achieved without custom coding.
- Business processes can be started by events generated by ClarifyCRM.
- Business processes can request and receive data from your ClarifyCRM system using services.
- Adapter events and services are standards-based. The adapter services and events provide extensions to the *J2EE Connector Architecture* (JCA) version 1.0 from Sun Microsystems, Inc. For more information, see the Sun JCA page at the following URL:

  `http://java.sun.com/j2ee/connector/`

- The adapter and WebLogic Integration solution is scalable. The BEA WebLogic Platform provides clustering, load balancing, and resource pooling for a scalable solution. For more information about scalability, see the following URL:

  `http://edocs.bea.com/wls/docs81/cluster/index.html`

- The adapter and WebLogic Integration solution benefits from the fault-tolerant features of the BEA WebLogic Platform. For more information about high availability, see the following URL:

  `http://edocs.bea.com/wli/docs81/deploy/index.html`

- The adapter and WebLogic Integration solution is secure, using the security features of the BEA WebLogic Platform and the security of your ClarifyCRM system. For more information about security, see the following URL:

  `http://edocs.bea.com/wls/docs81/secintro/index.html`

Getting Started With the Adapter for ClarifyCRM

This section gives an overview of how to get started using the BEA WebLogic Adapter for ClarifyCRM within the context of an application integration solution. Integration with ClarifyCRM involves the following tasks:

- Step 1: Design the Application Integration Solution
- Step 2: Determine the Required ClarifyCRM Business Workflows
- Step 3: Generate Schemas for ClarifyCRM Services and Events
- Step 4: Define Application Views and Configure Services and Events
- Step 5: Integrate with Other BEA Software Components
- Step 6: Deploy the Solution to the Production Environment

Step 1: Design the Application Integration Solution

The first step is to design an application integration solution, which includes (but is not limited to) such tasks as:

- Defining the overall scope of application integration.
- Determining the business process(es) to integrate.
- Determining which WebLogic Platform components will be involved in the integration, such as web services or workflows designed in WebLogic Workshop, portals created in WebLogic Portal, and so on.
- Determining which external systems and technologies will be involved in the integration, such as ClarifyCRM systems and other EISs.
- Determining which BEA WebLogic Adapters for WebLogic Integration will be required, such as the BEA WebLogic Adapter for ClarifyCRM. An application integration solution can involve multiple adapters.

This step involves the expertise of business analysts, system integrators, and EIS specialists (including ClarifyCRM specialists). Note that an application integration solution can be part of a larger integration solution.

Step 2: Determine the Required ClarifyCRM Business Workflows

Within the larger context of an application integration project, you must determine which specific ClarifyCRM integration objects and workflows are required to handle services and events to support the business processes in the application integration solution. Or, if you are invoking ClarifyCRM business services or business components directly, rather than through a workflow, you must determine the tasks you need to complete.

Factors to consider include (but are not limited to):

- Type of ClarifyCRM integration objects, workflows, and transport used to access the ClarifyCRM system.
- ClarifyCRM transactions involved in business processes
- Logins required to access ClarifyCRM transports and perform the required operations
- Whether operations are, from the adapter point of view:
  - services, which notify the ClarifyCRM system, via an XML document, with a request for action, and, in addition, whether such services should be processed synchronously or asynchronously
  - events, which are notifications from the ClarifyCRM system that trigger workflows

This step involves the expertise of ClarifyCRM specialists, including analysts and administrators.

Step 3: Generate Schemas for ClarifyCRM Services and Events

After identifying the ClarifyCRM integration objects and workflows required for the application integration solution, you must generate the XML schemas that will be used to exchange data with one or more ClarifyCRM systems:

- Services require two XML schemas: one for the ClarifyCRM request and another for the ClarifyCRM response.
- Events require a single XML schema to handle the data sent by the ClarifyCRM system.

Since ClarifyCRM does not expose the metadata that the BEA Application Explorer tool uses to generate schemas, you must generate the schemas for ClarifyCRM services and events yourself. BEA supplies samples of the schemas. To learn more about the sample file, see the *BEA WebLogic Adapter for ClarifyCRM Installation and Configuration Guide*. To learn more about generating schemas, see Chapter 2, "Creating Schemas and Repositories."

Step 4: Define Application Views and Configure Services and Events

After you create the schemas for your ClarifyCRM services or events, you create an application view that provides an XML-based interface between WebLogic Server and a particular ClarifyCRM system within your enterprise. If you are accessing multiple ClarifyCRM systems, you define a separate application view for each ClarifyCRM system you want to access. To provide different levels of security access (such as "guest" and "administrator"), define a separate application view for each security level.

Once you define an application view, you can configure events and services in that application view that employ the XML schemas that you created in Step 3: Generate Schemas for ClarifyCRM Services and Events. To learn more about generating schemas, see Chapter 2, "Creating Schemas and Repositories."

To learn more about defining application views for the CBO interface, see Chapter 3, "Creating Application Views for the CBO Interface" in conjunction with*Using the Application Integration Design Console*, at the following URL:

`http://edocs.bea.com/wli/docs81/aiuser/index.html`

To learn more about defining application views for the ClearBasic interface, see Chapter 4, "Creating Application Views for the ClearBasic Interface" in conjunction with*Using the Application Integration Design Console*, at the following URL:

`http://edocs.bea.com/wli/docs81/aiuser/index.html`

Step 5: Integrate with Other BEA Software Components

Once you have configured and published one or more application views for ClarifyCRM integration, you can integrate these application views into other BEA software components, such as workflows or web services created in BEA WebLogic Workshop, or portals built with BEA WebLogic Portal.

For more information, see *Using the Application Integration Design Console*, particularly Chapter 3, "Using Application Views with Application Workflows," at the following URL:

`http://edocs.bea.com/wli/docs81/aiuser/index.html`

Step 6: Deploy the Solution to the Production Environment

After you have designed, built, and tested your application integration solution, you can deploy it into a production environment. The following list describes some of the tasks involved in deploying an application integration:

- Design the deployment.
- Deploy the required components of the BEA WebLogic Platform.
- Install and deploy the BEA WebLogic Adapter for ClarifyCRM as described in *BEA WebLogic Adapter for ClarifyCRM Installation and Configuration Guide*

- Deploy your application views and schemas for ClarifyCRM integration.
- Verify business processes in the production environment.
- Monitor and tune the deployment.

Creating Schemas and Repositories

Before you can create an application view and use the adapter in a workflow, you must have schemas that describe the XML corresponding to each ClarifyCRM event and service you expect to use with the adapter. Many EISes expose metadata that allows the BEA Application Explorer to generate these schemas. However, ClarifyCRM does not expose this metadata, so you must generate these schemas yourself.

This section explains how to generate the schemas and other files required by the Adapter for ClarifyCRM.

This section includes the following topics:

About Schema Repositories

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

When you are defining application views, each application view is assigned to a schema repository. You can assign several application views to the same repository.

Naming a Schema Repository

The schema repository has a three-part naming convention:

`session_path``\clarify\`*`connection_name`*

Here,

- *`session_path`* is the schema's session base path (also known as the session base directory), beneath which you can store multiple sessions of schemas.
- `clarify` is the type of adapter.
- *`connection_name`* is a name representing a particular instance of the adapter.

For example, if the session path is `/usr/opt/bea/bse`, the adapter type is ClarifyCRM, and the connection name is TestClarify, then the schema repository is the directory:

```
/usr/opt/bea/bse/ClarifyCRM/TestClarify
```

About 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 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 schema repository.

The following is an example of a manifest file. It illustrates the relationships between events and services and their related schemas:

Listing 2-1 Sample Manifest File

```
<manifest>
<connection/>
   <schemaref name="clarify">
         <request root="sp" file="bea_clarify_request.xsd" />
         <response root="response" file="bea_clarify_response.xsd" />
         <event root="ClarifyEvents" file="bea_clarify_events_row.xsd" />
   </schemaref>
</manifest>
```

The manifest has 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 Application View Console.

This sample manifest has these schema references:

- One for events.

  Events require only one schema, defined by the `event` tag. This relates the root tag of an XML document produced when a ClarifyCRM event occurs.

- Two for services.

  Services require two schemas: 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.

You may also create a schema reference that comprises both services and events.

Creating a Repository Manifest

The repository manifest is an XML file with the root element `manifest` and two sub-elements:

- `connection`, which appears once, and which you can ignore because it is not used by the BEA WebLogic Adapter for ClarifyCRM at runtime, but only by the BEA Application Explorer when generating schemas from ClarifyCRM metadata.

  Note: At the time this document was created, BEA Application Explorer did not support the dynamic creation of service and event schemas for ClarifyCRM. Check with BEA Systems for the latest BEA Application Explorer adapter support.

- `schemaref`, which appears multiple times, once for each schema name, and which contains all three schemas—request, response, and event.

To create a manifest:

1. Create an XML file with the following structure:

```
<manifest>
   <connection>
   </connection>
</manifest>
```

2. For each new event or service schema you define, create a `schemaref` section using this model:

```
<schemaref name="Product">
   <request root="sp" file="bea_clarify_request.xsd"/>
   <response root="response" file="bea_clarify_response.xsd"/>
   <event root="clarify" file="bea_clarify_event_row.xsd"/>
</schemaref>
```

   Here, the value you assign to:

   – `file` is the name of the file in the schema repository.

   – `root` is the name of the root element in the actual instance documents that will arrive at, or be sent to, the event or service.

The following is an example of an instance document for the Product event referred to in "Creating a Repository Manifest" on page 2-3.

Listing 2-2 Request Document for ClarifyCRM Contact_Insert Service

```
<?xml version="1.0"?>
      <sp>
       <data>
        <cbo name="contact" action="create">
         <keys>
          <key  name="first_name">Bill</key>
          <key  name="last_name">Wier</key>
          <key  name="phone">408-777-9000</key>
          <keyRel name="contact2contact_role">2030120</keyRel>
          <fields>
           <field name="address_1">235 darwin drive</field>
           <field name="address_2">APT 235</field>
          <field name="City">fremont</field>
          <field name="e_mail">bwier@bww.com</field>
          <field name="fax_number">408-999-9898</field>
          <field name="mail_stop">M123</field>
          <field name="salutation">Mr.</field>
          <field name="status">0</field>
          <field name="title">CEO</field>
         </fields>
         <Relations>
          <Rel name="contact2contact_role">26589412</Rel>
         </Relations>
        </keys>
       </cbo>
      </data>
     </sp>
```

The following listing is a schema matching this instance document; it may be manually coded or generated from any XML editor:

Listing 2-3 Schema Matching Contact_Add Service Request Document

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
elementFormDefault="qualified">
       <xsd:complexType name="cboType">
              <xsd:sequence>
                     <xsd:element name="keys" type="keysType"
minOccurs="0"/>
                     <xsd:element name="fields" type="fieldsType"
minOccurs="0"/>
              </xsd:sequence>
              <xsd:attribute name="name" type="xsd:string"
use="required"/>
              <xsd:attribute name="action" type="xsd:string"/>
              <xsd:attribute name="relation" type="xsd:string"/>
       </xsd:complexType>
```

```
        <xsd:complexType name=”dataType”>
                <xsd:sequence>
                        <xsd:element name=”cbo” type=”cboType”/>
                </xsd:sequence>
        </xsd:complexType>
        <xsd:element name=”sp” type=”spType”/>
        <xsd:complexType name=”fieldType”>
                <xsd:simpleContent>
                        <xsd:extension base=”xsd:string”>
                                <xsd:attribute name=”name” type=”xsd:string”
use=”required”/>
                                <xsd:attribute name=”data_type”
type=”xsd:string”/>
                        </xsd:extension>
                </xsd:simpleContent>
        </xsd:complexType>
        <xsd:complexType name=”fieldsType”>
                <xsd:sequence>
                        <xsd:element name=”cbo” type=”cboType” minOccurs=”0”
maxOccurs=”unbounded”/>
                        <xsd:element name=”field” type=”fieldType”
minOccurs=”0” maxOccurs=”unbounded”/>
                        <xsd:element name=”cbo” type=”cboType” minOccurs=”0”
maxOccurs=”unbounded”/>
                </xsd:sequence>
        </xsd:complexType>
        <xsd:complexType name=”keysType”>
                <xsd:sequence>
                        <xsd:element name=”field” type=”fieldType”/>
                </xsd:sequence>
        </xsd:complexType>
        <xsd:complexType name=”spType”>
                <xsd:sequence>
                        <xsd:element name=”data” type=”dataType”/>
                </xsd:sequence>
        </xsd:complexType>
</xsd:schema>
```

About the Sample Files

Supplied with the BEA WebLogic Adapter for ClarifyCRM are XSD schema files for service requests and responses and for events. With these schemas, all of the ClarifyCRM methods and objects that the adapter supports can be used in application views. These files are supplied in a zip file called `bea_clarify_samples.zip`. When you create an application view, the adapter will automatically extract the schemas to the directory you specify as the session path. These schemas can be used in the `manifest.xml` files you build.

For services, use the schemas as defined in the table that follows.

Table 2-1 Service Schemas

For . . .	Use the following schema . . .
Service requests	`bea_clarify_request.xsd`
Service responses	`bea_clarify_response.xsd`

For events, use one of three files in the `manifest.xml` file, depending on the format you select when you build the event, as defined in the table that follows.

Table 2-2 Event Schemas

Format	Use the following schema . . .
Row	`bea_clarify_event_row.xsd`
Column	`bea_clarify_event_col.xsd`
Field	`bea_clarify_event_field.xsd`

The `bea_clarify_samples.zip` file also includes sample request files that work with the above-mentioned schemas. These instances contain the object and operation inside ClarifyCRM to be executed, in addition to sample data.

Creating Application Views for the CBO Interface

The information in this chapter applies only to the CBO interface for the adapter for ClarifyCRM. To learn more about creating application views for the ClearBasic interface , see Chapter 4, "Creating Application Views for the ClearBasic Interface."

This section provides information on application views and deployed services and events, and includes the following topics:

How to Use This Document

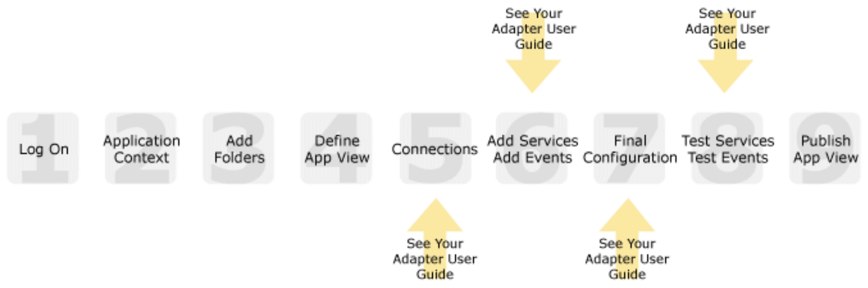
This document is designed to be used in conjunction with *Using the Application Integration Design Console*, available at the following URL:

http://edocs.bea.com/wli/docs81/aiuser/index.html

Using the Application Integration Design Console describes, in detail, the process of defining an application view, which is a key part of making an adapter available to process designers and other users. What *Using the Application Integration Design Console* does *not* cover is the specific information—about connections to your ClarifyCRM system, as well as supported services and events—that you must supply as part of the application view definition. You will find that information in this section.

At each point in *Using the Application Integration Design Console* where you need to refer to this document, you will see a note that directs you to a section in your adapter user guide, with a link to the edocs page for adapters. The following road map illustration shows where you need to refer from *Using the Application Integration Design Console* to this document.

Figure 3-1 Information Interlock with ***Using the Application Integration Design Console***


See Your Adapter User Guide
See Your Adapter User Guide
Log On
Application Context
Add Folders
Define App View
Connections
Add Services Add Events
Final Configuration
Test Services Test Events
Publish App View
See Your Adapter User Guide
See Your Adapter User Guide

Before You Begin

Before you define an application view, make sure you have:

- Installed and deployed the adapter according to the instructions in *BEA WebLogic Adapter for ClarifyCRM Installation and Configuration Guide*.
- Obtained the sample files for the Adapter for ClarifyCRM as described in the *BEA WebLogic Adapter for ClarifyCRM Installation and Configuration Guide*.
- Determined which business processes need to be supported by the application view. The required business processes determine the types of services and events you include in your

application views. Therefore, you must gather information about the application's business requirements from the business analyst. Once you determine the necessary business processes, you can define and test the appropriate services and events. For more information, see "Getting Started With the Adapter for ClarifyCRM" on page 1-5.

- Generated schemas and repositories for the ClarifyCRM services and events you wish to use with the adapter. To learn more about generating schemas, see Chapter 2, "Creating Schemas and Repositories."
- Gathered the connection information for your ClarifyCRM system.
- Prepared your ClarifyCRM database for events. To learn more about preparing your ClarifyCRM database, see "Obtain Samples and Run Database Scripts" in the *BEA WebLogic Adapter for ClarifyCRM Installation and Configuration Guide*.

About Application Views

An application view defines:

- Connection information for the EIS, including login information, connection settings, and so on.
- Service invocations, including the information the EIS requires for this request, as well as the request and response schemas associated with the service.
- Event notifications, including the information the EIS publishes and the event schema for inbound messages.

Typically, an application view is configured for a single business purpose and contains only the services and events required for that purpose. An EIS might have multiple application views, each defined for a different purpose.

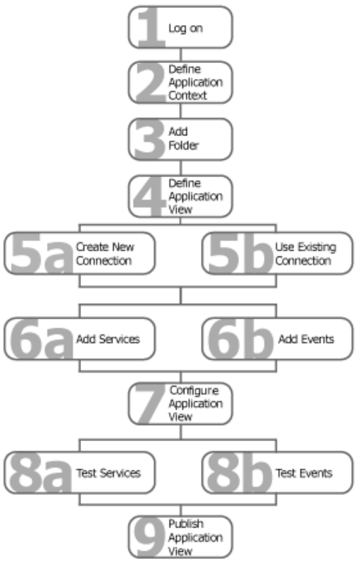
About Defining Application Views

Defining an application view is a multi-step process described in *Using the Application Integration Design Console*, available at the following URL:

`http://edocs.bea.com/wli/docs81/aiuser/index.html`

The information you enter depends on the requirements of your business process and your EIS system configuration. Figure 3-2 summarizes the procedure for defining and configuring an application view.

Figure 3-2 Process for Defining and Configuring an Application View


1 Log on
2 Define Application Context
3 Add Folder
4 Define Application View
5a Create New Connection
5b Use Existing Connection
6a Add Services
6b Add Events
7 Configure Application View
8a Test Services
8b Test Events
9 Publish Application View

To define an application view:

1. Log on to the WebLogic Integration Application View Console.
2. Define the application context by selecting an existing application or specifying a new application name and root directory.

   This application will be using the events and services you define in your application view. The application view works within the context of this application.
3. Add folders as required to help you organize application views.
4. Define a new application view for your adapter.
5. Add a new connection service or select an existing one.

   If you are adding a new connection service, see "Defining Service Connection Parameters" on page 3-5 for details about ClarifyCRM requirements.
6. Add the events and services for this application view.

   See the following sections for details about ClarifyCRM requirements:

   – "Setting Service Properties" on page 3-6

- "Setting Event Properties" on page 3-8

7. Perform final configuration tasks.

   If you are adding an event connection, see "Defining Event Connection Parameters" on page 3-10 for details about ClarifyCRM requirements.

8. Test all services and events to make sure they can properly interact with the target ClarifyCRM system.

   See the following sections for details about ClarifyCRM requirements:

   - "Testing Services" on page 3-12
   - "Testing Events Using a Service" on page 3-13
   - "Testing Events Manually" on page 3-14

9. Publish the application view to the target WebLogic Workshop application.

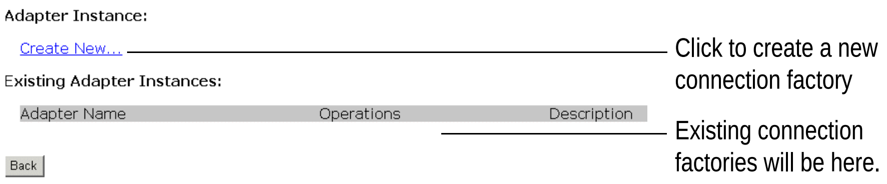
   This is the application you specified in step 2. Publishing the application view allows workflow developers within the target application to interact with the newly published application view using an Application View control.

Defining Service Connection Parameters

This information applies to "Step 5A, Create a New Browsing Connection" in *Using the Application Integration Design Console*, at the following URL:

http://edocs.bea.com/wli/docs81/aiuser/index.html

The Select Browsing Connection page allows you to choose the type of connection factory to associate with the application view. You can select a connection factory within an existing instance of the adapter or create a connection factory within a new adapter instance.

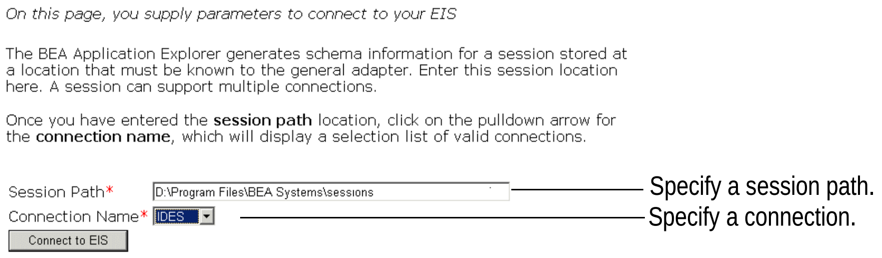

Adapter Instance:
Create New...
Existing Adapter Instances:
Adapter Name
Operations
Description
Back
Click to create a new connection factory
Existing connection factories will be here.

After you enter a connection name and description, you use the Configure Connection Parameters page to specify connection parameters for a connection factory.

To create a new browsing connection:

1. In the Create New Browsing Connections page, enter a connection name and description as described in *Using the Application Integration Design Console*.

   The Configure Connection Parameters page appears to allow you to configure the newly created connection factory within the new adapter instance.



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:\Program Files\BEA Systems\sessions

Connection Name* IDES

Note: A red asterisk (*) indicates that a field is required.

2. Specify a session path and connection name.

   This information enables the application view to interact with the target ClarifyCRM system. You need enter this information only once per application view.

3. Click Connect to EIS.

   You return to the Create New Browsing Connections, where you can specify connection pool parameters and logging levels. For more information, see *Using the Application Integration Design Console* at the following URL:

   http://edocs.bea.com/wli/docs81/aiuser/index.html

Setting Service Properties

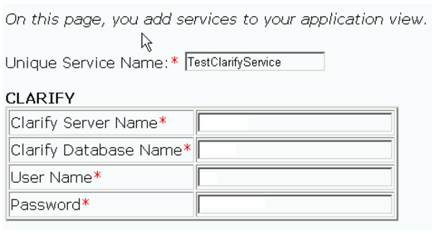
This information applies to “Step 6A, Add a Service to an Application View” in *Using the Application Integration Design Console*, at the following URL:

http://edocs.bea.com/wli/docs81/aiuser/index.html

Adapter for ClarifyCRM uses services to make requests of the ClarifyCRM system. A service consists of both a request and a response.

To configure a service:

1. Enter a unique service name that describes the function the service performs.



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

Unique Service Name:* TestClarifyService

CLARIFY

Clarify Server Name*	
Clarify Database Name*	
User Name*	
Password*	

Note: A red asterisk (*) indicates that a field is required.

2. Enter the following information:

Table 3-1 Service Parameters

Parameter	Description
Clarify Server Name	The name of the server hosting the ClarifyCRM system.
Clarify Database Name	The name of the database that you are using to store your ClarifyCRM data.
username	Your ClarifyCRM user name
password	The corresponding password for your ClarifyCRM user name

3. See "Common Service and Event Settings" on page 3-7 for information about selecting a schema and configuring logging and tracing.

Common Service and Event Settings

1 2 3 4 5 6 7 8 9

This information applies to "Step 6A, Add a Service to an Application View" in *Using the Application Integration Design Console*, at the following URL:

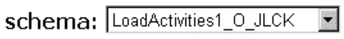
`http://edocs.bea.com/wli/docs81/aiuser/index.html`

You select a schema and select logging options the same way for all services.

To set common service settings:

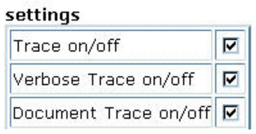
1. In the Schema list, select the schema you want to use with this service.

   For more information, see Chapter 2, "Creating Schemas and Repositories."


schema: LoadActivities1_O_JLCK

2. Configure logging and tracing for this service, as follows:

   Logging captures information from your adapter and writes it in a log file. Tracing displays runtime information in the console. You set the type and amount of information you wish to capture as part of the final configuration tasks. This is described in detail in *Using the Application Integration Design Console*.

   
   settings

Trace on/off	✓
Verbose Trace on/off	✓
Document Trace on/off	✓

   a. Select the Trace on/off check box to enable tracing for this service. Trace information appears in the runtime console.

   b. Select the Verbose Trace on/off check box to enable more detailed tracing for this service. Trace information appears in the runtime console.

   c. Select the Document trace check box to enable recording of additional trace information for deeper troubleshooting.

3. Click Add to add the service.

   For more information about the next step, see *Using the Application Integration Design Console* at the following URL:

   `http://edocs.bea.com/wli/docs81/aiuser/index.html`

Setting Event Properties

This information applies to "Step 6B, Add an Event to an Application View" in *Using the Application Integration Design Console*, at the following URL:

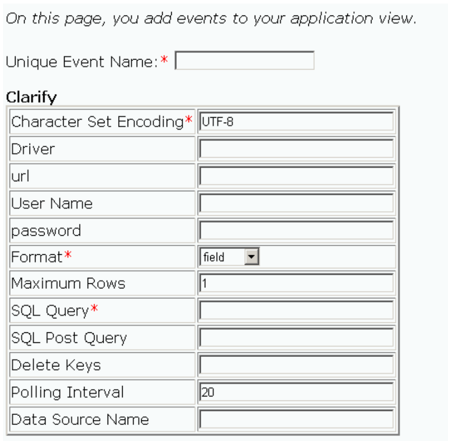
`http://edocs.bea.com/wli/docs81/aiuser/index.html`

An event defines how your application responds to events generated by ClarifyCRM.

To configure an event:

1. Enter a unique event name that describes the function the event performs.

   The Add Events page displays the fields this event requires.



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

Unique Event Name:*

Clarify

Character Set Encoding*	UTF-8
Driver	
url	
User Name	
password	
Format*	field
Maximum Rows	1
SQL Query*	
SQL Post Query	
Delete Keys	
Polling Interval	20
Data Source Name	

2. Specify the event's properties, which are described in the following table. Required values are indicated on the screen with an asterisk (*).

Table 3-2 Event Properties

Property	Description
Character Set Encoding	Sets the character set encoding to be used. (The default is UTF-8.)
Driver	The name of the JDBC driver that the adapter should use when querying the ClarifyCRM event table. Note that ClarifyCRM supports storing data in both Microsoft SQL Server and Oracle databases, so be sure to specify the JDBC driver for the DBMS your ClarifyCRM environment uses.
url	URL for the JDBC driver to access the database.
User Name	User name for the JDBC driver to access the database.
Password	Database password for user.
Format	Style of XML document containing results of the SQL query. The value is Column or Field.
Maximum Rows	Maximum number of rows to include in each document.

Table 3-2 Event Properties (Continued)

Property	Description
SQL_Query	The query that the BEA WebLogic Adapter for ClarifyCRM will issue against the event table. By default, the value is `SELECT * FROM IXTE_EVENTS`, but you can specify a different query so that the event captures the data you desire.
SQL Post Query	SQL statement issued after the query. If omitted, `delete <fields> from table where <field values> or <keys>` is used.
Delete Keys	Comma-separated list of keys to be used to build delete statement. If omitted, all fields are used. Do not use if a listener exit is provided. Case sensitive.
Polling Interval	Indicates how often, in seconds, the adapter should issue the SQL query. The higher this value, the longer the interval, and so the fewer system resources are used. The default value is 20 seconds.
Data Source Name	The name of the data source. Provide either the data source name or the driver and URL to establish the event connection.

3. See "Common Service and Event Settings" on page 3-7 for information about selecting a schema and configuring logging and tracing.

Defining Event Connection Parameters

This information applies to "Step 7, Perform Final Configuration Tasks" in *Using the Application Integration Design Console*, at the following URL:

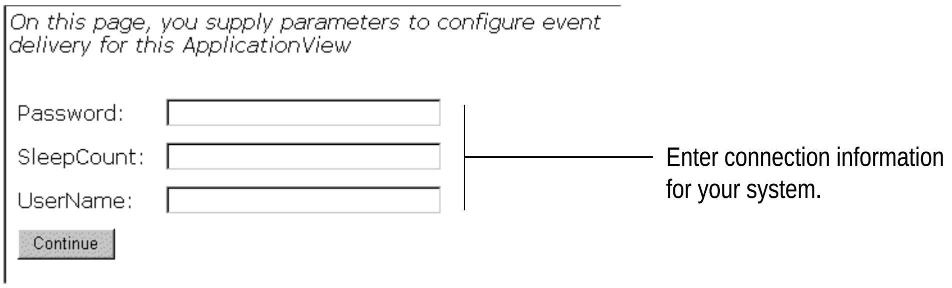
`http://edocs.bea.com/wli/docs81/aiuser/index.html`

Once you have finished adding services and events and have saved your application view, you must perform some final configuration tasks, including configuring event delivery connections, before testing the services and events. You perform these configuration tasks from the Final Configuration and Testing page.

To define event connection parameters:

1. In Connections area on the Application View Administration page, click Select/Edit.
2. In the Event Connection area, click Event to edit the default event connection.

   The Configure Event Delivery Parameters page appears.



On this page, you supply parameters to configure event delivery for this ApplicationView

Password:

SleepCount:

UserName:

Note: A red asterisk (*) indicates that a field is required.

3. Enter the following information:

Table 3-3 Event Connection Parameters

Parameter	Description
username	Your WebLogic Server Administration Console user name, defined in the startWebLogic script
password	The password for your WebLogic Server Administration Console user name
SleepCount	The number of seconds the adapter will wait between polling for events

The event delivery parameters you enter on this page enable connection to your ClarifyCRM system and are used when generating events. The parameters are specific to the associated adapter and are defined in the `wli-ra.xml` file within the base adapter.

4. Click Save to save your event delivery parameter settings. Click Continue to return to the Edit Event Adapter page, and then click Back to return to the Final Configuration and Testing page.

   The Edit Event Adapter page allows you to define event parameters and configure the information that will be logged for the connection factory. Select one of the following settings for the log:

   – Log errors and audit messages

- Log warnings, errors, and audit messages
- Log informational, warning, error, and audit messages
- Log all messages

Note: For maximum tracing, select Log all Messages. This is the recommended setting to use when you are collecting debugging information for BEA support.

The table that follows describes the type of information that each logging message contains.

Table 3-4 Logging message categories

This type of message	Contains
Audit	Extremely important information related to the business processing performed by an adapter.
Error	Information about an error that has occurred in the adapter, which may affect system stability.
Warning	Information about a suspicious situation that has occurred. Although this is not an error, it could have an impact on adapter operation.
Information	Information about normal adapter operations.

Testing Services

1 2 3 4 5 6 7 8 9

This information applies to "Step 8A, Test an Application View's Services" in *Using the Application Integration Design Console*, at the following URL:

`http://edocs.bea.com/wli/docs81/aiuser/index.html`

The purpose of testing an application view service is to evaluate whether that service interacts properly with the target ClarifyCRM system. When you test a service, you supply any inputs required to start the service. For the Adapter for ClarifyCRM, the input is in the form of a valid XML string (from the ClarifyCRM XML files created when you created schema) that acts as input for the service.

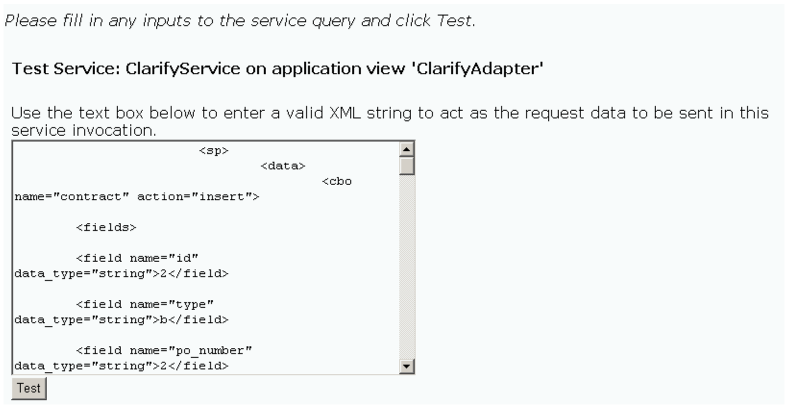
Note: You can test an application view only if it is deployed and only if it contains at least one event or service.

To test a service:

1. In the Application View Administration page, click the Test link beside the service to be tested.

   The Test Services page appears.

2. In the Test Service window, copy the appropriate XML strings from the ClarifyCRM XML file.



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

Test Service: ClarifyService on application view 'ClarifyAdapter'

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

```
                          <sp>
                                  <data>
                                          <cbo
name="contract" action="insert">

        <fields>

        <field name="id"
data_type="string">2</field>

        <field name="type"
data_type="string">b</field>

        <field name="po_number"
data_type="string">2</field>
```

3. Click Test.

   The results appear in the Test Results window.

Testing Events Using a Service

This information applies to "Step 8B, Test an Application View's Events" in *Using the Application Integration Design Console*, at the following URL:

http://edocs.bea.com/wli/docs81/aiuser/index.html

The purpose of testing an application view event is to make sure that the adapter correctly handles events generated by ClarifyCRM. When you test an event, you can trigger the event using a service or manually.

Note: You can test an application view only if it is deployed and only if it contains at least one event or service.

To test an event:

1. In the Application View Administration page, click the Test link beside the service to be tested.

The Test Events page appears.

2. Click Service and select a service that triggers the event you are testing.
3. In the Time field, enter a reasonable period of time to wait, specified in milliseconds, before the test times out (One second = 1000 milliseconds. One minute = 60,000 milliseconds.).
4. Click Test and enter the XML string needed to trigger the service.

   The service is executed.

   – If the test succeeds, the Test Result page appears, showing the event document, the service input document, and the service output document.

   – If the test fails, the Test Result page displays only a Timed Out message.

Testing Events Manually

This information applies to "Step 8B, Test an Application View's Events" in *Using the Application Integration Design Console*, at the following URL:

`http://edocs.bea.com/wli/docs81/aiuser/index.html`

To test an event manually:

1. In the Time field, enter a reasonable period of time to wait, specified in milliseconds, before the test times out (One second = 1000 milliseconds. One minute = 60,000 milliseconds.).
2. Click Test. The test waits for an event to trigger it.
3. Using the triggering ClarifyCRM application, perform an action that executes the service that, in turn, tests the application view event.

   – If the test succeeds, the Test Result page appears. This page displays the event document from the application, the service input document, and the service output document.

   – If the test fails or takes too long, the Test Result page appears, showing a Timed Out message.

Creating Application Views for the ClearBasic Interface

The information in this chapter applies only to the ClearBasic interface of the adapter for ClarifyCRM. To learn more about creating application views for the CBO interface of the adapter for ClarifyCRM, see Chapter 3, "Creating Application Views for the CBO Interface."

This section provides information on application views and deployed services and events, and includes the following topics:

How to Use This Document

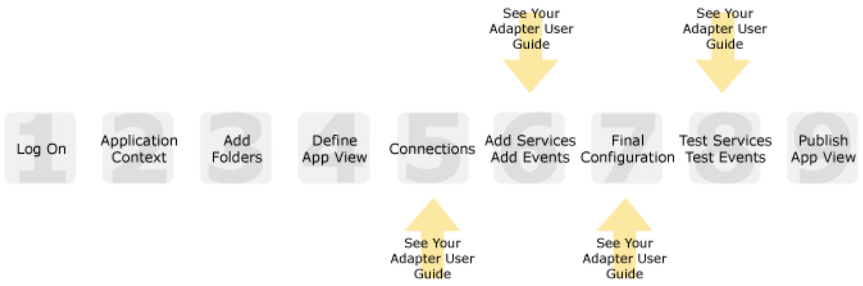
This document is designed to be used in conjunction with *Using the Application Integration Design Console*, available at the following URL:

http://edocs.bea.com/wli/docs81/aiuser/index.html

Using the Application Integration Design Console describes, in detail, the process of defining an application view, which is a key part of making an adapter available to process designers and other users. What *Using the Application Integration Design Console* does *not* cover is the specific information—about connections to your ClarifyCRM system, as well as supported services and events—that you must supply as part of the application view definition. You will find that information in this section.

At each point in *Using the Application Integration Design Console* where you need to refer to this document, you will see a note that directs you to a section in your adapter user guide, with a link to the edocs page for adapters. The following road map illustration shows where you need to refer from *Using the Application Integration Design Console* to this document.

Figure 4-1 Information Interlock with ***Using the Application Integration Design Console***


See Your Adapter User Guide
See Your Adapter User Guide
Log On
Application Context
Add Folders
Define App View
Connections
Add Services Add Events
Final Configuration
Test Services Test Events
Publish App View
See Your Adapter User Guide
See Your Adapter User Guide

Before You Begin

Before you define an application view, make sure you have:

- Installed and deployed the adapter according to the instructions in *BEA WebLogic Adapter for ClarifyCRM Installation and Configuration Guide*.
- Obtained the sample files for the Adapter for ClarifyCRM as described in the *BEA WebLogic Adapter for ClarifyCRM Installation and Configuration Guide*.
- Determined which business processes need to be supported by the application view. The required business processes determine the types of services and events you include in your

application views. Therefore, you must gather information about the application's business requirements from the business analyst. Once you determine the necessary business processes, you can define and test the appropriate services and events. For more information, see "Getting Started With the Adapter for ClarifyCRM" on page 1-5.

- Generated schemas and repositories for the ClarifyCRM services and events you wish to use with the adapter. To learn more about generating schemas, see Chapter 2, "Creating Schemas and Repositories."
- Gathered the connection information for your ClarifyCRM system.
- Prepared your ClarifyCRM database for events. To learn more about preparing your ClarifyCRM database, see "Obtain Samples and Run Database Scripts" in the *BEA WebLogic Adapter for ClarifyCRM Installation and Configuration Guide*.

About Application Views

An application view defines:

- Connection information for the EIS, including login information, connection settings, and so on.
- Service invocations, including the information the EIS requires for this request, as well as the request and response schemas associated with the service.
- Event notifications, including the information the EIS publishes and the event schema for inbound messages.

Typically, an application view is configured for a single business purpose and contains only the services and events required for that purpose. An EIS might have multiple application views, each defined for a different purpose.

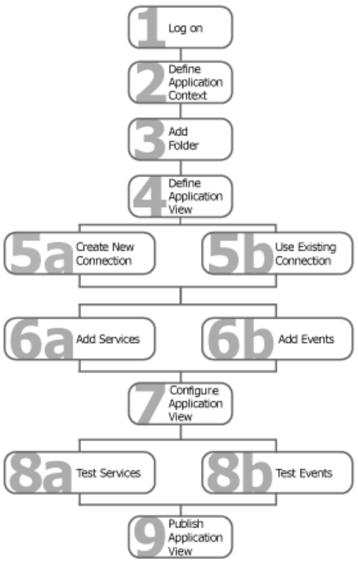
About Defining Application Views

Defining an application view is a multi-step process described in *Using the Application Integration Design Console*, available at the following URL:

`http://edocs.bea.com/wli/docs81/aiuser/index.html`

The information you enter depends on the requirements of your business process and your EIS system configuration. Figure 4-2 summarizes the procedure for defining and configuring an application view.

Figure 4-2 Process for Defining and Configuring an Application View


1 Log on
2 Define Application Context
3 Add Folder
4 Define Application View
5a Create New Connection
5b Use Existing Connection
6a Add Services
6b Add Events
7 Configure Application View
8a Test Services
8b Test Events
9 Publish Application View

To define an application view:

1. Log on to the WebLogic Integration Application View Console.
2. Define the application context by selecting an existing application or specifying a new application name and root directory.

   This application will be using the events and services you define in your application view. The application view works within the context of this application.

3. Add folders as required to help you organize application views.
4. Define a new application view for your adapter.
5. Add a new connection service or select an existing one.

   If you are adding a new connection service, see "Defining Service Connection Parameters" on page 4-5 for details about ClarifyCRM requirements.

6. Add the events and services for this application view.

   See the following sections for details about ClarifyCRM requirements:

   – "Setting Service Properties" on page 4-6

- "Setting Event Properties" on page 4-9

7. Perform final configuration tasks.

   If you are adding an event connection, see "Defining Event Connection Parameters" on page 4-10 for details about ClarifyCRM requirements.

8. Test all services and events to make sure they can properly interact with the target ClarifyCRM system.

   See the following sections for details about ClarifyCRM requirements:

   - "Testing Services" on page 4-12
   - "Testing Events Using a Service" on page 4-13
   - "Testing Events Manually" on page 4-14

9. Publish the application view to the target WebLogic Workshop application.

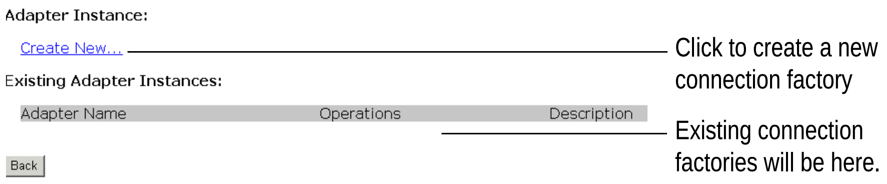
   This is the application you specified in step 2. Publishing the application view allows workflow developers within the target application to interact with the newly published application view using an Application View control.

Defining Service Connection Parameters

This information applies to "Step 5A, Create a New Browsing Connection" in *Using the Application Integration Design Console*, at the following URL:

`http://edocs.bea.com/wli/docs81/aiuser/index.html`

The Select Browsing Connection page allows you to choose the type of connection factory to associate with the application view. You can select a connection factory within an existing instance of the adapter or create a connection factory within a new adapter instance.

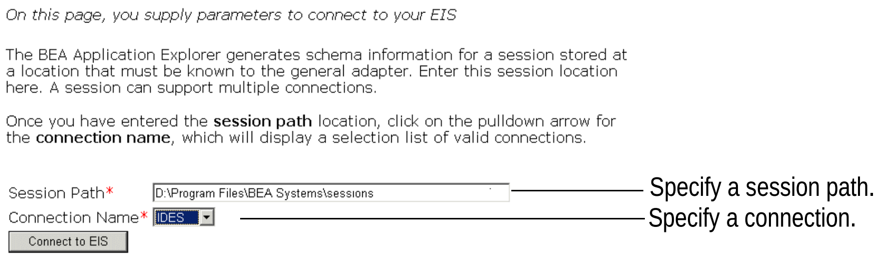

Adapter Instance:
Create New...
Existing Adapter Instances:
Adapter Name
Operations
Description
Back
Click to create a new connection factory
Existing connection factories will be here.

After you enter a connection name and description, you use the Configure Connection Parameters page to specify connection parameters for a connection factory.

To create a new browsing connection:

1. In the Create New Browsing Connections page, enter a connection name and description as described in *Using the Application Integration Design Console*.

   The Configure Connection Parameters page appears to allow you to configure the newly created connection factory within the new adapter instance.



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:\Program Files\BEA Systems\sessions

Connection Name* IDES

   Note: A red asterisk (*) indicates that a field is required.

2. Specify a session path and connection name.

   This information enables the application view to interact with the target ClarifyCRM system. You need enter this information only once per application view.

3. Click Connect to EIS.

   You return to the Create New Browsing Connections, where you can specify connection pool parameters and logging levels. For more information, see *Using the Application Integration Design Console* at the following URL:

   `http://edocs.bea.com/wli/docs81/aiuser/index.html`

Setting Service Properties

This information applies to “Step 6A, Add a Service to an Application View” in *Using the Application Integration Design Console*, at the following URL:

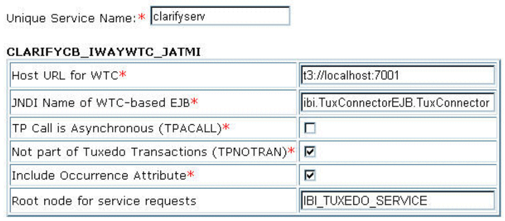
`http://edocs.bea.com/wli/docs81/aiuser/index.html`

Adapter for ClarifyCRM uses services to make requests of the ClarifyCRM system. A service consists of both a request and a response.

To configure a service:

1. Enter a unique service name that describes the function the service performs.

The Add Services page displays the fields required for this service type.



Unique Service Name:*	clarifyserv

CLARIFYCB_IWAYWTC_JATMI

Host URL for WTC*	t3://localhost:7001
JNDI Name of WTC-based EJB*	ibi.TuxConnectorEJB.TuxConnector
TP Call is Asynchronous (TPACALL)*	☐
Not part of Tuxedo Transactions (TPNOTRAN)*	☑
Include Occurrence Attribute*	☑
Root node for service requests	IBI_TUXEDO_SERVICE

Note: A red asterisk (*) indicates that a field is required.

2. Enter the following information:

Table 4-1 Service Properties

Property	Description
Host URL for WTC	URL of the WebLogic system hosting the WebLogic Tuxedo Connector (WTC).
JNDI Name of WTC-based EJB	Java Naming Service entry for WTC-based ATMI Enterprise Java Bean.
TP Call is Asynchronous (TPACALL)	A flag to determine whether the call to Tuxedo is done synchronously (the default) or asynchronously.
Not part of Tuxedo Transactions (TPNOTRAN)	A flag to determine whether this call to Tuxedo is done outside the current transaction. By default, the call is done withing the current transaction. To learn more about Tuxedo transactions, see your Tuxedo documentation.
Include Occurrence Attribute	If this field is checked, the occurrence attribute is set to true. This means that a request document is expected to have occurrence attribute set for every field entry. If the occurrence attribute is true and the request document is not present, it is automatically set to 0. The response document will have occurrence attributes set. If this field is not checked, a request document is not expected to have occurrence attributes set. The natural order of the fields becomes the occurrence number, and occurrence attributes are not generated for the response document. The default value for this field is false.

Table 4-1 Service Properties

Property	Description
Root Node for Service Requests	This property specifies the root node for service request.
schema	The name of the schema that describes the service.

3. See "Common Service and Event Settings" on page 4-8 for information about selecting a schema and configuring logging and tracing.

Common Service and Event Settings

This information applies to "Step 6A, Add a Service to an Application View" in *Using the Application Integration Design Console*, at the following URL:

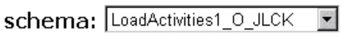
http://edocs.bea.com/wli/docs81/aiuser/index.html

You select a schema and select logging options the same way for all services.

To set common service settings:

1. In the Schema list, select the schema you want to use with this service.

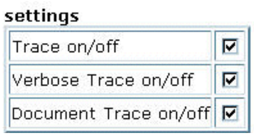
   For more information, see Chapter 2, "Creating Schemas and Repositories."



schema: LoadActivities1_O_JLCK

2. Configure logging and tracing for this service, as follows:

   Logging captures information from your adapter and writes it in a log file. Tracing displays runtime information in the console. You set the type and amount of information you wish to capture as part of the final configuration tasks. This is described in detail in *Using the Application Integration Design Console*.



settings

Trace on/off	✓
Verbose Trace on/off	✓
Document Trace on/off	✓

   a. Select the Trace on/off check box to enable tracing for this service. Trace information appears in the runtime console.

   b. Select the Verbose Trace on/off check box to enable more detailed tracing for this service. Trace information appears in the runtime console.

   c. Select the Document trace check box to enable recording of additional trace information for deeper troubleshooting.

3. Click Add to add the service.

   For more information about the next step, see *Using the Application Integration Design Console* at the following URL:

   http://edocs.bea.com/wli/docs81/aiuser/index.html

Setting Event Properties

This information applies to “Step 6B, Add an Event to an Application View” in *Using the Application Integration Design Console*, at the following URL:

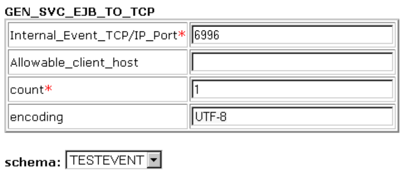
http://edocs.bea.com/wli/docs81/aiuser/index.html

An event defines how your application responds to events generated by ClarifyCRM.

To configure an event:

1. Enter a unique event name that describes the function the event performs.

   The Add Events page displays the fields this event requires.



GEN_SVC_EJB_TO_TCP

Internal_Event_TCP/IP_Port*	6996
Allowable_client_host	
count*	1
encoding	UTF-8

schema: TESTEVENT

   Note: A red asterisk (*) indicates that a field is required.

2. Specify the event’s properties, which are described in the following table. Required values are indicated on the screen with an asterisk (*).

Table 4-2 Event Properties

Property	Description
Unique Event Name	The name of the event that you are adding. The name must be unique within the application view. Valid characters include a-z, A-Z, 0-9, and underscore (_).
Internal_Event_TCP/IP_Port	The TCP port to which Tuxedo events are posted into WebLogic Integration.
Allowable Client Host	The TCP/IP host name or address of WTC ATMI EJB clients sending events into WebLogic Integration.
count	The number of simultaneous threads created to service Tuxedo event requests.
encoding	Sets the character set encoding to be used. (The default is UTF-8.)
schema	The name of the schema that describes this event.

3. See "Common Service and Event Settings" on page 4-8 for information about selecting a schema and configuring logging and tracing.

Defining Event Connection Parameters

This information applies to "Step 7, Perform Final Configuration Tasks" in *Using the Application Integration Design Console*, at the following URL:

http://edocs.bea.com/wli/docs81/aiuser/index.html

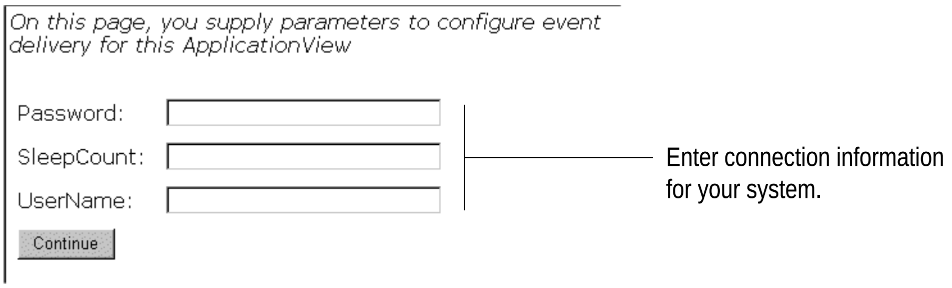
Once you have finished adding services and events and have saved your application view, you must perform some final configuration tasks, including configuring event delivery connections, before testing the services and events. You perform these configuration tasks from the Final Configuration and Testing page.

To define event connection parameters:

1. In Connections area on the Application View Administration page, click Select/Edit.

2. In the Event Connection area, click Event to edit the default event connection.

The Configure Event Delivery Parameters page appears.



On this page, you supply parameters to configure event delivery for this ApplicationView

Password:	
SleepCount:	
UserName:	

Note: A red asterisk (*) indicates that a field is required.

3. Enter the following information:

Table 4-3 Event Connection Parameters

Parameter	Description
username	Your WebLogic Server Administration Console user name, defined in the startWebLogic script
password	The password for your WebLogic Server Administration Console user name
SleepCount	The number of seconds the adapter will wait between polling for events

The event delivery parameters you enter on this page enable connection to your ClarifyCRM system and are used when generating events. The parameters are specific to the associated adapter and are defined in the `wli-ra.xml` file within the base adapter.

4. Click Save to save your event delivery parameter settings. Click Continue to return to the Edit Event Adapter page, and then click Back to return to the Final Configuration and Testing page.

   The Edit Event Adapter page allows you to define event parameters and configure the information that will be logged for the connection factory. Select one of the following settings for the log:

   – Log errors and audit messages

   – Log warnings, errors, and audit messages

   – Log informational, warning, error, and audit messages

   – Log all messages

Note: For maximum tracing, select Log all Messages. This is the recommended setting to use when you are collecting debugging information for BEA support.

The table that follows describes the type of information that each logging message contains.

Table 4-4 Logging message categories

This type of message	Contains
Audit	Extremely important information related to the business processing performed by an adapter.
Error	Information about an error that has occurred in the adapter, which may affect system stability.
Warning	Information about a suspicious situation that has occurred. Although this is not an error, it could have an impact on adapter operation.
Information	Information about normal adapter operations.

Testing Services

This information applies to "Step 8A, Test an Application View's Services" in *Using the Application Integration Design Console*, at the following URL:

`http://edocs.bea.com/wli/docs81/aiuser/index.html`

The purpose of testing an application view service is to evaluate whether that service interacts properly with the target ClarifyCRM system. When you test a service, you supply any inputs required to start the service. For the Adapter for ClarifyCRM, the input is in the form of a valid XML string (from the ClarifyCRM XML files created when you created schema) that acts as input for the service.

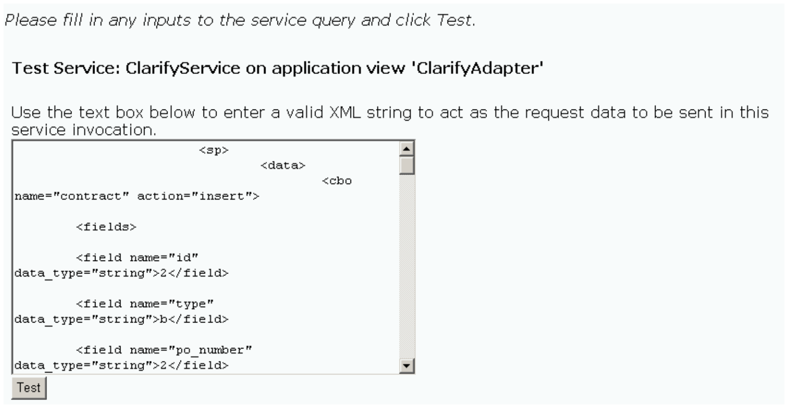
Note: You can test an application view only if it is deployed and only if it contains at least one event or service.

To test a service:

1. In the Application View Administration page, click the Test link beside the service to be tested.

   The Test Services page appears.

2. In the Test Service window, copy the appropriate XML strings from the ClarifyCRM XML file.



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

Test Service: ClarifyService on application view 'ClarifyAdapter'

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

```
                        <sp>
                                <data>
                                        <cbo
name="contract" action="insert">

        <fields>

        <field name="id"
data_type="string">2</field>

        <field name="type"
data_type="string">b</field>

        <field name="po_number"
data_type="string">2</field>
```

3. Click Test.

   The results appear in the Test Results window.

Testing Events Using a Service

1 2 3 4 5 6 7 **8** 9

This information applies to "Step 8B, Test an Application View's Events" in *Using the Application Integration Design Console*, at the following URL:

http://edocs.bea.com/wli/docs81/aiuser/index.html

The purpose of testing an application view event is to make sure that the adapter correctly handles events generated by ClarifyCRM. When you test an event, you can trigger the event using a service or manually.

Note: You can test an application view only if it is deployed and only if it contains at least one event or service.

To test an event:

1. In the Application View Administration page, click the Test link beside the service to be tested.

   The Test Events page appears.

2. Click Service and select a service that triggers the event you are testing.

3. In the Time field, enter a reasonable period of time to wait, specified in milliseconds, before the test times out (One second = 1000 milliseconds. One minute = 60,000 milliseconds.).

4. Click Test and enter the XML string needed to trigger the service.

   The service is executed.

   – If the test succeeds, the Test Result page appears, showing the event document, the service input document, and the service output document.

   – If the test fails, the Test Result page displays only a Timed Out message.

Testing Events Manually

1 2 3 4 5 6 7 **8** 9

This information applies to “Step 8B, Test an Application View’s Events” in *Using the Application Integration Design Console*, at the following URL:

`http://edocs.bea.com/wli/docs81/aiuser/index.html`

To test an event manually:

1. In the Time field, enter a reasonable period of time to wait, specified in milliseconds, before the test times out (One second = 1000 milliseconds. One minute = 60,000 milliseconds.).

2. Click Test. The test waits for an event to trigger it.

3. Using the triggering ClarifyCRM application, perform an action that executes the service that, in turn, tests the application view event.

   – If the test succeeds, the Test Result page appears. This page displays the event document from the application, the service input document, and the service output document.

   – If the test fails or takes too long, the Test Result page appears, showing a Timed Out message.

Customizing Tuxedo Field Tables

This appendix applies to the ClearBasic interface of ClarifyCRM only.

Field tables are required to describe all the fields that the BEA WebLogic Adapter for ClarifyCRM will handle in service requests and responses. This section describes how to customize Tuxedo field tables, and includes the following topics:

- Working With Field Tables
- Creating a WTC FML32 Field Table
- Creating a Tuxedo Field Table Header File
- Using Your Field Table in WebLogic Tuxedo Connector
- Using Your Field Table in Tuxedo

Working With Field Tables

Field Manipulation Language (FML) field tables and their field definition files describe the fields available in any Tuxedo typed buffer exchanged between a service requestor and service responder. All fields to be exchanged between ClarifyCRM and WebLogic Integration, either as an inbound parameter, or as an outbound response field, must be defined in the field definition file. The field definition file defines the field identifier, the field name, and the field type to the Tuxedo system. The BEA WebLogic Adapter for ClarifyCRM ships with a field definition file, but you will want to customize the field definition file for your purposes.

The field definition files must match on both the Tuxedo and WebLogic Tuxedo Connector (WTC) side of the domain communication channel. The field definitions need not be identical, but the fields used between the two systems must exist in both field definition files and the fields must match in ID and type. The field definition file is used at design and configuration time and is converted into an FML field table class for execution in the WTC.

The format of the field definition file is the same on both the Tuxedo and WTC platforms. There are different utilities to use the FML field table from the source field definition files. On the Tuxedo platform, the command line utility `mkfldhdr` can be used to create the header file for a user's C-based ATMI application. On the WTC platform, the `mkfieldclass32` utility produces Java source for the field table that must be compiled for use on WebLogic WTC.

Creating a WTC FML32 Field Table

By default, your adapter makes use of a field table named `_SrvFieldsTable`. This will be the name used for the field definition file and the associated `.java` and `.class` files. To produce a field class source Java file from your field definition file, run the following Java program:

```
java weblogic/wtc/jatmi/mkfldclass32 _SrvFieldsTable
```

The output from this program will be a Java definition of the field class. This Java file must be compiled using the `javac` compile command in the Java Development Kit (JDK). For example:

```
javac _SrvFieldsTable.java
```

The class path must include the `weblogic.jar` library.

Creating a Tuxedo Field Table Header File

To produce a Tuxedo FML field table header file from a field definition file, run the following command line utility:

```
mkfldhdr [-d targetDirectory] [field_table ...]
```

The resulting field table header file can be found in the current directory, or (if you include the `-d` parameter) in the specified target directory.

Here:

- *`targetDirectory`* specifies that the output header files are to be created in a directory other than the present working directory.
- *`field_table`* is a field table name.

  If you omit field table names, the utility uses the `FIELDTBLS` environment variable as the list of field tables to be converted, and `FLDTBLDIR` environment variable as a list of directories to be searched for the files.

`FIELDTBLS` specifies a comma-separated list of field table filenames. If `FIELDTBLS` has no value, fld.tbl is used as the name of the (only) field table file (in this case, the resulting header file will be (`fld.tbl.h`). The `FLDTBLDIR` environment variable is a colon-separated list of directories in which to look for each field table whose name is not an absolute path name; the search for field tables is very similar to the search for executable commands using the UNIX System `PATH` variable. If `FLDTBLDIR` is not defined, only the current directory is searched. Thus, if no field table names are specified on the command line and `FIELDTBLS` and `FLDTBLDIR` are not set, mkfldhdr will convert the field table `fld.tbl` in the current directory into the header file `fld.tbl.h`.

Using Your Field Table in WebLogic Tuxedo Connector

Once your field class is compiled, you may override the `_SrvFieldsTable.class` file included in the supplied BEA WebLogic Adapter for ClarifyCRM Web application (.ear file) by moving or copying the class file into the `WEB-INF/classes` directory of your WebLogic domain. For example, placing the `_SrvFieldsTable.class` file in the directory:

```
/usr/local/bea/user_projects/yourdomain/applications/DefaultWebApp_myserve
r/WEB-INF/classes
```

For more information on using field tables in your WebLogic Tuxedo Connector (WTC) environment, see the *WebLogic Tuxedo Connector Programmer's Guide*.

Using Your Field Table in Tuxedo

In your Tuxedo environment, the following variables are used throughout Field Manipulation Language (FML) to access field table files:

- `FIELDTBLS`, which should contain a comma-separated list of field table files for the application. Files given as full path names are used as is; files listed as relative path names are searched for through the list of directories specified by the `FLDTBLDIR` variable. `FIELDTBLS32` is used for FML32. If `FIELDTBLS` is not set, then the single file name `fld.tbl` is used. (`FLDTBLDIR` still applies.)
- `FLDTBLDIR`, which specifies a colon-separated list of directories to be used to find field table files with relative filenames. Its usage is similar to the `PATH` environment variable. If `FLDTBLDIR` is not set or is null, then its value is taken to be the current directory. `FLDTBLDIR32` is used for FML32.

For more information, see the *Programming a BEA Tuxedo Application Using FML* manual.

Index