SIEBEL 7
eBusiness

SIEBEL EFINANCE FOR TELLER CONNECTOR TO IBM WEBSPHERE BUSINESS COMPONENT COMPOSER GUIDE

VERSION 7.0, REV. H

12-BCK98J

August 2002

Siebel Systems, Inc., 2207 Bridgepointe Parkway, San Mateo, CA 94404

Contents

Chapter 3. Teller IFX Connector Roadmap

Appendix A. Teller Objects

Introduction

This book will be useful primarily to people whose title or job description matches one of the following:

Business Analysts	Persons responsible for analyzing application integration challenges and planning integration solutions at an enterprise.
Siebel Financial Services Application Administrators	Persons responsible for planning, setting up, and maintaining Siebel Financial Services applications.
Siebel Financial Services Application Developers	Persons responsible for planning, implementing, and configuring Siebel Financial Services applications, and possibly adding new functionality.
Siebel Financial Services Integration Developers	Persons responsible for analyzing a business situation or using the analysis of a business analyst to build the integration solution at an enterprise for Siebel Financial Services applications.
Siebel System Administrators	Persons responsible for the whole system, including installing, maintaining, and upgrading Siebel products.
System Integrators	Persons responsible for analyzing a business situation or using the analysis of a business analyst to build the integration solution at an enterprise for specific applications, to develop custom solutions, or both.

Also included in the audience for this book is any user with experience in data integration, data transformation (data mapping), scripting or programming, and XML.

How This Guide Is Organized

This guide describes the Siebel Connector for Teller architecture as well as how to configure and use the Siebel Connector for Teller to integrate, share, and replicate data between Siebel Financial Services applications and external applications. Following is a list of documentation for Siebel Financial Services eBusiness Application Integration (eAI) and all the pre-built connectors.

- *Siebel Financial Services eBusiness Application Integration Guide*
- *Siebel Financial Services Connector for ACORD P&C and Surety Guide*
- *Siebel Financial Services Connector for IFX XML Guide*

For all other information about the Siebel Connector for Teller, refer to the DTD files located in the FINS Tools installation directory:
< *tools home directory* > \FINSEAI\TELLER\DTDV101.

Additional Resources

The product documentation set for Siebel Financial Services applications is provided on the *Siebel Financial Services Bookshelf* CD-ROM. For general information about Siebel product documentation, see the *Siebel Financial Services Bookshelf* Welcome page and *Siebel Financial Services Documentation Roadmap*.

EAI-Associated Books. Review *Overview: Siebel eBusiness Application Integration Volume I* and *Siebel Financial Services eBusiness Application Integration Guide* to learn what Siebel eAI is offering as a base product for Siebel Financial Services eAI. Refer to *Tools Online Help* if you plan on using COM, CORBA, or the ActiveX Plug-ins to accomplish integration. *Tools Online Help* serves as a reference for Siebel business objects and components. Read both *Siebel Tools Guide* and *Siebel Workflow Administration Guide* to gain an understanding of the tools required for creating integrations. Read the *Siebel Enterprise Integration Manager Administration Guide* if you will perform bulk loading or unloading of data. The Connector books provide specifics on each of the associated connectors.

Teller Domain Message Specification and Data Type Definition (DTD). The DTD files are located in the FINS Tools installation directory: < tools home directory > \FINSEAI\TELLER\DTDV101.

Interactive Financial Exchange (IFX) Specification. The Interactive Financial Exchange (IFX) Specification is managed by IFX Forum and is freely available at www.ifxforum.org. It provides a framework for the exchange of financial data and instructions independent of a particular network technology or computing platform. The information-sharing potential of IFX has been designed to support communication not only between a financial institution and its customers, but also between a financial institution and its service providers, between financial institutions, and eventually directly between customers.

For information on the XFS standard, go to www.xfsws.com.

Using the Siebel Product

It is strongly recommended that you read *Fundamentals* so that you can make optimal use of your Siebel application, especially if you are new to Siebel software. *Fundamentals* provides detailed coverage of the Siebel user interface and how to use it; working with data; locating information with the query and find features; sharing information with other users; and so on. The features presented in *Fundamentals* appear throughout the Siebel application suite; they are introduced through procedures you can learn and use in your own Siebel application.

Revision History

Siebel eFinance for Teller Connector to IBM WebSphere Business Component Composer Guide, Version 7.0, Rev. H

Overview: Siebel Teller Messages 1

This chapter describes Siebel Connector for Teller architecture and the required components for the IFX connector. It also includes information about IFX-XML and Siebel Teller messages, as well as information about how to modify Siebel Connector for Teller.

Siebel Teller Architecture

The Siebel Connector for Teller extends the functionality of the Siebel Connector for IFX XML to provide Teller-specific data exchange between Siebel and other systems.

A Siebel connector is a configured set of components that allow data to be exchanged between internal and external applications and databases. The Siebel software uses XML (Extensible Markup Language) to format the data so that it can move between systems with different data structures and be based on different database concepts.

The Siebel Connector for Teller is a set of teller-specific XML business messages that extend the capabilities of the Siebel Connector for IFX XML, as shown in Figure 1.

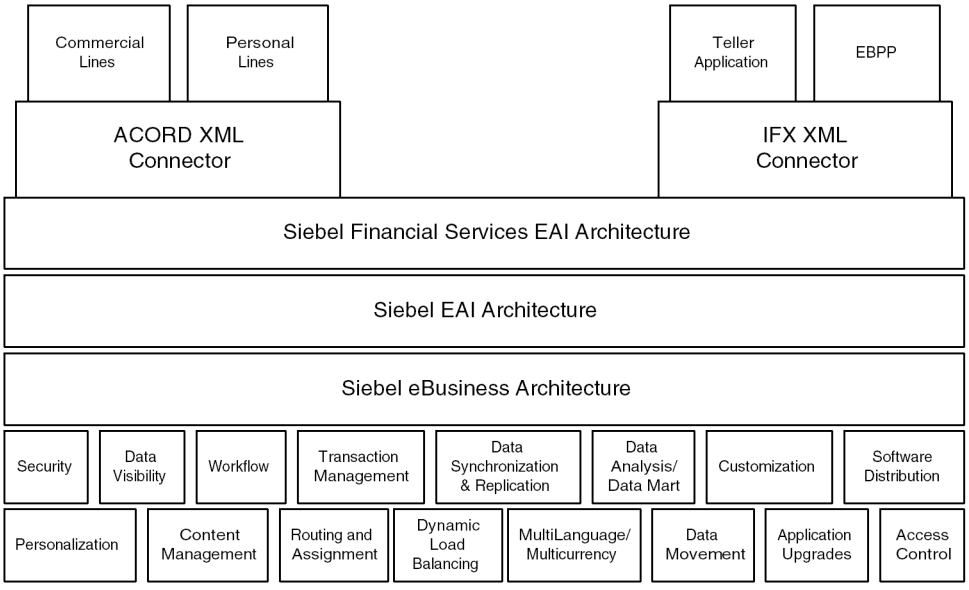


Figure 1. The Siebel Connector for Teller Extending the Siebel Connector for IFX XML

The Siebel Connector for Teller extends the functionality of the Siebel Connector for IFX XML, but makes no change to the IFX XML architecture. Figure 2 displays the basic connector architecture.

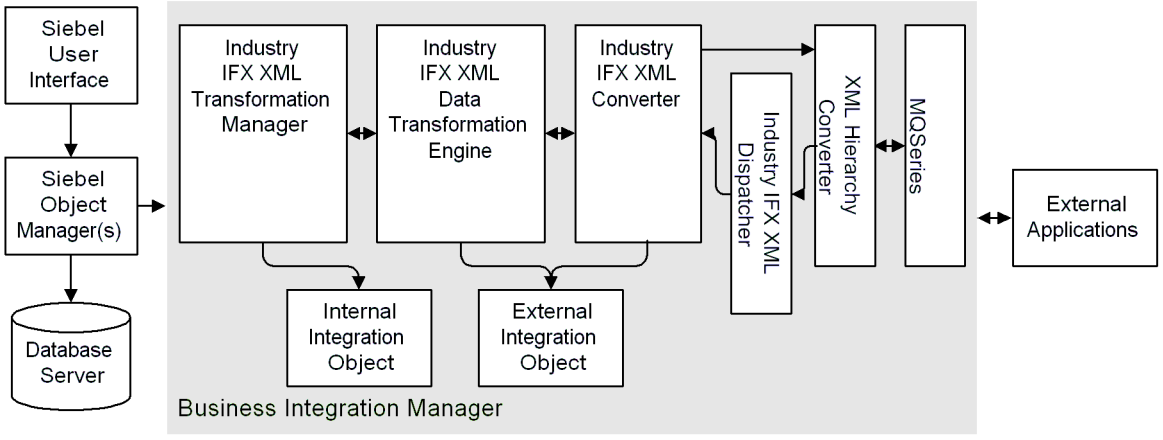


Figure 2. Connector Architecture

The basic structure of the Siebel Connector for Teller is exactly the same as the basic structure of the Siebel Connector for IFX XML. The methods and arguments used to configure the components are the same, with a few additional arguments for some of the methods. Use the IFX XML Wizard to create the necessary integration objects for the Siebel Connector for Teller.

The main difference between the Siebel Connector for Teller and the Siebel Connector for IFX XML is that you will use the Teller DTD instead of the IFX DTD when creating integration objects.

For information about the Siebel Connector for IFX XML, see *Siebel Financial Services Connector for IFX XML Guide*. The remaining chapters in this document assume that you have read and understood *Siebel Financial Services Connector for IFX XML Guide*.

Required Components

The IFX connector requires the following components in order to implement message exchanges between Siebel applications and IFX-compliant applications:

- Siebel Financial Services application.
- A license to use the Siebel IFX Connector.

  The IFX connector license key can be obtained from Siebel Manufacturing Operations. Please ask your technical account manager to obtain your license key, or log a service request on Siebel SupportWeb.

  NOTE: Siebel Connector for IFX XML is not automatically available as part of Siebel Financial Services, but must be purchased separately.

- Siebel Event Manager to initiate a workflow process through a Siebel workflow manager (optional). In the absence of Event Manager, an eScript can initiate a workflow process. Siebel Workflow is delivered as a part of Siebel Financial Services.

NOTE: You should also be familiar with IFX XML models. Additional information about these models can be obtained by visiting www.IFX.org.

This document assumes that all these products have been successfully installed and tested for completeness by trained personnel before starting to use the IFX Connector for integration.

IFX XML and Siebel Teller Messages

IFX is a financial industry definition of XML. It contains messages appropriate for the financial and banking industries. The Siebel Teller extension to the IFX XML message set provides additional messages for use in exchanging information from teller stations to non-Siebel databases.

The Teller XML standard defines the required structure and format of an XML message for use with an IFX converter. The definition exists in the Teller DTD, and the Teller DTD incorporates the IFX DTD to construct messages.

For a summary of the IFX XML syntax and rules, see *Siebel Financial Services Connector for IFX XML Guide.*

Teller XML Message Example

The following is a sample Teller-XML document.

```
<?xml version="1.0" encoding="UTF-8"?>
<?ifx version="1.0.1"  oldfileuid="" newfileuid="" ?>
<!DOCTYPE IFX SYSTEM "http://server/teller.dtd">
<IFX>
   <SignonRq>
      <SessKey>123-ABC2001-11-14T14:22:24.0-08:00</SessKey>
      <ClientDt>2001-11-14T14:22:30.0-08:00</ClientDt>
      <CustLangPref>ENU</CustLangPref>
      <ClientApp>
         <Org>ZKB</Org>
         <Name>Siebel</Name>
         <Version>1.0</Version>
      </ClientApp>
   </SignonRq>
   <tllr.TlrSvcRq>
      <SPName>Legacy System</SPName>
      <RqUID>4E583573-ADA8-41c3-A002-CB70EE6D2BAB</RqUID>
      <tllr.BrnOpenRq>
         <RqUID>534572C9-4C9B-4f1e-A8DE-69F3C9AA8874</RqUID>
         <tllr.BranchId>BRANCH001</tllr.BranchId>
      </tllr.BrnOpenRq>
   </tllr.TlrSvcRq>
</IFX>
```

Teller Domain Message Specification

The Teller message specification is an extension to the standard IFX XML specification. It is delivered as a separate DTD file that includes the IFX DTD. It was written based on IFX version 1.0.1. Both the Teller and IFX DTD files are located in the FINS Tools installation directory:
< *tools home directory* > \FINSEAI\TELLER\DTDV101.

Teller DTD

Use the Teller DTD with the IFX XML wizard to create an appropriate set of integration objects and properly update the Dispatcher Map.

Select the Teller DTD (teller.dtd) when using the IFX XML wizard to create a Siebel Connector for Teller.

Siebel Connector for Teller Configuration

A Siebel Connector for Teller consists of a combination of the Siebel technologies shown in Table 1.

Table 1. Siebel Connector for Teller Technologies

Component	Role
Runtime Events	Triggers the connector workflow.
Command Objects	Triggers the connector workflow.
Workflow Processes	Controls the flow of data between Siebel and the external system. Calls the various business services to perform their tasks.
Integration Objects	Stores data from Siebel (as internal integration objects) or from the XML message (as external integration objects).
Business Services	Perform various tasks to handle the transformation of data from Siebel to external system and back.

Table 2 displays a list of the major business services and their functions. The complete details of the business services are described in Chapter 2, "Teller IFX XML Connector."

Table 2. Business Services

Business Service	Function
FINS IFX XML Transaction Manager	Call a predefined business service (usually the Siebel Adapter) to extract data from Siebel and to update data in Siebel.
FINS IFX XML DTE	Transform integration objects from internal format to external format, and vice versa.
FINS IFX XML Converter	Manage the IFX session key and transform date formats.
FINS IFX XML Dispatcher	Handle incoming messages by identifying the body and header sections.

Modifying a Siebel Connector for Teller

In general, you will not need to create a Siebel Connector for Teller from scratch. This section describes how to modify the existing connector infrastructure to accommodate new and modified messages.

For an example of creating a connector, see "Configuration Roadmap" i*n Siebel Financial Services Connector for IFX XML Guide*.

To add a new message to a Siebel Connector for Teller

1. Run the IFX XML wizard to create internal and external integration objects.

   The external integration object is based on the DTD. Be sure to choose the teller.dtd file.

   The internal integration object is based on the business component.

2. Create a data transformation engine (DTE) map to map the integration object fields to each other.
3. Build a workflow or modify an existing one to handle the new message.
4. Add a button or event to an applet to trigger the workflow.
5. Test the workflow.

Sample Workflow

Table 3 presents a sample workflow for a Siebel Connector for Teller.

Table 3. Sample Workflow

Workflow	Description
FINS Teller Session Connector	Manages all Teller transaction messages from the Teller Session business component.

Teller IFX XML Connector 2

This chapter describes the methods, input arguments, and output arguments for configuring the components of a Siebel Connector for IFX XML.

The Siebel Connector for IFX XML consists of the following components:

- Transaction Manager
- Transformation Engine
- Converter
- Dispatcher
- Transport Adapter

Figure 3 shows the connector components.

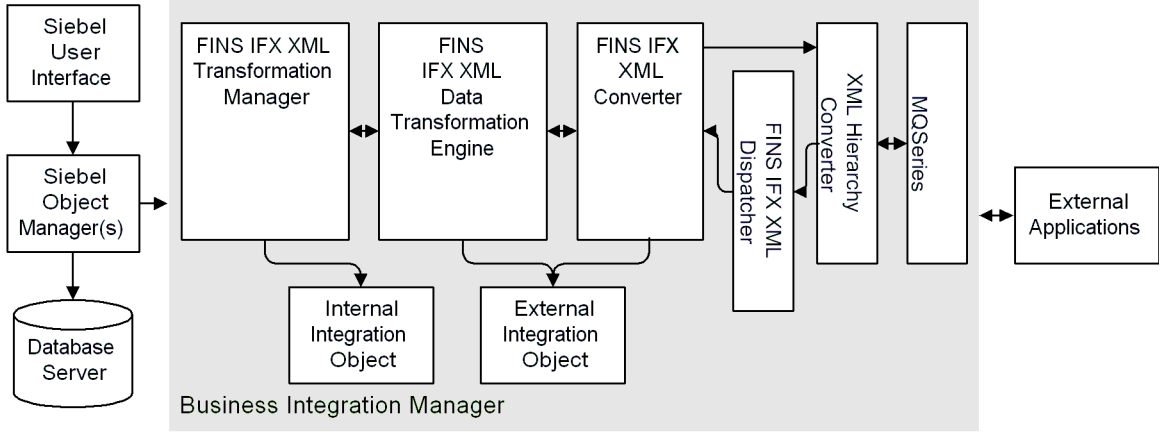


Figure 3. Siebel Connector for IFX XML Components

The Connector components are Siebel business services, which are configured in the Workflow view. The integration objects are created using the FINS IFX XML wizard, and they are configured using the Data Map editor.

NOTE: For information about Siebel integration objects, converter elements, and XML, see *XML Integration Reference*.

IFX XML Syntax and Rules

IFX is a financial-industry version of XML. It contains messages that are appropriate for the financial and banking industries.

The IFX standard defines the required structure and format of an XML message for use with a Siebel connector. The definition is in the IFX DTD, and the IFX DTD is incorporated by the Siebel Connector to construct messages.

This section provides a summary of the IFX XML syntax and rules, and provides the appropriate vocabulary for discussing IFX XML messages. This section supplies knowledge that is basic for any troubleshooting you may need to do.

IFX XML Documents

Each IFX XML document has three distinct parts:

- Envelope
- Header
- Body

The parts are presented as a hierarchy. The envelope is the root, which contains the header and the body. Elements of an IFX XML document that contain other elements are called aggregates.

The envelope and header provide information required by the XML converter and by other components in the connector. The services identify the kind of business service affected by the information, and the messages provide the data that is being exchanged. There are elements that precede the message proper, which specify the versions of XML and IFX.

Sample IFX XML Document

```
<?xml version="1.0" encoding="UTF-8" ?>
<?ifx version="1.0.1" oldfileuid="00000000-0000-0000-0000-
000000000000" newfileuid="11111111-1111-1111-1111-111111111111"
   ?>
<IFX>
   <SignonRq>
      <SessKey>ABCDEFGHIJKLMNOPQRSTUVWXYZYXWVUTSRQPONML</
```

```
SessKey>
      <ClientDt>2001-10-10T17:04:33.0-07:00</ClientDt>
      <CustLangPref>ENU</CustLangPref>
      <ClientApp>
         <Org>Customer Organization</Org>
         <Name>Siebel FINS</Name>
         <Version>7.0</Version>
      </ClientApp>
   </SignonRq>
   <PaySvcRq>
      <SPName>IFX Service Provider</SPName>
      <RqUID>7796AAAA-685E-47b0-9C2F-27FB475B05FA</RqUID>
      <PmtAddRq>
         <RqUID>83DA5F9C-7781-4ebb-BB62-311B8B9C6AD7</RqUID>
         <PmtInfo>
            <RemitInfo>
               <CustPayeeId>SibelCustomerBank</CustPayeeId>
               <CurAmt>
                  <Amt>500</Amt>
                  <CurCode>US</CurCode>
               </CurAmt>
            </RemitInfo>
            <DepAcctIdFrom>
               <AcctId>2547-86392</AcctId>
               <AcctType>CDA</AcctType>
               <BankInfo>
                  <Name>SiebelCustomerBank</Name>
               </BankInfo>
            </DepAcctIdFrom>
            <DueDt>2001-12-24</DueDt>
         </PmtInfo>
         <DupChkOverride>1</DupChkOverride>
      </PmtAddRq>
   </PaySvcRq>
</IFX>
```

Envelope

The envelope is the root element of an XML document. For an IFX XML document, it begins with <IFX> and ends with </IFX>.

The indicator <IFX> is the only item in the envelope.

Header

Every message header has a sign-on element that authenticates the message, and it may have a sign-off element that ends a particular session.

The header has four terms:

- SignonRq
- SignonRs
- SignoffRq
- SignoffRs

The header for a request has the header element < SignonRq > . The header for the response has the header element < SignonRs > . Similarly, the sign-off elements are specifically for requests and responses.

NOTE: IFX XML messages must be either requests or responses. Requests and responses cannot be mixed in a single message. A request uses < SignonRq > . A response uses < SignonRs > .

<Signon> Information

The < SignonRq > or < SignonRs > header element provides a location for status information, authentication information, date and time stamps, language preferences, and identification of the application that will use the data. You can find complete information in the IFX specification.

Authentication Information

The *initial* < SignonRq > for any session must provide authentication information, typically the user name and password or a certificate ID. When the server authenticates the user, using the information in the header, the server issues a session key in the < SignonRs > . Subsequent messages use the session key as a token. After a session has finished, any subsequent session must start with the authentication information again.

The following is an example of an initial SignonRs authentication element.

```
<SignonRq>
   <SignonPswd>
      <CustId>
         <SPName>com.siebel</SPName>
         <CustLoginId>RLIU</CustLoginId>
      </CustId>
      <CustPswd>
         <CryptType>NONE</CryptType>
         <Pswd>DROWSSAP</Pswd>
      </CustPswd>
      <GenSessKey>1</GenSessKey>
   </SignonPswd>
   <ClientDt>2001-11-16T16:56:39.0-08:00</ClientDt>
   <CustLangPref>ENU</CustLangPref>
   <ClientApp>
      <Org>Siebel FINS</Org>
      <Name>Siebel FINS Application</Name>
      <Version>7.0</Version>
   </ClientApp>
</SignonRq>
```

Additional elements may be included in a <Signon> element. You can find complete information in the IFX specification.

Status information, which includes error codes, may also appear in the <Signon> element. Status information is discussed in "Status Information and Error Codes" on page 28.

<Signoff> Information

The <Signoff> header element is used to end a session. A typical time to end a session is at the close of business for the day.

The <Signoff> element, <SignoffRq> or <SignoffRs>, appears at the end of the message, just before the end of the envelope </IFX>. The <Signoff> element may optionally contain a <custID> element.

Body

The body of an IFX XML document provides the content of the information request or response. The body serves as an aggregate containing services and messages. Services and messages, in turn, are aggregates that contain smaller elements.

- **Service.** A service identifies the kind of service being requested or delivered, identifying the business function that will be affected. For example, <PaySvcRq> is a request for a payment service, and <BankSvcRq> is a request for a bank service.
- **Message.** A message identifies the business object affected by the message and the operation that is to be performed on the data. For example, <PmtAddRq> is a request to add a payment.
- **Data Element.** A data element identifies the business component or fields affected by an operation defined in the message. For example, <FirstName> is a data element that contains information about a person's first name.

Services

The basic body element is a service, for example <PaySvcRq>, <BaseSvcRq>, or <BankSvcRq>. <BaseSvcRq> is a request for the base service, which all service providers can provide.

An IFX body can include multiple services. A body almost always contains at least one service. A body with no service would provide only authentication.

The same service may be included in a body more than once, but each service must be for a different service provider.

The following is an example of a message with a single payment service request.

```
<PaySvcRq>
   <SPName>Partner IFX Middleware</SPName>
   <RqUID>50DBF4F7-7888-480b-927E-333652FEBF87</RqUID>
   <PmtAddRq>
      <RqUID>BD620AC4-53E7-4UIL-588C-YOR8D6224FE9</RqUID>
      <PmtInfo>
         <RemitInfo>
            <CustPayeeId>0VF-VEBQ</CustPayeeId>
            <CurAmt>
               <Amt>2500</Amt>
               <CurCode>USD</CurCode>
            </CurAmt>
         </RemitInfo>
         <CardAcctIdFrom>
            <AcctId>2574-86392</AcctId>
            <AcctType>Savings/MMA</AcctType>
         </CardAcctIdFrom>
         <DueDt>2001-11-13</DueDt>
      </PmtInfo>
      <DupChkOverride>1</DupChkOverride>
   </PmtAddRq>
</PaySvcRq>
```

The service aggregate includes a universally unique identifier (UUID) to match responses to requests. The UUID is generated using an algorithm that makes it unique. It appears in the < RqUID > element. It is generated by the client (which sends out the request). It is stored at the client site, which then matches it to the UUID in the response message. The UUID generator can be a Siebel business service or an extension provided by a third party. In any case, the UUID generator is identified by a parameter to the IFX Converter.

Messages

Messages (sometimes called business messages) are contained in Service aggregates. Each service can contain one or more business messages. Each service can contain any number of messages.

The message tag identifies the business object that is affected by the message and a command operator. A business object can be a payment or a cash drawer or anything on which an operation can be performed.

A message uses one of the following operations:

- Add
- Delete
- Cancel
- Inquiry
- Modify
- Audit
- Synchronize

The business message name tag contains the object and the operation. For example, a business message called <PmtAddRq> identifies "payment" as the business object and "add" as the operation. The details of the added payment are provided within the message.

A complete list of business messages for IFX XML is provided in the IFX XML implementation specification.

Data Elements

Within the business message are additional elements that identify the record that should be affected by the request or response and provide any other specifications, such as <CustName>, <PostAddr>, <FirstName>, and <LastName>.

The additional elements include field labels, field information, and tags that provide program access to the data.

The following sample shows data elements for the add payment request.

```
<PmtAddRq>
   <RqUID>BD620AC4-53E7-4UIL-588C-YOR8D6224FE9</RqUID>
   <PmtInfo>
      <RemitInfo>
         <CustPayeeId>0VF-VEBQ</CustPayeeId>
         <CurAmt>
            <Amt>2500</Amt>
            <CurCode>USD</CurCode>
         </CurAmt>
      </RemitInfo>
      <CardAcctIdFrom>
         <AcctId>2574-86392</AcctId>
         <AcctType>Savings/MMA</AcctType>
      </CardAcctIdFrom>
      <DueDt>2001-11-13</DueDt>
   </PmtInfo>
   <DupChkOverride>1</DupChkOverride>
</PmtAddRq>
```

The information in this request is sent to the external application, which performs the request and returns a response.

Status Information and Error Codes

Status information is information about the current status of a message. It appears only in response documents. It can appear in a response header or in any element of a response body.

The external server inserts status information after processing the document. If the processing is satisfactory, status information may or may not be inserted. If there is a problem in the processing, the status information identifies the problem.

A status code of zero means the status is satisfactory. If any other number appears, it is an error code or warning, and the message is flagged. The error code can be used in troubleshooting.

Status information in the header applies to the entire IFX XML document. Status information in a service applies to that service. Status information in a message applies to that message. The following sample shows status information in a header.

```
<Status>
   <StatusCode>100</StatusCode>
   <Severity>Error</Severity>
   <StatusDesc>General Error</StatusDesc>
</Status>
```

For details of status codes, see the IFX XML specification, which provides a description of all error codes.

FINS IFX XML Wizard

Siebel eBusiness Applications provide wizards to guide you through the process of building integration objects and updating dispatcher maps.

You can use the FINS IFX XML Wizard to build integration objects for the Siebel Connector for IFX XML. The wizard guides you through the process of selecting objects (from the Siebel repository or from an external system) on which you can base your new Siebel integration object. The wizard builds a list of valid components from which you choose the specific components to be included in your Siebel integration object.

You access Siebel wizards within the Siebel Integration Object Builder in Siebel Tools. Use the IFX XML wizard to create an appropriate elements hierarchy that reflects the IFX XML DTD. The wizard:

- Creates a set of integration objects to handle outbound and inbound messages and to handle internal and external integration.
- Updates the dispatcher map, which is later used by the dispatcher.

Integration Objects

Siebel integration objects allow you to represent integration metadata between a Siebel business object and an external XML standard, using the IFX XML DTD. The integration object represents a common structure that the eBusiness Application Integration (eAI) infrastructure can understand.

Because these integration objects adhere to a set of structural conventions, they can be traversed and transformed programmatically, using Siebel eScript objects, methods, and functions, or transformed declaratively using Siebel Data Mapper.

To use the Siebel Connector for IFX XML to integrate data you need to build three different integration objects:

- **IFX XML Envelope Integration Object.** An envelope integration object provides envelope and header information for an IFX XML document.

  User properties in an IFX XML envelope provide flexibility to the connector. For example, when a user sends an initial IFX XML request, the IFX XML document uses a <SignonRq> header that is different from subsequent <SignonRq> headers. Two different integration component user properties, initsignon and sessionsignon, can be used to construct different headers under the same envelope.

- **IFX XML Internal Integration Object.** An internal integration object represents the Siebel business object hierarchy for a particular Siebel business object.

- **IFX XML External Integration Object.** An external integration object represents the IFX XML hierarchy for a particular IFX XML message.

FINS IFX XML Dispatcher Map

The dispatcher map is used by FINS IFX XML Dispatcher. The dispatcher map is another integration object that provides a rule set for handling incoming IFX XML messages. The dispatcher map is created and updated by the FINS IFX XML Wizard during the process of creating an external integration object.

The map contains information that associates message instances with the appropriate internal and external integration objects for incoming and outgoing messages. It associates each incoming or outgoing message with all the Siebel Connector for IFX XML elements that are necessary to translate it into Siebel data.

The map contains DTE map names, the internal integration object name, the external integration object name, and Siebel adapter operations. These elements make up the translation scheme for the message instance. The dispatcher map allows the dispatcher to associate the proper translation scheme with each message instance.

All the mapping information is stored in the user property part of the dispatcher map integration object.

IFX XML Transaction Manager

The FINS IFX XML Transaction Manager is responsible for retrieving data from a Siebel application. It may invoke the Siebel adapter or another business service configured in its user properties. It is an adapter that resides logically between the Siebel object manager and the rest of the connector. It executes operations specified in an XML message instance as Siebel database transactions.

The Transaction Manager translates XML command elements into Siebel eAI Adapter actions. The Transaction Manager either carries out the action or finds another business service to carry out the action.

The Transaction Manager combines return results into a single property set. A property set is an intermediate data store that can be used in subsequent operations within the connector.

For inbound processing, the Transaction Manager accepts an IFX XML property set, which may contain multiple integration object instances for multiple transactions. It pairs each individual transaction request with an integration object instance and invokes methods in the Siebel eAI Adapter.

For outbound processing, the Transaction Manager pairs a transaction request with an integration object instance and sends an IFX XML property set to the DTE.

Transaction Manager User Properties

Table 4 describes the FINS IFX XML Transaction Manager user properties.

Table 4. FINS IFX XML Transaction Manager User Properties

Name	Value	Description
DispatcherMapName	< Integration object name >	The dispatch map name. Transaction Manager will use this map to tag the Body information for other components. This value can be set as a runtime input argument, which will overwrite this value.
< Operation Name >	< Service name > / < Method name/ < Argument List >	< OperationName > can be any literal value you want to use to name the operation. The operation can be invoked from the MethodName method in the ServiceName business service passing Argument arguments. < OperationName > is an alias for the method specified by ServiceName/MethodName. < OperationName > is referenced in dispatcher map entries. For information about configuring < Operation Name >, see the following section.

Configuring the <Operation Name> Property

Use a meaningful name for the operation name, such as IXMLOperation_Query.

The value must follow this format:

- "Service/Method/Argument;Argument;"
- "/Method/Argument;Argument;"
- The service, method, and argument are separated by a slash (/)
- Each argument ends with a semicolon (;)
- The default service name is "EAI Siebel Adapter"
- The default argument name is "SiebelMessage"

Uses of the Siebel operation include the following:

- "EAI Siebel Adapter/Query/PrimaryRowId;!SiebelMessage;SearchSpec;"
- "EAI Siebel Adapter/Query/#XMLHierarchy;"
- "EAI Siebel Adapter/Delete/RollbackOnSame;"

Table 5. Operation Examples

Example	Meaning
!SiebelMessage	The default value is to use SiebelMessage as the type of integration object instance. !SiebelMessage means to not use the default value.
#XMLHierarchy	Replace SiebelMessage with XMLHierarchy.

Transaction Manager Methods and Arguments

The FINS IFX XML Transaction Manager methods and arguments are described in the following tables. Table 6 describes the FINS IFX XML Transaction Manager methods.

Table 6. FINS IFX XML Transaction Manager Methods

Method	Display Name	Function
Execute	Execute Transaction	Can be used for inbound or outbound messages when the integration object instance is provided. When only Row_Id is available, use the Execute Outbound method.
Execute Outbound	Execute Outbound	Use only for executing an outbound message.

Table 7 describes the arguments for the Execute Outbound method.

Table 7. Method Arguments for Execute Outbound

Argument	Value	Description
DispatcherMapName	< Integration object name >	Required input string. The name of the dispatcher map that contains the target IFX message.
IsVBC	true, false	Optional input string. Value is TRUE if the source Business Component is a VBC.
IXMLMapPath	< IFX absolute path >	Stores the key for looking up a dispatcher map entry. Transaction Manager uses it to look up the entry value for the integration object instance. Absolute path of the target IFX message.
PrimaryRowId	< row_id >	The primary row Id of the integration object.
SearchSpec	< Search spec >	The search specification of a query for the operation business service to retrieve an integration object instance from the Siebel database.
SiebelFINSOperationOut	< Operation name >	The operation to be used by the Transaction Manager, which is predefined in the user properties of the Transaction Manager.
VBCFieldMap	< Property Set >	Optional input hierarchy. A hierarchy contains the field values for the target VBC.
XMLHierarchy	< XML property set >	A property set that contains an IFX message instance in Siebel internal integration object format.

Table 8 provides specifications for the Execute Outbound method arguments.

Table 8. Argument Specifications for Execute Outbound Method

Name	Display Name	Data Type	Type	Optional
IXMLMapPath	IXML Map Path	String	Input	No
PrimaryRowId	Primary Row Id	String	Input	No
SiebelFINSOperationOut	Outbound operation	String	Input	No
SearchSpec	Search Spec	String	Input	Yes
XMLHierarchy	XML Property Set	Hierarchy	Output	No

Table 9 describes the arguments for the Execute method.

Table 9. Method Arguments for Execute Method

Argument	Value	Description
OnlyIOI	true, false	For an inbound message, the integration object instance for a request may contain header, body, and envelope portions. When the Transaction Manager takes the proper operation against the Siebel application, the integration object instance for a response is generated as well. If this value is set to TRUE, all information from the request message is dropped. In this case, the converter and DTE do not need to deal with the information overhead. If this value is set to FALSE, request information is carried over.
XMLHierarchy	< XML property set >	A property set that contains an IFX document instance in Siebel internal integration object format.

Table 10 provides specifications for the Execute method arguments.

Table 10. Argument Specifications for Execute Method

Name	Display Name	Data Type	Type	Optional
Only IOI	Produce only an integration object instance	String	Input	No
XMLHierarchy	XML Property Set	Hierarchy	Input or Output	No

FINS IFX XML Data Transformation Engine (DTE)

The FINS IFX XML DTE transforms property sets in a Siebel internal integration hierarchy to an external integration object hierarchy, and vice versa. This function allows the FINS IFX XML Converter to exchange data between two systems with different data models. The transformation map is defined at run time from Siebel Administration views.

For inbound processing, the DTE accepts a property set from the FINS IFX XML Converter and transforms it into a property set to be used by the FINS IFX Transaction Manager. The incoming property set is made up of one or more external integration object instances. If there are multiple instances, the DTE parses them into individual instances and transforms them. The DTE then packages the returned transformed instances as an output property set as internal integration object instances.

For outbound processing, the DTE accepts a property set from the Transaction Manager and transforms it into a property set to be used by the converter. The outgoing property set is made up of one or more internal integration object instances. The DTE then packages the returned transformed instances as an output property set as external integration object instances.

DTE Methods and Arguments

The FINS IFX XML DTE methods and arguments are described in the following tables. Table 11 describes the methods for the FINS IFX XML DTE.

Table 11. FINS IFX XML DTE Methods

Method	Display Name	Function
ToExternal	Transform to External Hierarchy	Transforms a Siebel hierarchy into an external hierarchy.
ToInternal	Transform to Siebel Hierarchy	Transforms an external hierarchy into a Siebel hierarchy.

Table 12 describes the arguments for the ToExternal method.

Table 12. Method Arguments for the ToExternal Method

Argument	Value	Description
XMLHierarchy	< XML property set >	Takes as input the output of the Execute outbound method of the IFX XML Transaction Manager. Output hierarchy that contains the IFX Document in Siebel external integration object format.
< MapArgs >		Runtime input arguments that can be used by DTE maps. See the explanation in the following section.
DTE Argument 1	< Any literal value >	
DTE Argument 2	< Any literal value >	
DTE Argument 3	< Any literal value >	

Table 13 provides specifications for the ToExternal method arguments.

Table 13. Argument Specifications for the ToExternal Method

Name	Display Name	Data Type	Type	Optional
XMLHierarchy	XML Property Set	Hierarchy	Input Output	No
< MapArgs >		String	Input	Yes
DTE Argument *n*				Yes

Table 14 describes the arguments for the ToInternal method.

Table 14. Method Arguments for the ToInternal Method

Argument	Value	Description
XMLHierarchy	< XML property set >	Takes as input the output of the XMLPropetySetToPropertySet method of the IFX Converter. Output hierarchy that contains the IFX Document in Siebel internal integration object format.
< MapArgs >		Runtime input arguments that can be used by DTE maps. See the explanation in the following section.
DTE Argument 1	< Any literal value >	
DTE Argument 2	< Any literal value >	
DTE Argument 3	< Any literal value >	

Table 15 provides specifications for the ToInternal method arguments.

Table 15. Argument Specifications for the ToInternal Method

Name	Display Name	Data Type	Type	Optional
XMLHierarchy	XML Property Set	Hierarchy	Input Output	No
< MapArgs >		String	Input	Yes
DTE Argument *n*				

Using <MapArgs>

< MapArgs > is a runtime input argument used by the DTE map to match an integration map argument of an integration object map. The FINS IFX XML DTE can take as many < MapArgs > as needed as long as each name is unique among all the < MapArgs > that are passed to the FINS IFX XML DTE at the same time.

For example, suppose that the output integration object instance has some fields mapping to a workflow process property, such as an ID field.

1. Using the Data Map view, select the integration map to edit in the Integration Object Map applet.
2. In the Integration Map Argument applet, create the map and set the following values:
   - Name = CompId
   - Data Type = “DTYPE_TEXT”
   - Display Name = Component ID
3. In the Integration Field Map applet, set the following values:
   - Target Field Name = [Id]
   - Source Expression = [&CompId]
4. In the workflow, set the data transformation engine input argument as follows:
   - Input Argument = CompId
   - Type = Process Property
   - Property Name = Object Id

At runtime, the DTE replaces [&CompId] with the value of the Object ID.

For some mappings, if the DTE cannot find the source field value, the DTE creates empty tags by default. To remove the empty tags, add IgnoreEmptyTag as the map argument.

For complete information, see *Business Processes and Rules: Siebel eBusiness Application Integration Volume IV*.

FINS IFX XML Converter

The purpose of the FINS IFX XML Converter is to generate and process IFX XML-specific elements, such as the <SignonRq> aggregate and the <SignonRs> aggregate.

The FINS IFX XML Converter receives hierarchy output and converts it into a property set or an XML string.

Converter User Properties

Table 16 describes the FINS IFX XML Converter user properties.

Table 16. Converter User Properties

Name	Value	Description
PI_Parameter: <PI_Name>	<PI_Value>	PI_Parameter: is a constant prefix. PI_Name is provided by the configurator. "PI_Name = PI_Value" would be a PI name-value pair included in IFX PI. Zero or more pairs can be defined. Examples: PI_Parameter:newfileuid PI_Parameter:oldfileuid PI_Parameter:version
PI_Type	IFX	Process Instruction Type.
XMLEnvIntObjectName	<Integration object name>	Integration object name that defines the IFX envelope.
ExceptionForIFXErr	true, false	If true, IFX converter generates an exception when any <StatusCode> in the incoming response message contains a non-zero value. This behavior can be overridden by providing a ProcessStatus outcall function. If a ProcessStatus outcall is provided, this flag does not take effect whether its value is true or false.

Converter Methods and Arguments

The FINS IFX XML Converter methods and arguments are described in the following tables. Table 17 describes the FINS IFX XML Converter methods.

Table 17. FINS IFX XML Converter Methods

Method	Display Name	Function
PropSetToXML	PropSetToXML	Generate the XML message to be sent.
PropSetToXMLPropSet	PropSetToXMLPropSet	Prepare the DOM structure of the XML message to be sent.
XMLPropSetToPropSet	XMLPropSetToPropSet	Convert the XML message received into hierarchical property sets.
XMLToPropSet	XMLToPropSet	Prepare the hierarchical property sets from DOM structure of the XML message received.
Generate Envelope Only	Generate Envelope Only	Generate an IFX document that contains only an IFX envelope and header. No body portion.

Table 18 describes the arguments common to FINS IFX XML Converter methods.

Table 18. Method Arguments for the FINS IFX XML Converter

Argument	Values	Description
initsignon	< Integration component user property name >	Integration component user property of the envelope integration object. The initial signon header type name. Determines which user property in the envelope integration object will be used to construct the initial < SignonRq > header.
sessionsignon	< Integration component user property name >	Integration component user property of envelope integration object. The subsequent signon header type name. Determines which user property in the envelope integration object will be used to construct subsequent < SignonRq > headers.

Table 18. Method Arguments for the FINS IFX XML Converter

Argument	Values	Description
Client Application Name	< Client application name >	Required input string. Value of < Name > of < ClientApp > aggregate.
Client Application Organization	< Client organization name >	Required input string. Value of < Org > of < ClientApp > aggregate.
Client Application Version	< Client application version >	Required input string. Value of < Version > of < ClientApp > aggregate.
Date Input Format	< Date format >	Optional input string. Default value: YYYY-MM-DD
Date Output Format	< Date format >	Optional input string. Default value: YYYY-MM-DD
DateTime Input Format	< Date, time format >	Optional input string. Default value: YYYY-MM-DDTHH:mm:ss.0Z
DateTime Output Format	< Date, time format >	Optional input string. Default value: YYYY-MM-DDTHH:mm:ss.0Z
Time Input Format	< Time format >	Optional input string. Default value: HH:mm:ss.0Z
Time Output Format	< Time format >	Optional input string. Default value: HH:mm:ss.0Z
YrMon Input Format	< Year, month format >	Optional input string. Default value: YYYY-MM
YrMon Output Format	< Year, month format >	Optional input string. Default value: YYYY-MM
Enable Data Formatting	true, false	Optional input string. Default value: false.

Table 18. Method Arguments for the FINS IFX XML Converter

Argument	Values	Description
Client Application Name	< Client application name >	Required input string. Value of < Name > of < ClientApp > aggregate.
Client Application Organization	< Client organization name >	Required input string. Value of < Org > of < ClientApp > aggregate.
Client Application Version	< Client application version >	Required input string. Value of < Version > of < ClientApp > aggregate.
Date Input Format	< Date format >	Optional input string. Default value: YYYY-MM-DD
Date Output Format	< Date format >	Optional input string. Default value: YYYY-MM-DD
DateTime Input Format	< Date, time format >	Optional input string. Default value: YYYY-MM-DDTHH:mm:ss.0Z
DateTime Output Format	< Date, time format >	Optional input string. Default value: YYYY-MM-DDTHH:mm:ss.0Z
Time Input Format	< Time format >	Optional input string. Default value: HH:mm:ss.0Z
Time Output Format	< Time format >	Optional input string. Default value: HH:mm:ss.0Z
YrMon Input Format	< Year, month format >	Optional input string. Default value: YYYY-MM
YrMon Output Format	< Year, month format >	Optional input string. Default value: YYYY-MM
Enable Data Formatting	true, false	Optional input string. Default value: false.

Table 18. Method Arguments for the FINS IFX XML Converter

Argument	Values	Description
IFX Application Business Service Name	< Business service name >	Optional input string. Business service name that contains out-called functionality. If no value is entered, the connector uses default functionality. See the next section for further information about outcalls.
Is Log Out	true, false	Optional input string. Set to “true” for signing off IFX session.
Is Client	true, false	Optional input string. Set to “true” if connector is used at client side. Default value: “True”.
Service Provider Name	< IFX service provider name >	Optional input string. Value of < SPName > of < xxxSvcRq > aggregate.
XMLEnvIntObjectName	< Integration object name >	Required input string. Name of the integration object that defines IFX envelope.
XML Property Set	< XML hierarchy >	Required input hierarchy.

Table 19 provides specifications for the PropSetToXML method arguments.

Table 19. Argument Specifications for the PropSetToXML Method

Name	Display Name	Data Type	Type	Optional
initsignon	initsignon	String	Input	Yes
sessionsignon	sessionsignon	String	Input	Yes
Client Application Name	Client Application Name	String	Input	No
Client Application Organization	Client Application Organization	String	Input	No

Table 19. Argument Specifications for the PropSetToXML Method

Name	Display Name	Data Type	Type	Optional
Client Application Version	Client Application Version	String	Input	No
Date Output Format	Date Output Format	String	Input	Yes
DateTime Output Format	DateTime Output Format	String	Input	Yes
Time Output Format	Time Output Format	String	Input	Yes
YrMon Output Format	YrMon Output Format	String	Input	Yes
Enable Data Formatting	Enable Data Formatting	String	Input	Yes
IFX Application Business Service Name	IFX Application Business Service Name	String	Input	Yes
Is Log Out	Is Log Out	String	Input	Yes
Is Client	Is Client	String	Input	Yes
Service Provider Name	Service Provider Name	String	Input	Yes
XMLEnvIntObjectName	XMLEnvIntObjectName	String	Input	Yes
XML Property Set	XML Property Set	String	Input	Yes
XML Property Set	XML Property Set	String	Output	No

Table 20 provides specifications for the PropSetToXMLPropSet method arguments.

Table 20. Argument Specifications for the PropSetToXMLPropSet Method

Name	Display Name	Data Type	Type	Optional
initsignon	initsignon	String	Input	Yes
sessionsignon	sessionsignon	String	Input	Yes
Client Application Name	Client Application Name	String	Input	No
Client Application Organization	Client Application Organization	String	Input	No
Client Application Version	Client Application Version	String	Input	No

Table 20. Argument Specifications for the PropSetToXMLPropSet Method

Name	Display Name	Data Type	Type	Optional
Date Output Format	Date Output Format	String	Input	Yes
DateTime Output Format	DateTime Output Format	String	Input	Yes
Time Output Format	Time Output Format	String	Input	Yes
YrMon Output Format	YrMon Output Format	String	Input	Yes
Enable Data Formatting	Enable Data Formatting	String	Input	Yes
IFX Application Business Service Name	IFX Application Business Service Name	String	Input	Yes
Is Log Out	Is Log Out	String	Input	Yes
Is Client	Is Client	String	Input	Yes
Service Provider Name	Service Provider Name	String	Input	Yes
XMLEnvIntObjectName	XMLEnvIntObjectName	String	Input	Yes
XML Property Set	XML Property Set	String	Input	Yes
XML Property Set	XML Property Set	String	Output	No

Table 21 provides specifications for the XMLPropSetToPropSet method arguments.

Table 21. Argument Specifications for the XMLPropSetToPropSet Method

Name	Display Name	Data Type	Type	Optional
initsignon	initsignon	String	Input	Yes
sessionsignon	sessionsignon	String	Input	Yes
Date Input Format	Date Input Format	String	Input	Yes
DateTime Input Format	DateTime Input Format	String	Input	Yes
Time Input Format	Time Input Format	String	Input	Yes
YrMon Input Format	YrMon Input Format	String	Input	Yes
Enable Data Formatting	Enable Data Formatting	String	Input	Yes

Table 21. Argument Specifications for the XMLPropSetToPropSet Method

Name	Display Name	Data Type	Type	Optional
IFX Application Business Service Name	IFX Application Business Service Name	String	Input	Yes
XML Property Set	XML Property Set	String	Input	No
XML Property Set	XML Property Set	String	Output	No

Table 22 provides specifications for the XMLToPropSet method arguments.

Table 22. Argument Specifications for the XMLToPropSet Method

Name	Display Name	Data Type	Type	Optional
initsignon	initsignon	String	Input	Yes
sessionsignon	sessionsignon	String	Input	Yes
Date Input Format	Date Input Format	String	Input	Yes
DateTime Input Format	DateTime Input Format	String	Input	Yes
Time Input Format	Time Input Format	String	Input	Yes
YrMon Input Format	YrMon Input Format	String	Input	Yes
Enable Data Formatting	Enable Data Formatting	String	Input	Yes
IFX Application Business Service Name	IFX Application Business Service Name	String	Input	Yes
XML Property Set	XML Property Set	String	Input	Yes
XML Property Set	XML Property Set	String	Output	No

Table 23 provides specifications for the Generate Envelope Only method arguments.

Table 23. Argument Specifications for the Generate Envelope Only Method

Name	Display Name	Data Type	Type	Optional
initsignon	initsignon	String	Input	Yes
sessionsignon	sessionsignon	String	Input	Yes
Client Application Name	Client Application Name	String	Input	No
Client Application Organization	Client Application Organization	String	Input	No
Client Application Version	Client Application Version	String	Input	No
Date Output Format	Date Output Format	String	Input	Yes
DateTime Output Format	DateTime Output Format	String	Input	Yes
Time Output Format	Time Output Format	String	Input	Yes
YrMon Output Format	YrMon Output Format	String	Input	Yes
Enable Data Formatting	Enable Data Formatting	String	Input	Yes
IFX Application Business Service Name	IFX Application Business Service Name	String	Input	Yes
Is Log Out	Is Log Out	String	Input	Yes
Is Client	Is Client	String	Input	Yes
Service Provider Name	Service Provider Name	String	Input	Yes
XMLEnvIntObjectName	XMLEnvIntObjectName	String	Input	Yes
XML Property Set	XML Property Set	String	Input	Yes
XML Property Set	XML Property Set	String	Output	No

Outcalls

The Siebel Connector for IFX XML provides default behaviors for a variety of methods (shown in Table 24 on page 50). If the default behaviors do not suit your needs, the Siebel application allows outcalls to override the defaults.

For example, the Siebel default for the <SignonRq> method is user name and password. You can provide an outcall function for a certificate instead.

The outcall function is activated by the IFX Application Business Service Name parameter. If you want to use an outcall, you define a business service that encloses the outcall functionality, and then enter the business service name in the IFX Application Business Service Name parameter.

If the connector cannot find the business service identified in the IFX Application Business Service Name parameter, default functionality is used instead.

IFX Connector Outcall Methods

This section lists all methods that the IFX connector can use for outcalls. You can implement them using either eScript or VB. For more details, see *Tools Online Help*.

In Table 24, the input argument values are provided by the Siebel Connector for IFX XML, and the output argument values are returned to the Siebel Connector for IFX XML.

Table 24. Connector Outcall Methods

Method	Input Arguments	Output Arguments	Remarks
GenerateSignonRq	An empty property set	Property set contains <SignonRq> aggregate in property set format	Generate a customized <SignonRq> aggregate instead of having the IFX connector generate a default <SignonRq> aggregate.
GenerateSignonRs	Complete IFX request document in property set format	Property set contains <SignonRs> aggregate	Generate a customized <SignonRs> aggregate instead of having the IFX connector generate a default <SignonRs> aggregate.
GenerateSignoffRq	An empty property set	Property set contains <SignoffRq> aggregate in property set format	Generate a customized <SignoffRq> aggregate instead of having the IFX connector generate a default <SignoffRq> aggregate.

Table 24. Connector Outcall Methods

Method	Input Arguments	Output Arguments	Remarks
GenerateSignoffRs	Complete IFX request document in property set format	Property set contains < SignoffRs > aggregate	Generate a customized < SignoffRs > aggregate instead of having the IFX connector generate a default < SignoffRs > aggregate.
GenerateUUID	A property set with Type = GUID	A property set with generated GUID stored as its < Value >	Generate a valid GUID to be < RqUID > .
ProcessSignonRq	Complete IFX request document in property set format	Not required[1]	Process < SignonRq > .
ProcessSignonRs	Complete IFX request document in property set format	Not required	Process < SignonRs > .
ProcessSignoffRq	Complete IFX request document in property set format	Not required	Process < SignoffRq > .
ProcessSignoffRs	Complete IFX response document in property set format	Not required	Process < SignoffRs > .
ProcessStatus	An IFX response message	Not required	Process < Status > . This method is called once for every response message.

Table 24. Connector Outcall Methods

Method	Input Arguments	Output Arguments	Remarks
FormatFieldFromXML	A property set with ■ Type: IFX data type for this element ■ Value: Element value in IFX data format	A property set with ■ Value: Element value in Siebel data format	Convert element value from IFX data format to Siebel data format. This method is called for every element when data formatting is enabled.
FormatFieldToXML	A property set with ■ Type: IFX data type for this element ■ Value: Element value in Siebel data format	A property set with ■ Value: Element value in IFX data format	Convert element value from Siebel data format to IFX data format. This method is called for every element when data formatting is enabled.

1. Process methods do not require output values because the connector is not expecting any value in return. For example, if StatusCode is not 0, you may want to process the information in your own application, but no value is expected by the FINS IFX XML Converter.

FINS IFX XML Dispatcher

The FINS IFX XML dispatcher handles inbound XML hierarchy instances. It provides the necessary information for subsequent modules to perform their operations, such as the integration objects to be used.

The dispatcher identifies incoming messages and parses them into header and envelope sections. It also analyzes incoming message body sections, walking through each command. Using the dispatcher map, the dispatcher associates the message with the appropriate external integration object so that the FINS IFX XML Converter can use it. It also associates the message with the DTE map so that the FINS IFX XML DTE can use it.

Dispatcher User Properties

Table 25 shows the user properties for the dispatcher.

Table 25. User Properties for the FINS IFX XML Dispatcher

Name	Value	Comments
DispatcherMapName	< Integration object name >	Name of an integration object that details the dispatching rules and syntax for the IFX XML standard. This map is usually created along with all the other integration objects needed by the wizard.
XMLEnvIntObjectName	< Integration object name >	Name of an integration object that defines the content and hierarchy for the envelope and header sections of IFX XML.

Dispatcher Methods and Arguments

The FINS IFX XML Converter methods and arguments are described in the following tables. Table 26 describes the FINS IFX XML Dispatcher method.

Table 26. Dispatcher Method

Method	Display Name	Function
Dispatch Message	Dispatch Message	Validates the incoming XML message. if the message conforms to the dispatching rules, the integration object names and other necessary information will be attached to the message.

Table 27 describes the arguments for the Dispatch Message method.

Table 27. Method Arguments for the Dispatch Message Method

Argument	Value	Description
DispatcherMapName	< Integration object name >	Required input string
XMLEnvIntObjectName	< Integration object name >	Required input string
XML Hierarchy	< XML property set >	Required input hierarchy
XML Hierarchy	< XML property set >	Required output hierarchy

Table 28 provides specifications for the Dispatch Message method arguments.

Table 28. Argument Specifications for the Dispatch Message Method

Name	Display Name	Data Type	Type	Optional
DispatcherMapName	DispatcherMapName	String	Input	No
XMLEnvIntObjectName	XMLEnvIntObjectName	String	Input	
XML Hierarchy	XML Hierarchy	Hierarchy	Input	
XML Hierarchy	XML Hierarchy	Hierarchy	Output	

Transport Adapter

The transport adapter is a Siebel business service that provides the interface between the outside data source and the Siebel Connector. The connector can use any of the following standard transport mechanisms.

- MQSeries
- MQSeries AMI
- HTTP
- MSMQ

For details about the transport adapter, see *Transports and Interfaces: Siebel eBusiness Application Integration Volume III*.

Teller IFX XML Workflow Processes

The Siebel Connector for Teller consists of three layers of workflows. Table 29 shows the layers and example workflows.

Table 29. Workflow Layers

Workflow Layer	Example Workflow
Business object-dependent workflow layer	FINS Teller Session Connector FINS Teller Branch Connector FINS Teller Contact Connector
Outbound Teller Connector layer	FINS IFX Connector Outbound
Low-Level Teller Connector layer	FINS IFX Connector Outgoing FINS IFX Connector Incoming FINS IFX Connector Transport Manager

Each layer calls the workflows of the next layer down as a subprocess. Each call to a subprocess passes parameters into the subprocess to control the workflow.

Business Object-Dependent Workflow Layer

The workflows in this layer are the first point of entry for sending a message. They are usually initiated from either a Runtime Event or from a Command Object. See "Initiating Messages" on page 116 for more information on Runtime Events and Command Objects. Modify this workflow layer to add a new message to an existing screen.

These workflows are called business object-dependent because they specify the business object that must be active at the time the workflow is initiated. If the active business object in the UI does not match the business object of the workflow that is being called, the workflow will produce an error. When configuring your workflows, you will need at least one workflow in this layer per business object in the UI that you want to generate messages for.

If you want to add a message to a business object that already has a workflow process defined at this layer, add support for that message into the workflow that is already defined instead of creating a new workflow. For example, if you wanted to add another message to the FINS Teller Session business object, you would edit the existing FINS Teller Session Connector workflow to add the logic for the message rather than creating a new workflow process.

Table 30 displays typical properties of a business object-dependent workflow.

Table 30. Typical Business Object-Dependent Workflow Process Properties

Name	Typical Value	Source	Description
Error Code	0	FINS IFX Connector Outbound workflow process	A standard workflow process property that identifies the error code of the last operation in the workflow. In addition, if there is an error in the incoming XML document, this will be the <StatusCode> value of the <Status> aggregate.
Error Location		FINS IFX Connector Outbound workflow process	Identifies the location of the error in the incoming XML – either the Signon part of the message or the Body part of the message.
Error Message	Success	FINS IFX Connector Outbound workflow process	The message to be presented to the user if this workflow stops with an error status.
Error Severity	Info	FINS IFX Connector Outbound workflow process	Indicates if the error in the received IFX message has a status of Error, Warn, or Info. Maps to the <Severity> element of the <Status> aggregate. This is extracted from the incoming XML propset by the Parse Status method of the FINS Teller Converter Extensions business service.
EventName	Post Transaction	The calling runtime event or command object sets this process property.	The only input parameter to this workflow. A unique string that allows the workflow to branch to the proper step to set the parameters for calls to lower levels in the connector.

Table 30. Typical Business Object-Dependent Workflow Process Properties

Name	Typical Value	Source	Description
Exception Status	< Hierarchy >	FINS IFX Connector Outbound workflow process	A hierarchy that contains all the < Status > aggregates from the incoming message. This process property is used at this level to pass data in to the InsertOverrideRecord method of the FINS IFX XML Extension business service. This is used to insert the error code and error message into the VBC that a supervisor enters his or her override user name and password into.

Table 30. Typical Business Object-Dependent Workflow Process Properties

Name	Typical Value	Source	Description
IFXSearchSpec IFXSearchSpec%1 IFXSearchSpec%2	[FINS Teller Session Transaction.Id] = %1 1-ABC 2-DEF	This workflow process.	These process properties define the search specification string that is used by the FINS IFX XML Transaction Manager business service in the FINS IFX Connector Outgoing workflow. This is the search specification that is passed to the Siebel Adapter to extract the proper business component records. Note that a search specification is not needed if the integration object you are querying with the Siebel adapter only has one component to it: the primary business component of the business object. The reason is that in all workflows, the Object Id process property stores the row Id of the active row in the primary business component, and that is passed in to the FINS IFX XML Transaction Manager. For each %n value in the IFXSearchSpec process property, you need to define a corresponding IFXSearcSpec%1 process property to store the value that will be mapped to the search specification. For example, if the IFXSearchSpec%1 process property evaluates to the value of '1-ABC', and the IFXSearchSpec process property has the value of [FINS Teller Session Transaction.Id] = %1, the FINS IFX XML Transaction Manager will apply the following search specification to its query:[FINS Teller Session Transaction.Id] = '1-ABC'. You can define multiple %n replacement strings in the IFXSearchSpec process property, as long as they have matching IFXSearchSpec%n process properties that contain the value to be replaced. The IFXSearchSpec%1 value maps to the Child Object Id process property in the FINS IFX Connector Outbound workflow. If needed, the IFXSearchSpec%2 value should map to the Grandchild Object Id process property in the FINS IFX Connector Outbound workflow.

Table 30. Typical Business Object-Dependent Workflow Process Properties

Name	Typical Value	Source	Description
Override BusObj Override ViewName	FINS Teller Session FINS Teller Session View	This workflow process	These process properties identify the business object and view to be returned to after an administrator enters that administrator's user name and password to override a transaction. For example, if an override required code is received from the middleware, the FINS Teller Session Connector will navigate to the FINS Teller Override View. After the Override button is pressed on that view, the system will navigate back to the view and business object specified by these process properties. If the message is not going to use an override, you can ignore these process properties.
Object Id	1-ABC	Workflow Engine	This is the active row Id of the primary business component in the business object.
Override Approval Comments		FINS Teller Override Connector workflow process	These are the comments entered on the FINS Teller Override view at the time the supervisor overrides a transaction. If you do not plan to support Approval comments for override in your application, you can ignore this process property.
Override UserName Override Password	SADMIN Db2	FINS Teller Override Connector workflow process	The user name and password entered by the supervisor on the override screen. These values are mapped back in to the message and it is re-sent to the middleware to override a transaction that violates a limit.
OverrideId		FINS Teller Override Connector workflow process	The row Id of the supervisor who performed the override. This is the row Id from the S_PARTY table within Siebel.
Process Instance Id		Workflow Engine	A standard workflow process property that identifies the currently running instance of this process. See *Siebel Business Process Designer Administration Guide* for more information.

Table 30. Typical Business Object-Dependent Workflow Process Properties

Name	Typical Value	Source	Description
Siebel Operation Object Id		Workflow engine	A standard workflow process property that identifies the row Id of the last row affected by a Siebel Operation step.
Teller IFX Application Business Service Name	FINS Teller Converter Extensions	This workflow process.	This identifies the business service used by the connector to perform certain standard functions like generated unique Ids. This is specified explicitly here to allow you to write your own business service to replace the standard functionality provided by the FINS Teller Converter Extensions business service. If you choose to implement your own replacement business service, you need to make sure all the functionality implemented by the FINS Teller Converter Extensions is implemented in it.
Teller IFX Client Application Name	Siebel	This workflow process.	This value is mapped to the < ClientApp > . < Name > element in the outgoing < SignonRq > message by the FINS IFX XML Converter business service.
Teller IFX Client Application Organization	Siebel	This workflow process.	This value is mapped to the < ClientApp > . < Org > element in the outgoing < SignonRq > message by the FINS IFX XML Converter business service.
Teller IFX Client Application Version	1.0	This workflow process.	This value is mapped to the < ClientApp > . < Version > element in the outgoing < SignonRq > message by the FINS IFX XML Converter business service.
Teller IFX Dispatcher Map Integration Object Name	FINSTlrDrwrDispMap	This workflow process.	This identifies the integration object to be used as the dispatcher map for this message.
Teller IFX Doctype	< !DOCTYPE IFX SYSTEM "http://www.siebel.com/teller.dtd" >	This workflow process.	This identifies the doctype string to be used for this message.

Table 30. Typical Business Object-Dependent Workflow Process Properties

Name	Typical Value	Source	Description
Teller IFX Envelope Integration Object	FINS IFX Envelope v101	This workflow process.	This identifies the integration object to be used for the envelope section of the outgoing message. This is used by the FINS IFX XML Converter business service.
Teller IFX Message Full Name	IFX/ tllr.TlrSvcRs/ tllr.EndOfDayRs	This workflow process.	This identifies the user property key in the dispatcher map integration object. This is used to retrieve the value field that is needed by the connector.
Teller IFX Service Provider	Legacy System	This workflow process.	This value is mapped to the <SPName> element that is present in the <CustId> and other aggregates by the DTE engine. This value is passed into the DTE map and converter business service.
Teller IFX Signoff	FALSE	This workflow process.	Set to true if you want the FINS IFX XML Converter business service to generate a <SignoffRq> section at the end of the XML message.
Teller Operation Name	SAQuery	This workflow process.	This value tells the FINS IFX XML Transaction Manager which service to call. This value will match a user property in the business service definition. This user property contains a field that identifies which business service to call to extract data from Siebel. By default, SAQuery will call the eAI Siebel Adapter. However, it can be configured to call any business service you want to generate an internal integration object.
Transport Connect Information	http:// middleware_ip_ address/servlet / com.ibm.dse.cs.ifx.IfxRequestServlet	This workflow process.	This is the address to use to send the generated XML document to. This is passed to the FINS IBC Transport Manager business service.
Type	00 - Deposit	This workflow process.	This process property is specific to the FINS Teller Session Connector workflow. It contains the type of the transaction that is requested to be posted. This is set using an expression within a wait step. The expression queries the current value of the deposit type from the business component.

Outbound Teller Connector Workflow Layer

There is only one workflow in this layer, called FINS IFX Connector Outbound. It is called by all of the business object-dependent workflows to initiate the sending of a message. In general, you do not need to update or create a workflow in this layer.

This workflow binds the request, transport, and response cycles together and implements the IFX session key signon protocol. The IFX session key signon protocol defines how the Siebel Connector for Teller authenticates to the external host.

Within each Siebel Object Manager session, the Siebel Connector for Teller maintains an IFX session key. The first time a Siebel user logs on, this session key is undefined. The first time a user attempts to send a Teller message, this workflow checks the IFX session key. If it is undefined, a signon request message is sent to the middleware to authenticate the user with the Siebel user name and password. Within the signon request message, the < GenSessKey > flag is set to 1 to request that the middleware respond with a session key. If this authentication is successful, the middleware is expected to respond with a session key. This session key is stored internally in the Siebel object manager session and is used for all subsequent messages to the middleware. Using a session key in this way saves time because the middleware does not have to re-authenticate the user with every message.

Conceptually, this is the algorithm implemented by the Outbound Teller Connector:

- Check IFX session key.
- If it is undefined, then
  - Send a signon message with the Siebel username and password. Set the GenSessKey element in the SignonPswd IFX element to request a new session key.
  - Receive the signon response from the middleware.
  - If the signon response was successful, the Converter business service will store the session key in the signon response in the object manager memory for future messages.
- Check the IFX session key again. If the original signon request and response was successful, it should be set.
- Send the message using the IFX session key.

This workflow handles the session key protocol. Most of the work is handled by calling subprocesses. The process properties in this workflow are usually passed into or received from subprocess calls.

Table 31 displays the process properties of the FINS IFX Connector outbound process.

Table 31. FINS IFX Connector Outbound Process Properties

Name	Typical Value	Source	Description
Application SearchSpec Application SearchSpec%1 Application SearchSpec%2 Application SearchSpec%3		Business Object-Dependent workflow layer	See "Business Object-Dependent Workflow Layer" on page 56 for details.
Application Siebel FINS Operation	SAQuery	Business Object-Dependent workflow layer	This value tells the FINS IFX XML Transaction Manager which service to call. This value will match a user property in the business service definition. This user property contains a field that identifies which business service to call to extract data from Siebel. By default, SAQuery will call the eAI Siebel Adapter. However, it can be configured to call any business service you want to generate an internal integration object instance. See "Business Object-Dependent Workflow Layer" on page 56 for details on this business service.

Table 31. FINS IFX Connector Outbound Process Properties

Name	Typical Value	Source	Description
Child Object Id	1-ABC	Business Object-Dependent workflow layer	This is the row Id of the second level of the integration object hierarchy. The row Id of the primary business component in the root of the integration object hierarchy is stored in the Object Id process property. This process property stores the row Id of the child business component of the primary business component.
Error Code	0	FINS IFX Connector Incoming workflow process	A standard workflow process property that identifies the error code of the last operation in the workflow. In addition, if there is an error in the incoming XML document, this will be the < StatusCode > value of the < Status > aggregate.
Error Location		FINS IFX Connector Incoming workflow process	Identifies the location of the error in the incoming XML – either the Signon part of the message or the Body part of the message.
Error Message	Success	FINS IFX Connector Incoming workflow process	The message to be presented to the user if this workflow stops with an error status.
Error Severity	Info	FINS IFX Connector Incoming workflow process	Indicates if the error in the received IFX message has a status of Error, Warn, or Info. Maps to the < Severity > element of the < Status > aggregate. This is extracted from the incoming XML propset by the Parse Status method of the FINS Teller Converter Extensions business service.

Table 31. FINS IFX Connector Outbound Process Properties

Name	Typical Value	Source	Description
Exception Status	< Hierarchy >	FINS IFX Connector Incoming workflow process	A hierarchy that contains all the < Status > aggregates from the incoming message. The value of this process property is just passed up to the Business Object-dependent workflow layer.
IFX Application Business Service Name	FINS Teller Converter Extensions	Business Object-Dependent workflow layer.	Contains the value passed in from the Business Object-dependent workflow layer from the Teller IFX Application Business Service Name process property.
IFX Client Application Name	Siebel	Business Object-Dependent workflow layer.	Passed on to the FINS IFX Connector Outgoing sub process.
IFX Client Application Organization	Siebel	Business Object-Dependent workflow layer.	Passed on to the FINS IFX Connector Outgoing sub process.
IFX Client Application Version	1.0	Business Object-Dependent workflow layer.	Passed on to the FINS IFX Connector Outgoing sub process.
IFX DOCTYPE	< !DOCTYPE IFX SYSTEM "http://www.siebel.com/teller.dtd" >	Business Object-Dependent workflow layer.	Passed on to the FINS IFX Connector Outgoing sub process.
IFX DispMap Integration Object Name	FINSTlrDrwrDispMap	Business Object-Dependent workflow layer.	Passed on to the FINS IFX Connector Outgoing sub process.
IFX Envelope Integration Object Name	FINS IFX Envelope v101	Business Object-Dependent workflow layer.	Passed on to the FINS IFX Connector Outgoing sub process.
IFX IsSignoff	FALSE	Business Object-Dependent workflow layer.	Passed on to the FINS IFX Connector Outgoing sub process.
IFX Outgoing Message Lookup Key	IFX/tllr.TlrSvcRs/tllr.EndOfDayRs	Business Object-Dependent workflow layer.	Passed on to the FINS IFX Connector Outgoing sub process.

Table 31. FINS IFX Connector Outbound Process Properties

Name	Typical Value	Source	Description
IFX Service Provider Name	Legacy System	Business Object-Dependent workflow layer.	Passed on to the FINS IFX Connector Outgoing sub process.
IFX Session Key	WLAEJFKCREIDJSKLAL	GetSessionID method of the FINS Teller Converter Extensions business service.	The session key assigned to the current session. Used to check if a session key has been received from the middleware.
Object Id	1-ABC	Workflow Engine	The object Id of the primary business component in the active business object.
Override Approval Comments Override UserName Override Password OverrideId	Approved by Bob Smith SADMIN Db2 1-ABC	Business Object-Dependent workflow layer	All these process properties are passed in from the Business Object-dependent workflow layer.
Process Instance Id	1-ABC	Workflow engine	A standard workflow process property that identifies the running instance of this process. See *Siebel Business Process Designer Administration Guide* for more information.
Root Object Id	1-ABC	Business Object-Dependent workflow layer	The row Id of the root of the integration component tree. This value is passed in from the Object Id value in the calling workflow process.
Siebel Operation Object Id	1-ABC	Workflow engine	A standard workflow process property that identifies the row Id of the last row affected by a Siebel Operation step. See *Siebel Business Process Designer Administration Guide* for more information.
Transport Connect Information	http://middleware_ip_address/servlet / com.ibm.dse.cs.ifx.IfxRequestServlet	Business Object-Dependent workflow layer	Contains the value passed in from the Business Object-dependent workflow layer from the Transport Connect Information process property.

Table 31. FINS IFX Connector Outbound Process Properties

Name	Typical Value	Source	Description
Transport Logging Flag			Passed in to the FINS IFX Connector Outgoing subprocesses to enable or disable logging. Setting this flag to Y will cause the outgoing and incoming messages to be logged to the default logging file defined in the FINS IFX Connector Outbound workflow process. The file name is specified in the steps that call the FINS IFX Connector Transport Manager sub processes. By default, this flag will be set to N. Before setting it to Y, examine the default logging directory to make sure it is valid for your system.
VBC Business Component Name	FINS Teller Electronic Journal	Business Object-Dependent workflow layer	This value contains the name of the business component that is the VBC. It is set by the calling workflow process. Ignore this process property when you are not using VBC.
VBC IsVBC	true	Business Object-Dependent workflow layer	If this value is true, it indicates that the business component that the Transaction Manager is querying is a VBC.
VBCFieldMap	< Hierarchy >		This process property contains the field-value map of the values in the VBC. Its value is set by the FINS IFX Connector Outgoing subprocess.
XML Document Request		FINS IFX Connector Outgoing workflow process	The generated XML request that is sent to the middleware. This is passed as input to the FINS IFX Connector Transport Manager workflow.
XML Document Response		FINS IFX Connector Transport Manager workflow process	The response XML request that is received from the middleware. This is passed as input to the FINS IFX Connector Incoming subprocess.

Low-Level Teller Connector Workflow Layer

Table 32 displays the three Siebel Connector for Teller workflows at this level.

Table 32. Low-Level Teller Connector Workflows

Workflow	Description
FINS IFX Teller Connector Outgoing	Generates the outgoing XML string from data in the Siebel business components.
FINS IFX Teller Connector Transport Manager	Sends the XML string using one of the supported eAI transports to the middleware.
FINS IFX Teller Connector Incoming	Receives the incoming XML string and maps it back to Siebel business components.

Table 33 displays the properties for the FINS IFX Connector Outgoing Process.

Table 33. FINS IFX Connector Outgoing Process Properties

Name	Typical Value	Source	Description
Business Component Name	FINS Teller Electronic Journal	FINS IFX Connector Outbound workflow process	Identifies the VBC name that should be the source BC to be queried by the FINS IXML Transport Manager business service. Ignore this process property if you are not using VBC.
Error Code	0	Workflow Engine	A standard workflow process property that identifies the error code of the last operation in the workflow. In addition, if there is an error in the incoming XML document, this will be the < StatusCode > value of the < Status > aggregate.

Table 33. FINS IFX Connector Outgoing Process Properties

Name	Typical Value	Source	Description
Error Message	Success	Workflow Engine	The message to be presented to the user if this workflow stops with an error status.
IFX Application Business Service Name	FINS Teller Converter Extensions	FINS IFX Connector Outbound workflow process	Contains the value passed in from the FINS IFX Connector Outbound workflow process. Identifies the extension business service for the FINS IFX XML Converter to use.
IFX Client Application Name IFX Client Application Organization IFX Client Application Version	Siebel Systems, Inc. Siebel Systems, Inc. 1.0	FINS IFX Connector Outbound workflow process	These properties are all passed in to the FINS IFX XML Converter business service for inclusion in the generated XML string.
IFX DOCTYPE	<!DOCTYPE IFX SYSTEM "http://www.siebel.com/teller.dtd" >	FINS IFX Connector Outbound workflow process	Passed to the XML Hierarchy Converter business service for inclusion in the generated XML string.
IFX DispMap Integration Object Name	FINSTlrDrwrDispMap	FINS IFX Connector Outbound workflow process	Passed to the FINS IFX Transaction Manager business service.
IFX Envelope Name	FINS IFX Envelope v101	FINS IFX Connector Outbound workflow process	Passed to the FINS IFX XML Converter business service.

Table 33. FINS IFX Connector Outgoing Process Properties

Name	Typical Value	Source	Description
IFX Message Full Name	IFX/ tllr.TlrSvcRs/ tllr.EndOfDayRs	FINS IFX Connector Outbound workflow process	If this value is NULL, there is no message body to be included in the workflow and a simple signon message is generated. This is passed to the FINS IFX XML Transport Manager business service.
IFX Service Provider Name	Legacy System	FINS IFX Connector Outbound workflow process	Used by the FINS IFX XML Converter and the FINS IFX XML DTE business services. The Converter uses this to get the value for <SPName> tags in the header. The DTE uses this to get the same value for <SPName> tags in the body of the message.
IFX Signoff	FALSE	FINS IFX Connector Outbound workflow process	Set to true if you want the FINS IFX XML Converter business service to generate a <SignoffRq> section at the end of the XML message.
IFXSearchSpec IFXSearchSpec%1 IFXSearchSpec%2 IFXSearchSpec%3		FINS IFX Connector Outbound workflow process	Contains the search specification that the FINS IFX XML Transaction Manager should use to pass to the Siebel Adapter.
IsVBC	false	FINS IFX Connector Outbound workflow process	If this value is true, it indicates that the business component that the Transaction Manager is querying is a VBC.

Table 33. FINS IFX Connector Outgoing Process Properties

Name	Typical Value	Source	Description
Object Id	1-ABC	Workflow engine	The object Id of the primary business component in the currently active business object.
Override Password Override UserName	Db2 SADMIN	FINS IFX Connector Outbound workflow process	The user name and password to be mapped into the current message by the DTE map.
PrimaryRowId	1-ABC	FINS IFX Connector Outbound workflow process	Used by the FINS IFX XML Transaction Manager to pass to the Siebel Adapter to perform the query.
PropSet Converter Out	< Hierarchy >	FINS IFX XML Converter business service	The output property set from the converter step.
PropSet DTE Out	< Hierarchy >	FINS IFX XML DTE business service	The output property set from the DTE step.
PropSet TransMgr Out	< Hierarchy >	FINS IFX XML Transaction Manager business service	The output property set from the Transaction Manager step.
Siebel FINS Operation	SAQuery	FINS IFX Connector Outbound workflow process	Used by the FINS IFX XML Transaction Manager to identify the service to run. This identifies a user property on that service.
Transport Connect Info	http://middleware_ip_address/servlet/com.ibm.dse.cs.ifx.IfxRequestServlet	FINS IFX Connector Outbound workflow process	The address to send the generated XML document to. This is passed to the FINS IBC Transport Manager business service.

Table 33. FINS IFX Connector Outgoing Process Properties

Name	Typical Value	Source	Description
Transport Logging Flag	N	FINS IFX Connector Outbound workflow process	Enables or disables the logging of the XML request and response files in the FINS IFX Connector Transport Manager workflow subprocess.
VBCFieldMap	Hierarchy	FINS IFX Connector Outbound workflow process	This process property contains the field-value map of the values in the VBC. Its value is set by the FINS IFX Connector Outgoing subprocess.
XMLDocument	< ?xml version = "1.0" encoding = "UTF -8"? > < ?Siebel-Property-Set EscapeNames = " true"? > < IFX > < Signon Rq > …	XML Hierarchy Converter business service	A string representing the XML request message.

Table 34 displays the properties for the FINS IFX Connector Transport Manager process.

Table 34. FINS IFX Connector Transport Manager Process Properties

Name	Typical Value	Source	Description
Connect Info	http://middleware_ip_address/servlet /com.ibm.dse.cs.ifx.IfxRequestServlet	FINS IFX Connector Outbound workflow process	The address to send the generated XML document to. This is passed to the FINS IBC Transport Manager business service.
Error Code	0	Workflow engine	A standard workflow process property that identifies the error code of the last operation in the workflow In addition, if there is an error in the incoming XML document, this will be the < StatusCode > value of the < Status > aggregate.
Error Message	Success	Workflow engine	The message to be presented to the user if this workflow stops with an error status.
Log File Incoming	c:\temp\IFXMessageLogIncoming.xml	FINS IFX Connector Outbound workflow process	Specifies where to log the incoming XML file.
Log File Outgoing	c:\temp\IFXMessageLogIncoming.xml	FINS IFX Connector Outbound workflow process	Specifies where to log the outgoing XML file.

Table 34. FINS IFX Connector Transport Manager Process Properties

Name	Typical Value	Source	Description
XML Document Request	< ?xml version = "1.0" encoding = "UTF -8"? > < ?Siebel-Property-Set EscapeNames = " true"? > < IFX > < Signon Rq > …	FINS IFX Connector Outbound workflow process	A string representing the complete XML request document.
XML Document Response	< ?xml version = "1.0" encoding = "UTF -8"? > < ?Siebel-Property-Set EscapeNames = " true"? > < IFX > < Signon Rs > …	FINS IBC Transport Manager business service	A string representing the complete XML response document.

Table 35 displays the properties for the FINS IFX Connector Incoming process.

Table 35. FINS IFX Connector Incoming Process Properties

Name	Typical Value	Source	Description
Error Code	0		A standard workflow process property that identifies the error code of the last operation in the workflow. In addition, if there is an error in the incoming XML document, this will be the <StatusCode> value of the <Status> aggregate.
Error Code - Siebel	Success	EAI Value Map Translator business service	Contains the result of looking up the error code received in the XML response message and translating it to a text status. The text status is usually 'Success,' but it can be 'Warning' or 'Error' or other strings. The correlation between XML response codes and text statuses is maintained in the eAI Value Maps screen.
Error Location		FINS Teller Converter Extensions business service, Parse Status method	Identifies the location of the error in the incoming XML – either the Signon part of the message or the Body part of the message.
Error Message	Success	Workflow engine or FINS Teller Converter Extensions business service, Parse Status method	The message to be presented to the user if this workflow stops with an error status.

Table 35. FINS IFX Connector Incoming Process Properties

Name	Typical Value	Source	Description
Error Severity	Info	FINS Teller Converter Extensions business service, Parse Status method	Indicates if the error in the received IFX message has a status of Error, Warn, or Info. Maps to the <Severity> element of the <Status> aggregate. This is extracted from the incoming XML propset by the Parse Status method of the FINS Teller Converter Extensions business service.
Exception Status		FINS IFX XML Converter	A hierarchy that contains all the <Status> aggregates from the incoming message. The converter extracts this hierarchy so it can be parsed by the Parse Status business service. The Parse Status service in turn scans this hierarchy looking for non-zero status codes, and sets the Error Severity, Error Location, Error Code, and Error Message process properties accordingly.
Root Object Id	1-ABC	FINS IFX Connector Outbound workflow process	The row Id of the primary business component in the business object. This is mapped into the internal integration object by the DTE step, and is usually used by the Siebel Adapter as the user key.

Table 35. FINS IFX Connector Incoming Process Properties

Name	Typical Value	Source	Description
Child Object Id	1-DEF	FINS IFX Connector Outbound	The row Id of the child business component in the business object. This is mapped into the internal integration object by the DTE step, and is usually used by the Siebel Adapter as the user key.
Grandchild Object Id	1-GHI	FINS IFX Connector Outbound	The row Id of the grandchild business component in the business object. This is mapped into the internal integration object by the DTE step, and is usually used by the Siebel Adapter as the user key.
IFX Application Business Service Name	FINS Teller Converter Extensions	FINS IFX Connector Outbound workflow process	Contains the value passed in from the FINS IFX Connector Outbound workflow process. Identifies the extension business service for the FINS IFX XML Converter to use.
IFX Dispatcher Map Integration Object Name	FINSTlrDrwrDispMap	FINS IFX Connector Outbound workflow process	Passed to the FINS IFX Dispatcher business service.
IFX Enable Default Data Formatting	true	This workflow process	Passed to the FINS IFX XML Converter business service to enable or disable date/time formatting.

Table 35. FINS IFX Connector Incoming Process Properties

Name	Typical Value	Source	Description
IFX Formatting Date Input Format	YYYY-MM-DD	This workflow process	Passed to the FINS IFX XML Converter business service to control the output date/time format for IFX date values.
IFX Formatting DateTime Input Format	YYYY-MM-DDTHH:mm:ss.0Z	This workflow process	Passed to the FINS IFX XML Converter business service to control the output date/time format for IFX date/time values.
IFX Formatting Time Input Format	HH:mm:ss.0Z	This workflow process	Passed to the FINS IFX XML Converter business service to control the output time format for IFX time values.
IFX Formatting YrMon Input Format	YYYY-MM	This workflow process	Passed to the FINS IFX XML Converter business service to control the output time format for IFX year/month values.
IsVBC	false	FINS IFX Connector Outbound workflow process	Set to true to indicate to the FINS IFX XML Transaction Manager that the internal integration object you are processing maps to a virtual business component.
Object Id	1-ABC	Workflow engine	The object Id of the primary business component in the active business object.

Table 35. FINS IFX Connector Incoming Process Properties

Name	Typical Value	Source	Description
Override Approval Comments	Approved by Bob Smith	FINS IFX Connector Outbound workflow process	Used by the DTE to map the approval comments into the internal integration object. This allows the approval comments to be upserted into the Siebel database after a successful override transaction.
OverrideId		FINS IFX Connector Outbound workflow process	Used by the DTE to map the row Id of the user who approved the override into the internal integration object. This allows the identity of the person who did the override to be upserted into the Siebel database.
PropSet Converter Out	< Hierarchy >	FINS IFX XML Converter	The output from the converter step. This performs date/time format conversion and sets the session key if necessary.
PropSet DTE Out	< Hierarchy >	FINS IFX XML DTE	The output from the DTE step. This converts the property set to internal integration object format from external integration object format.
PropSet Dispatcher Out	< Hierarchy >	FINS IFX XML Dispatcher	The output from the dispatcher step. This tags the hierarchy with properties that identify the envelope and body sections.
PropSet TransMgr Out	< Hierarchy >	FINS IFX XML Transaction Manager	The output from the Transaction Manager step.

Table 35. FINS IFX Connector Incoming Process Properties

Name	Typical Value	Source	Description
PropSet XML Doc	< Hierarchy >	XML Hierarchy Converter	The output from the hierarchy converter step. This is the direct hierarchical representation of the incoming XML document.
StatusCode	0	FINS Teller Converter Extensions business service, Parse Status method	The status code retrieved from the input Exception Status hierarchy. This represents the < StatusCode > value from the < Status > aggregate. If there is more than one < Status > aggregate in the message, this is the one with the error. If more than one < Status > aggregate has an error, this is the error from the < SignonRs > part of the message.
XML Document	< ?xml version = "1.0" encoding = "UTF -8"? > < ?Siebel-Property-Set EscapeNames = " true"? > < IFX > < Signon Rs > …	FINS IFX Connector Outbound workflow process	A string representing the complete XML response document.

Low-Level Teller Connector Workflow Process Flow

This section provides a summary of the algorithms used in the FINS IFX Connector Outgoing and FINS IFX Connector Incoming workflow processes to help you understand how the various components fit together.

NOTE: This description is for a general case that is used most of the time. It does not cover the details of situations like using VBCs or overrides.

Table 36 displays the steps of the FINS IFX Connector Outgoing flow.

Table 36. FINS IFX Connector Outgoing Flow Steps

Step	Description
FINS IFX XML Transaction Manager	Input: DispatcherMapName, IXMLMapPath, isVBC (optional), and VBCFieldMap(optional) argument. ■ Looks up the dispatcher map entry for the message. ■ From the dispatcher map entry, identifies the internal integration object to instantiate. ■ Calls the Siebel Adapter to create an internal integration object instance. Passes in the search specification to constrain the rows returned by the Siebel Adapter to just those that are specified. ■ Adds a property to the internal integration object instance that identifies the DTE map name obtained from the dispatcher map. ■ If a virtual business component is used, transforms the VBCFieldMap into an integration object instance.
FINS IFX XML DTE	Input: Internal integration object instance. ■ Using the internal integration object name and DTE map name, looks up the external integration object name from the DTE map. ■ Instantiates a new external integration object. ■ Passes the internal integration object instance, DTE map name, and external integration object instance to the eAI Data Transformation Engine business service. ■ Returns the external integration object instance after the transformation that contains the message body.

Table 36. FINS IFX Connector Outgoing Flow Steps

Step	Description
FINS IFX XML Converter	Input: External integration object instance representing the message body and the Envelope integration object name. ■ Creates a new instance of the envelope integration object and populates it with the proper data values such as the IFX session key, user name, password, and other values passed in. ■ Performs any date/time conversion to convert the internal date formats to the external date formats. ■ Generates and adds the UUIDs to the proper places in the external integration object. ■ Returns the external integration object instance that contains the envelope and message body. Date/time values have been converted, UUIDs have been generated, and the session key or user name/password information has been attached.
XML Hierarchy Converter	Input: External integration object instance representing the complete message and envelope. A string for the DOCTYPE declaration. ■ Serializes the integration object instance to a string value. ■ Adds the DOCTYPE declaration in the appropriate place.

Table 37 displays the steps of the FINS IFX Connector Incoming flow.

Table 37. FINS IFX Connector Incoming Flow

Step	Description
XML Hierarchy Converter	Input: The incoming XML string. ■ Converts the incoming XML string to a generic property set.
FINS IFX XML Dispatcher	Input: The incoming property set, the dispatcher map name, and the envelope integration object name. ■ Creates a property set with various nodes tagged to identify them as header or body nodes. ■ Looks up the dispatcher map to find a key that matches the message name, then decomposes the dispatcher map value into its components and adds those values as properties to the output property set.
FINS IFX XML Converter	Input: The tagged property set from the Dispatcher. ■ Creates an external integration object instance for the incoming property set. The external integration object name is the value extracted from the dispatcher map. ■ Converts the IFX date/time format to internal Siebel format. ■ Creates a new property set that holds all the < Status > aggregates from the incoming message.
Parse Status	Input: The status property set generated from the converter. ■ Scans the status property set and sets the error code, error message, error location, and error severity arguments based on the status information. ■ If the error code is nonzero, it will use the Tools object called Message Category to look up the corresponding translated message for the given error code. The Message Category object in Tools should have an entry called IFX Messages. This will contain a list of error codes and error messages that are displayed to the user. The error message displayed to the user is obtained from this table using the incoming IFX error code.

Table 37. FINS IFX Connector Incoming Flow

Step	Description
Translate Error Code	Input: The error code process property. ■ Calls the EAI Value Map Translator to translate the numeric error code to a word like 'Success' or 'Warning.'
FINS IFX XML DTE	Input: The external integration object from the converter. ■ Calls the FINS IFX XML DTE service to transform the external integration object to an internal integration object. The DTE map name is identified by the value extracted from the dispatcher map. ■ Returns an internal integration object.
FINS IFX XML Transaction Manager	Input: The internal integration object from the DTE. ■ Calls the configured business service from the dispatcher map, usually the Siebel Adapter, to update, insert, or delete data from the Siebel application.

Teller IFX Connector Roadmap 3

This chapter discusses the steps needed to perform common connector configuration tasks.

Integration Objects

Integration objects define the intermediate format of the data so that it can be used by the connector components to translate between Siebel data formats and Teller IFX XML data formats. An integration object is an empty template that defines a set of fields that can be stored within it. There are two main types of integration objects: internal integration objects and external integration objects. Internal integration objects are based on Siebel business objects. External integration objects are based on XML messages. For more information on Siebel integration objects, see *Integration Platform Technologies: Siebel eBusiness Application Integration Volume II*.

Several wizards are available to create integration objects for you. Before using a wizard to create integration objects, you should understand the basics of the wizard, the dispatcher map, the envelope, the external integration objects, and the internal integration objects.

Configuring a Dispatcher Map

The Dispatcher Map is an integration object used internally by a business service FINS IFX XML Dispatcher. It stores information on how to process a message coming from an external system. Unlike regular integration objects, it stores the information on how to process messages as user properties. There are no integration components or fields in a dispatcher map integration object.

You can associate a dispatcher map with an IFX XML wizard by editing the user properties of the Wizard business service. Add a user property to the IFX XML wizard called DispatcherMapName. When configured as the dispatcher map entry in an IFX XML wizard, the dispatcher map integration object definition will be updated every time the wizard is used to create integration objects based on a DTD.

Teller uses the following three dispatcher maps:

- FINSTlrBranchDispMap
- FINSTlrDrwrDispMap
- FINSTlrTranDispMap

Each of these maps is associated with a different IFX XML Wizard. There are three identical IFX XML wizards configured with different user properties to support development on three different projects at once.

Examining an Existing Dispatcher Map

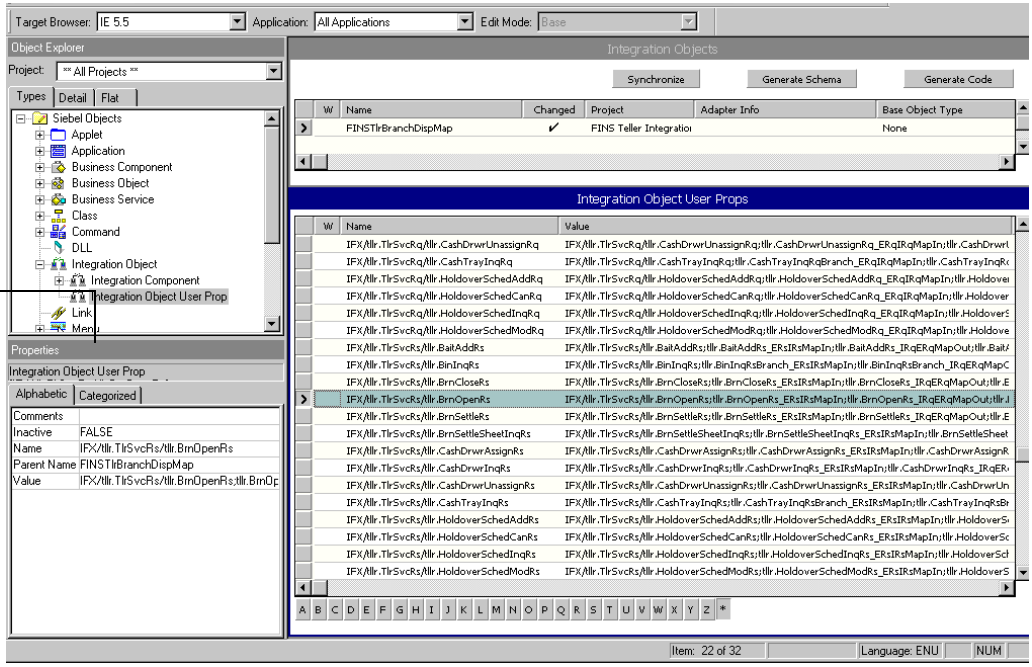
To learn more about the dispatcher map, open and study a standard dispatcher map, such as FINSTlrBranchDispMap. FINSTlrBranchDispMap is the dispatcher map entry of the FINS Teller Integration Branch Wizard business service. This business service is associated with the dispatcher map through the DispatcherMapName user property.

Examine the map structure in Siebel Tools, the FINS Teller Integration Wizard user property where it is referred to in Siebel Tools, and Workflow FINS Teller Branch Connector, where it is used.

To review the dispatcher map and Wizard business service

1. Open Siebel Tools.
2. Choose Integration Object from the Object Explorer window and query for the integration object FINSTlrBranchDispMap.
3. Expand the Integration Object from the Object Explorer Window and click Integration Object User Prop.

   The details of a sample user property value are displayed in the figure below.



4. Choose Business Service from the Object Explorer Window and query for the business service FINS Teller Integration Branch Wizard.
5. Expand the Business Service from the Object Explorer Window and click the user property.

Example

The following is a sample value from the FINSTlrBranchDispMap dispatcher map entry for the IFX/tllr.TlrSvcRs/tllr.BrnOpenRs name.

```
IFX/tllr.TlrSvcRs/
tllr.BrnOpenRs;tllr.BrnOpenRs_ERsIRsMapIn;tllr.BrnOpenRs_IRqERqM
apOut;tllr.BrnOpenRs;FINS Teller Branch Status Chg Int;Upsert
```

Table 38 describes the elements of the sample.

NOTE: Generally, you will not need to manually create values. These are set to default values by the FINS IFX XML Wizards.

Table 38. Dispatcher Map Sample Elements

Element	Description
IFX/tllr.TlrSvcRs/tllr.BrnOpenRs	The value used by the dispatcher business service to tag the tllr.BrnOpenRs node as the node that represents the message body.
tllr.BrnOpenRs_ERsIRsMapIn	The name of the DTE map to transform the external response integration object to the internal response integration object (ERsIRs).
tllr.BrnOpenRs_IRqERqMapOut	The name of the DTE map to transform the internal request integration object to the external request integration object (IRqERq).
tllr.BrnOpenRs	The name of the external integration object. This is not used by outgoing messages. For outgoing messages, the external integration object name is derived by the FINS IFX DTE service by looking at the external integration object associated with the outgoing DTE map name. This is used by incoming messages, as the FINS IFX Converter service instantiates a new integration object of this type.
FINS Teller Branch Status Chg Int	The name of the internal integration object. This is used by outgoing messages, as the FINS IFX Transaction Manager service instantiates a new integration object of this type. This is not used by incoming messages. The internal integration object name is derived by the FINS IFX DTE service by looking at the internal integration object associated with the incoming DTE map name.
Upsert	The action for the FINS IFX Transaction Manager service to perform for incoming messages. This string should match a user property defined on the FINS IFX Transaction Manager service that defines which business service and method the Transaction Manager should invoke.

Configuring a Teller XML Integration Object Wizard

Teller XML Integration Object Wizards are configured as business services using Siebel Tools. Use the wizards to create both internal and external integration objects for both request and response messages, and a dispatcher map entry in a single run. The wizard can create up to four new integration objects, and add an entry to the dispatcher map in a single run. Use the wizards to create both internal and external integration objects, an envelope integration object, and a dispatcher map entry in a single run. This behavior is different from that of eBusiness Application Integration (eAI) Siebel Wizard, which can generate only a single internal integration object at a time. The Teller XML Integration Wizard is capable of generating a set of integration objects at a time.

The Wizards have the capability to create envelope integration objects, but a predefined envelope integration object called FINS IFX Envelope v101 is already supplied and tested. This envelope integration object should be used for all Teller messages.

These generated integration objects are closely related. They will be used together by the connector to finish the integration job, generate the request message, and parse the response message. There are three Teller wizards used to create the internal integration objects and the external integration objects, as well as to update the dispatcher map. They are:

- FINS Teller Integration Branch Wizard
- FINS Teller Integration Drawer Wizard
- FINS Teller Integration Transaction Wizard.

To invoke the wizard

1. In Siebel Tools, choose File > New Object.
2. Click the EAI tab.
3. Click Integration Object.

Examining an Existing Wizard

To learn more about using the Teller XML Integration Object wizard, open and study one in Siebel Tools. A good wizard to study for this purpose is the FINS Teller Integration Branch Wizard. Examine the user properties of the business service.

Table 39 shows the presetup user properties for the FINS Teller Integration Branch Wizard.

Table 39. User Properties for the FINS Teller Integration Branch Wizard

Name	Value	Description
DispatcherMapName	FINSTlrBranchDispMap	The Dispatch Map Name wizard will use this map to update the key and value.
HasUIControl	True	Internal use.
Integration Object Wizard	Y	Internal use.
Integration Object Base Object Type	Siebel Business Object	Internal use.
Operation KeyWord Match:0	Add/IXMLOperation_ADD	Internal use. This means that when the wizard generates an external integration object for a message with the word Add in it, it creates a dispatcher map entry with the action of IXMLOperation_Add. This entry should match a user property on the FINS IFX XML Transaction Manager. This user property identifies a business service and method to run when the response message received is an “Add” message. The operation is defined in the Transaction Manager as “SAUpsert.” The operation name will be recorded in the dispatch map.
Default Envelope Tag	IFX	Value for the Envelope Tag.

You can customize the Teller XML Integration Object Wizards to fit your needs or create new ones under certain conditions.

In the following situations, customize the available wizard:

- When you need to use a Dispatcher Map other than the one provided.
- When you need to use different Siebel operations for the same type of messages.

In the following situations, create a new wizard:

- When you need to work in parallel and you want your integration objects to be created in different Tools projects. (All integration objects are created in the same Tools project that the Wizard business service is in.) To do this, copy the Wizard business service to the new project and rename it.
- When the same message needs to be sent from events that are based on different internal integration objects, which means you need the same message entry in different Dispatcher maps. To do this, you could manually adjust the dispatcher map instead of creating a new Wizard business service.

Once you have customized the existing wizard or created a new wizard, compile the definition into the repository that Siebel Tools is using before the changes take effect.

NOTE: The repository Siebel Tools is using is different from the repository SRF file the application uses.

Configuring the Envelope Integration Object

An envelope integration object provides the envelope and header information needed by the IFX XML hierarchy. It needs to be created only once and will be shared throughout the application. Teller uses the integration object FINS IFX Envelope v101 as the envelope, which is sufficient for general IFX messages.

The Teller XML Integration Object Wizard has the ability to create new envelope integration objects, but you should use this predefined envelope integration object.

Configuring the External Integration Object

An external integration object establishes the hierarchy for an IFX XML message. Each Teller message needs one and only one integration object that defines the message portion of the IFX XML document. The integration object preserves the cardinality and data type information as well as the hierarchy. In general, an IFX XML message maps to an external integration object, elements within that message map to integration components, and attributes of the elements map to integration component fields.

Examining an Existing External Integration Object

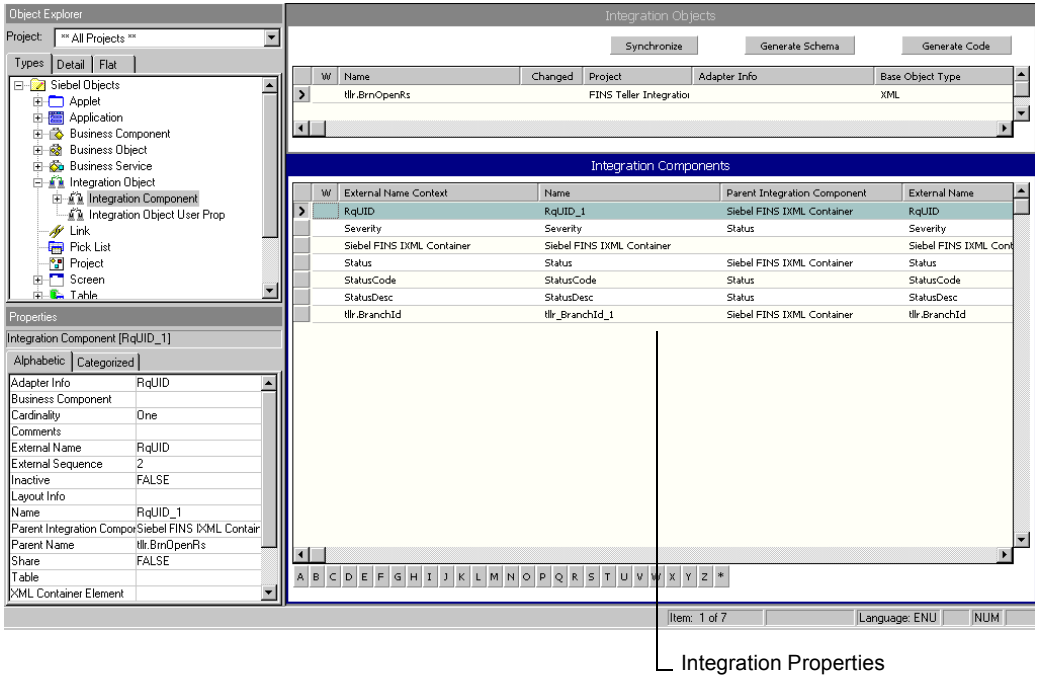
To learn more about the external integration object, open and study a standard external integration object, such as tllr.BrnOpenRs, in Siebel Tools. Examine how the hierarchy is preserved, how the cardinality and data type are indicated, and how to relate an integration component to an XML element in the message definition.

To open the integration object definition in Siebel Tools

1. Open Siebel Tools.
2. Choose Integration Object from the Object Explorer Window.
3. In the Objects Window, query for integration object tllr.BrnOpenRs.

4 Expand the Integration Object in the Object Explorer Window and choose the child object Integration Component.

The integration components are shown in the figure below.



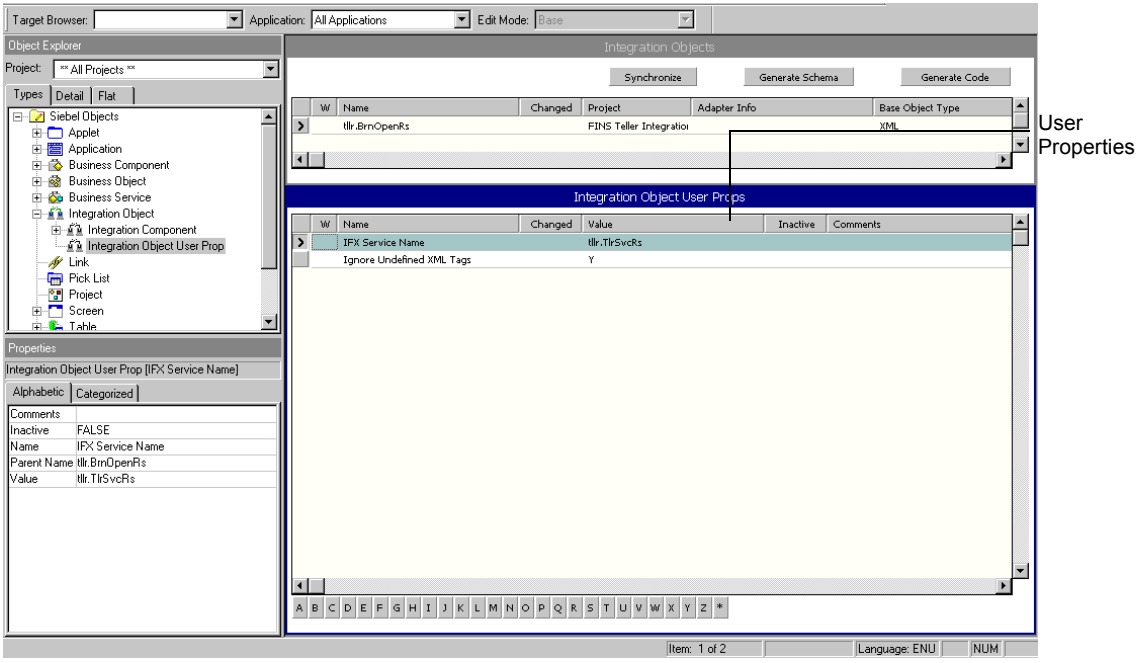
Notice the following features of the integration components:

- Aggregates and elements are treated the same and map to integration components.
- SiebelFINSIXMLContainer is always the root integration component. This is a system-defined integration component and should not be modified.
- The property Parent Integration Component and the property Cardinality are used to preserve the hierarchy and cardinality information.
- The integration component name mimics the XML tag except:
  - The dot in the XML tag is replaced with an underscore.
  - Numbers are appended to resolve naming conflicts.

- If the integration component represents an element that can have a value, it will have at least one Integration Component Field child object, which is PCDATA.
- If the integration component represents an aggregate that cannot have a value, it will have no Integration Component Field child object.

5 In the Object Explorer Window, choose child object Integration Object User Prop.

The user properties are shown in the figure below.



Notice the following features of the integration objects user properties:

- Property IFX Service Name is used to provide information about what IFX Service this message belongs to. It is a required user property.

- Property Ignore Undefined XML Tags is for internal use only and is a required user property.

NOTE: If the underlying DTD changes, it is necessary to revise your integration object definition accordingly. Be careful of changes to commonly used aggregates, as this will force you to regenerate external integration objects for many messages.

Configuring Internal Integration Object

An internal integration object is a conventional Siebel integration object. It creates a structure that matches the data structure of a Siebel business object. It contains interfaces that allow outside systems to interact with internal Siebel data.

Creating New Internal Integration Objects Versus Reusing Existing Internal Integration Objects

You can choose an existing internal integration object for your message or create a new integration object.

In the following situations, consider creating a new internal integration object:

- When no existing integration object matches the business object you want to map to.
- When the existing integration object that matches the business object you want to map to does not include all the integration components that you need.
- When the existing integration object includes many integration components that you do not need.

Internal integration objects can be created using EAI Siebel Wizard, FINS Teller Integration Branch Wizard, FINS Teller Integration Drawer Wizard, or FINS Teller Integration Transaction Wizard.

The internal integration object definition mimics the business object definition. Once the business object definition or underlying business component definition changes, it is necessary to synchronize the integration object definition.

If you change the DTD and update your external integration objects, it is important to review your DTE maps to make sure that they are still valid and mapping the correct fields.

Creating Integration Objects

Whenever a new message is integrated, the following objects are affected in the following ways:

- An external integration object request/response pair needs to be created by the Wizard.
- An internal integration object request/response pair can be reused or needs to be created by the wizard.
- The dispatcher map integration object is updated with the new entry for this message.

You can use a single internal integration object to support request and response messages. However, because of the different message structures of request and response XML messages, you must have two different external integration objects.

NOTE: If you are using a SetFieldValue runtime event to trigger messages, use two different internal integration objects based on different business components for requests and responses as described in "Runtime Event Problems and Solutions" on page 120.

The following workflow describes how to create integration objects for integration of tllr.BrnOpenRq/Rs messages using the FINS Teller Integration Branch wizard.

1. **Lock the project and create a dispatcher map integration object.** For more information, see "Preparation: Locking the Project and Creating the Dispatcher Map Integration Object" on page 102.
2. **Lock the project and select the DTD file.** For more information, see "Locking the Project and Selecting the DTD File" on page 102.
3. **Create request and response external integration objects.** For more information, see "Creating External Integration Objects" on page 103.
4. **Create an internal integration object or objects.** For more information, see "Creating New Internal Integration Objects" on page 105.

Preparation: Locking the Project and Creating the Dispatcher Map Integration Object

Before using the wizard to create integration objects for the first time, check to see if the DispatcherMap integration object already exists. If it does not exist, create an empty integration object with the name specified in the DispatcherMapName user property as described below.

To create an empty integration object

1. Start Siebel Tools.
2. Lock the project under which you want the dispatcher map to be created.
3. From the Object Explorer window, select Integration Object.
4. Create a new record.
5. For the name property, enter the name used in the DispactherMapName user property of the Wizard.

Locking the Project and Selecting the DTD File

When you create a FINS Teller integration object based on the DTD, you must lock the project you are working with in Siebel Tools. The Teller XML wizard requires that the project be locked.

To lock the project and select the DTD file

1. Lock the project in which you will create the integration objects.
2. Lock the project used by the dispatcher map.
3. Choose File > New Object.

   The New Object Wizards dialog box appears.
4. Select the EAI tab and click Integration Object.

   The Integration Object Builder wizard appears.
5. Select the project you locked.

6 Select FINS Teller Integration Branch Wizard Service from the Business Service list.

7 Click Next.

You may receive a warning that reads "Attach the Wizard to a Dispatcher Map." This warning means that the project does not contain a dispatcher map. The warning offers to write to a temporary dispatcher map, which you can merge with the connector's proper dispatcher map later on.

8 Choose the Teller DTD file you want to use and click Next.

It may take some time for the wizard to parse the DTD file and to display the next page.

Creating External Integration Objects

An external integration object establishes the hierarchy for an IFX XML message. Each Teller message needs one and only one integration object that defines the message portion of the IFX XML document. The integration object preserves the cardinality, data type information, and hierarchy.

Every time you run the wizard, the wizard creates a pair of external integration objects, one for the request portion of the cycle and one for the response portion of the cycle.

To create external integration objects

1 Lock the project and select the FINS Teller IFX DTD file, usually named teller.dtd.

2 Create a default envelope integration object.

You will reuse the existing FINS IFX Envelope v101 envelope integration object, so you can skip the envelope creation step of the wizard.

3 Choose a Request Service and Response Service pair and click Next.

For external integration object creation, you need to specify a request service and a response service, for example <tllr.TlrSvcRq> and <tllr.TlrSvcRs>.

These are the service aggregate elements.

4 Choose the Request Message you want to use.

The request message is automatically paired with an appropriate response message. For this example, choose <tllr.BrnOpenRq> and it will automatically be paired with <tllr.BrnOpenRs>.

5 Note the integration object names.

You will need to know the names when you configure the DTE map.

You can change the Integration Object Name for the request and response integration objects for administrative convenience. Consider establishing a set of naming conventions to make groups of objects easy to recognize. This example uses the default names.

6 Click Next to display the Message Elements screen, in which you will select the request message elements to include.

This screen displays a hierarchy of the message structure. It provides all the available aggregates and elements for the message. By default, all elements are selected.

7 Select the Teller message components you want to activate for this integration object.

Notice that if you deselect the parent, all the child items are deselected. Reselecting the parent does not reselect the child items, so that you can select a subset of child items. For improved performance, select only the fields that you might use in a message.

NOTE: The smaller the integration object is, the faster the connector will run. If you exclude too many components and you need to redo the external integration object, you will need to check your DTE maps to make sure they are still valid.

---

8 Click Next to display the second Message Elements screen and choose the Teller message components you want to activate for the response integration object.

9 Click Next to display the Select Internal Integration Object screen.

For information on creating a new internal integration object, see the next section.

Creating New Internal Integration Objects

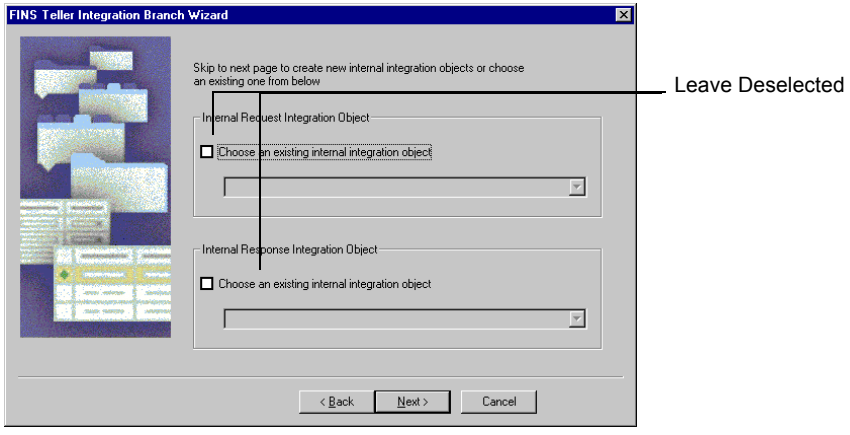
An internal integration object creates a structure that matches the data structure of a Siebel business object. An internal integration object contains interfaces for outside systems to interact with internal Siebel data.

This example uses the IFX wizard to create a new internal integration object, as shown in the following procedure.

To create an internal integration object

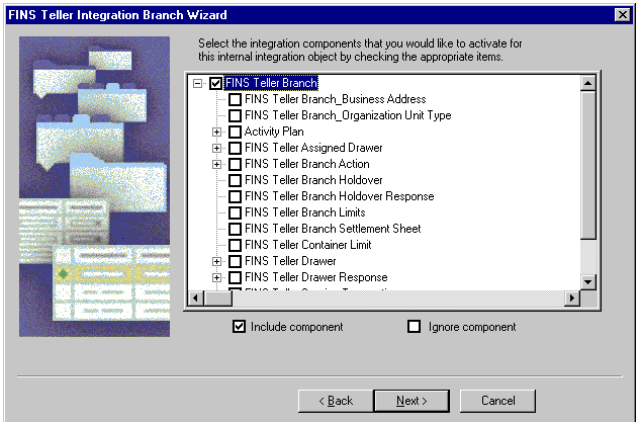
1. Lock the project and select the FINS Teller IFX DTD file.
2. Create a default envelope integration object and external integration objects.

   You should use the predefined envelope integration called FINS IFX Envelope v101, so you can skip the step to create envelope integration objects.
3. From the Select Internal Integration Object screen, leave the check boxes deselected and click Next to display the New Integration Object screen.



4. Under Internal Request Integration Object, choose the business object that contains the information to be exchanged with the connector, and enter the name of the integration object.

5 Click Next to display the integration components available from the business object you selected in the previous screen.



This screen displays a hierarchy of the business object structure. It provides all the available aggregates and elements for the message business components and fields available for mapping into the message.

6 Select the elements you want to include and click Next.

For improved performance, select only the fields that you might use in a message. The smaller the integration object is, the faster the connector will run. If you exclude too many components and you have to redo the internal integration object, you must check your DTE maps to make sure they are still valid.

The screen displays a warning about how long it might take to create the integration objects.

7 Click Yes to create the integration objects.

A window appears, indicating which objects have been created.

8 Close the window.

9 In the Object Explorer window, select Integration Object.

Compiling the Integration Object

After you create the integration objects, compile them into the repository file your application will use.

To compile the integration objects

1. From the Application-level menu, choose View > Site Map > Siebel Tools > Compile > Projects.

   The Object Compiler dialog box appears.

2. Choose the project.

3. Click Compile.

Configuring the Connector Components

As described in Chapter 2, "Teller IFX XML Connector," Siebel Connector for Teller provides four pre-built business services that you can configure for your specific use:

- FINS IFX XML Transaction Manager
- FINS IFX XML Data Transformation Engine
- FINS IFX XML Converter
- FINS IFX XML Dispatcher

Each business service has its own user properties. The values of these user properties are decided at configuration time. However, you can also override those values in the workflow by entering a run-time value. The meanings of the user properties are described in Chapter 2, "Teller IFX XML Connector." This section discusses the configuration of the user properties for each of these business services.

About FINS IFX XML Transaction Manager

The Transaction Manager provides several pre-built operations. These operations are sufficient to support most needs in the FINS IFX connector. Change these values only if you need to add new operations.

Table 40 displays user properties for the Transaction Manager.

Table 40. User Properties for the Transaction Manager

Name	Value	Description
BlankIOI	FINS IFX XML Extension/ CreateIntObjInstance/	Creates an integration object instance containing all the elements defined in the integration object with empty values. Follows the format Service/ Method/Argument;Argument;
QuickQuery	FINS IFX Extension/QuickQuery/ #XMLHierarchy;BC Name;ReturnField;SearchSpec;	Retrieves the value of <ReturnField> of the first record of the search result based on <SearchSpec> against the <BC Name> business component. Follows the format /Method/ Argument;Argument;
SAQuery	EAI Siebel Adapter/Query/	Executes the EAI Siebel Adapter's Query method. Service, method, and argument are separated by "/"
SARowIdQuery	EAI Siebel Adapter/Query/ PrimaryRowId;!SiebelMessage;	Each argument ends with ";"
SASynchronize	EAI Siebel Adapter/Synchronize/	Executes the EAI Siebel Adapter's Synchronize method. The default Service name is "EAI Siebel Adapter."
SAUpsert	EAI Siebel Adapter/Upsert/	Executes the EAI Siebel Adapter's Upsert method. The default argument name is "SiebelMessage."

FINS IFX XML Data Transformation Engine

You do not need to provide new values for the pre-built business service.

FINS IFX XML Converter

Set the user property values according to Table 41. These values will appear in the processing instruction section of an IFX message.

Table 41. User Properties for Converter

Name	Value	Description
XMLEnvIntObjectName	FINS IFX Envelope v101	The envelope integration object to use. You should use the prebuilt FINS IFX Envelope v101 integration object for your Teller IFX envelopes.
EscpeNames	TRUE	Instructs the converter to escape special characters in generated text strings.
PI_Parameter:version	V1.0.1	This value is mapped to the <?ifx version = "" > processing instruction in the XML file header.
PI_Parameter:newfileuid	(default empty)	This value is mapped to the <?ifx newfileuid = "" > processing instruction in the XML file header.
PI_Parameter:oldfileuid	(default empty)	This value is mapped to the <?ifx oldfileuid = "" > processing instruction in the XML file header.

FINS IFX XML Dispatcher

For the Dispatcher user properties, enter the names of the dispatch map and the envelope integration object created by the FINS IFX wizard.

Table 42 displays the dispatcher user properties.

Table 42. FINS IFX XML Dispatcher User Properties

Name	Value
DispatcherMapName	IFXDispMap
XMLEnvIntObjectName	NewDefaultEnv

NOTE: After configuring each business service, you need to compile the new business service definition into the application repository file. Follow the procedure in "Compiling the Integration Object" on page 107.

Configuring the Data Transformation Maps

When you configure the integration objects, the fields in an internal integration object are associated with the message elements in an external integration object. The result is the creation of the DTE map that will be used by the data transformation engine.

All entries created by the wizard are stored in the Integration Object User Properties of the Dispatcher Map.

In the example, you need to configure four maps in order to make a complete outbound/inbound transaction route available. You can find each map in the user properties entry in the IFXDispMap dispatch map integration object.

The integration object for the server entry is IFX/PaySvcRq/PmtAddRq. It has the following two maps:

- PmtAddRq_ERqIRqMapIn (server receiving incoming request)
- PmtAddRq_IrsERsMapOut (server sending outgoing response)

The integration object for the client is IFX/PaySvcRs/PmtAddRs. It has the following two maps:

- PmtAddRs_IRqERqMapOut (client sending outgoing request)
- PmtAddRs_ErsIRsMapIn (client receiving incoming response)

You can change the map name in the Dispatcher Map list and use the new name for the DTE map.

The following procedure provides instructions for configuring one map. For more information, see *Siebel Financial Services eBusiness Application Integration Guide*.

To configure the DTE map

1. Open the Integration Administration View.
2. From the Application-level menu, choose View > Site Map > Integration Administration.

3 Click Data Map Editor and in the Integrated Object Map Applet, create a new map.

   Give the map the same name as the DTE map name created by the wizard and stored in the Dispatcher Map list.

4 Select the Internal Integration Object and the External Integration Object you created with the wizard.

   Keep the following definitions in mind:

   - Source Object: For an outgoing message, the source object is the internal integration object. For an incoming message, the source object is the external integration object.
   - Target Object: For an outgoing message, the target object is the external integration object. For an incoming message, the target object is the internal integration object.

5 Map the source components and the target components.

6 Map fields to fields.

For detailed information on creating and using dispatch rules, see *Siebel Financial Services eBusiness Application Integration Guide*. For more information on data mapping, see *Business Processes and Rules: Siebel eBusiness Application Integration Volume IV*.

Configuring the Workflow Process

The example in this section shows how to modify an existing workflow process to add an additional outbound Teller XML message. Keep the following distinctions between outbound and inbound in mind.

Outbound messages start in a Siebel client. The Siebel application sends an outgoing request to the external data source and receives an incoming response from the external data source. Usually, this is for a client's transaction.

Inbound messages start in an external data source. The Siebel application receives an incoming request from the external data source and sends an outgoing response to the external data source. Usually, this is for a server's transaction.

All Teller messages are outbound messages. You can configure the connector to handle inbound messages, but no inbound messages are configured out-of-the-box.

Whether outbound or inbound, the cycle consists of an outgoing and an incoming message (request and response). In an outbound message, the outgoing part of the cycle occurs first (request), and the incoming part occurs second (response). In an inbound message, the incoming part of the message occurs first (request) and the outgoing part occurs second (response).

Adding a New Message to a Workflow

To support a new message, modify the processes in the Business Object-dependent workflow layer as described in the following procedure.

To add a new message to a workflow

1. Identify the business object and business component or components that will be active when the message is initiated.
2. Find the existing business object layer workflow process that is based on the same business object in Step 1.

   The existing workflow processes follow a naming convention to make them easier to find. For example, the business object layer workflow process that handles messages on the FINS Teller Branch business object is called the FINS Teller Branch Connector.

3 From the Siebel Workflow Administration screen, Workflow Processes view, revise the workflow in Step 2 on page 114.

4 Add a branch and a wait step in a manner similar to that used in the existing messages.

5 Edit the wait step so that in the output arguments section, values are assigned to the following process properties:

- Teller IFX Dispatcher Map Integration Object Name
- Teller IFX Message Full Name

The existing Teller workflow processes use the default FINS IFX Envelope v101 envelope, so you do not have to explicitly identify it.

6 If you want to query records from more than one business component (in other words, if your integration object has more than just a primary integration component), set values for the following process properties:

- IFXSearchSpec
- IFXSearchSpec%1

See Table 30 on page 57 for details on setting these values.

7 Edit the branch step near the start of the workflow to branch to your new wait step.

Use the EventName process property to determine which branch to take.

If your message is initiated from a business object that does not have an existing business object-dependent workflow defined, you will need to create a new workflow to support this business object. To do this, copy an existing business object-dependent workflow process, rename it, and change the business object to match the one that is needed to initiate the message. You can then edit the workflow to set the branches and wait steps accordingly.

For more information on how to set the EventName process property, see "Initiating Messages" on page 116.

Initiating Messages

XML messages can be initiated from the UI in one of two ways: Runtime Events or Command Objects. Each has a different set of advantages and disadvantages. Table 43 displays a list of factors and whether or not they are applicable to runtime events and command objects.

Table 43. Runtime Event and Command Object Factors

Factors	Runtime Events	Command Objects
Can be associated with applet-level menus	No	Yes
Can be triggered by hotkeys	No	Yes
Can be triggered by business component, applet, or application events	Yes	No
Configured using seed data and can be adjusted without restarting the server	Yes	No
Configured using Tools and compiled to an SRF file	Usually no. But if your runtime event is triggered from a button on the UI, it will require a repository change to configure the button to invoke the correct method.	Yes
Requires a UI element to be associated with it	No	Yes

Runtime Events

Runtime events are seed data that associate an event in the application with a business service to run. The Runtime Events Administration screen is used to administer runtime events. Use the Events view to identify the event that should trigger an action. Use the Action Sets view to define the action that should be taken when the event occurs.

After you change either the runtime events or the action sets, purge the cache by choosing Reload Personalization from the applet-level menu of the Events applet.

For information on the basics of creating and maintaining runtime events and action sets, see *Personalization Administration Guide*.

Action Set Configuration

You configure Teller action sets to run a business service. Table 44 displays the important configuration parameters for action sets.

Table 44. Action Set Configuration Parameters

Action Set Parameter	Value
Action Type	BusService
Business Service Name	Workflow Process Manager
Business Service Method	RunProcess
Business Service Context	“ProcessName”, “FINS Teller Drawer Connector”, “EventName”, “Bait Query”

The values in the business service context parameter are passed to the Workflow Process Manager in a “name, value” pair format. The Process Name parameter identifies the process to run. The EventName parameter identifies a process property in the workflow. Notice that the EventName parameter is the only parameter needed to trigger the message. The rest is handled by the workflow.

When editing the Business Service Context string, you must enclose your parameter names in double quotes (“) and separate parameters using a comma and a single space.

For the runtime event to invoke the workflow successfully, the workflow must have Active status and current date not exceeding the expiration date if an expiration date is set. By default, workflow processes have In Progress status, so they need to be activated from the Siebel Workflow Administration screen.

Runtime Event Configuration

Once the action set is configured, you need to identify the event that will trigger the action set from the Events view of the Runtime Events Administration screen.

Table 45 displays the events commonly used to trigger messages with Teller.

Table 45. Trigger Events

Object Type	Event	Use in Teller
Applet	DisplayApplet	Use to send query messages that are automatically triggered to refresh data when an applet is navigated to. This is used in Teller to make sure the data on the Teller Drawers screen is up to date when a user navigates to it.
Applet	InvokeMethod	Usually issues query messages, but also issues action messages like post transaction or end customer session. These events are usually triggered from buttons on the UI.
Application	Logout WebLogout	These events are used to send the SignoffRq to temporarily sign off from the middleware.
BusComp	InvokeMethod	An alternative way to implement events from buttons on an applet. This is similar to the Applet Invokemethod technique. In this case, the method defined on the applet initiates this event in the defined business component. Defining this event at the business component level is useful if you have a number of applets on the same business component that initiate the same message. Then you can define the event once at the business component level instead of defining it for each applet.
BusComp	PreDeleteRecord	Used to send a message before a record is deleted.
BusComp	SetFieldValue	Used to send 'modify' messages.
BusComp	WriteRecordNew	Used to send 'add' messages or 'Start Customer Session' messages.
BusComp	WriteRecordUpdated	Used to send a message when an entire modified record is saved. Usually used to send modify messages.

Configuring Events in Tools

Of the events listed in Table 45 on page 118, two require additional configuration in Tools: the Applet and BusComp InvokeMethod.

To trigger an event from a Tools object

1. Create a control on an applet.
2. Set the Type property to MiniButton.
3. Set the Method Invoked property to EventMethodxxx, where xxx is the unique name for this event.

   The Method Invoked name must start with EventMethod.
4. Compile the SRF.

Enabling Runtime Events

By default, the Teller runtime events are disabled because the Conditional Expression field is set to 0.

To enable your runtime events

1. Clear the Conditional Expression field.
2. Choose Reload Personalization from the applet-level menu of the Events applet to reload the personalization cache.

Runtime Event Problems and Solutions

This section presents issues that may arise with runtime events and suggests workarounds.

BusComp SetFieldValue and Infinite Message Loops

Using the BusComp SetFieldValue to initiate eAI messages may result in an infinite message loop, as shown in the following scenario.

1. A SetFieldValue event on business component A initiates a message.
2. The connector workflow runs and generates an XML string to send to the middleware.
3. The middleware responds with an XML string.
4. The connector handles the incoming XML string and maps the data back to an internal integration object that updates business component A.
5. In the last step of the incoming workflow, the Siebel Adapter runs, calling SetFieldValue again on business component A, and so initiates a new message. This results in an infinite message loop and crashes the application once it runs out of memory.

To prevent an infinite message loop

1. Create a copy of the business component you want to update.
2. Add it to your business object.

   You will need to create additional links to do this. If this is the primary business component, you will need to create a new business object.
3. Create a new internal integration object using the copy of the business component.
4. Update the proper dispatcher map entry to reference the new internal integration object.

BusComp SetFieldValue and the EAI Siebel Adapter

Using SetFieldValue to fire events does not guarantee that the new data will be written to the database. When a Pick Applet calls SetFieldValue, it forces a write to the database so that the EAI Siebel Adapter will pick up the correct data. However, if a picklist or regular typing is using to set a value in a field, the SetFieldValue event will fire before the data is written to the database. In this case, the EAI Siebel Adapter will read the old data.

To work around this problem, if you need to use a SetFieldValue event, use a WriteRecordUpdated event instead. If using a WriteRecordUpdated event will not work for your situation, consider using a button or Command object instead.

DisplayApplet and Connector Errors

If you use the DisplayApplet event to issue a message and the message returns an error, or the connector has an error, part of the view may not appear. Only use the display applet event in cases where you have a high degree of confidence that the underlying message is reliable.

WriteRecordNew and Connector Errors

If you use WriteRecordNew to initiate a message and the message returns with an Error status, the data from the response message will not be updated into the Siebel application. However, the record in Siebel will no longer be considered a new record. So if the user saves the record in an attempt to initiate the message, the WriteRecordNew event will not fire and no message will be sent.

To work around this problem you will need to implement a WriteRecordUpdated event to send a message when the record is modified and saved again.

NOTE: If no fields have changed in the applet, neither event will fire. If you need complete control over when to send a message, using a button as described above will give you complete control over when the message can be sent.

Phasing of Events and Database State

When various events fire, the database will be in a certain state. Table 46 describes what data you can expect in the database when certain events fire, and what will happen if the associated message fails.

Table 46. events and Data

Event	Database State
WriteRecord WriteRecordNew WriteRecordUpdated	The new or updated record will be committed in the database and available for the Siebel Adapter to read. If the message fails, the record with its new or modified data remains in the Siebel database.
PreDelete	No change is made to the record when this message fires. If the message fails, then the delete record will fail and the record will remain in Siebel.
InvokeMethod EventMethodxxx	Generally these are used for query messages or for posting transactions. If the message fails, no updates to the Siebel database will occur and the existing records will remain unchanged.
SetFieldValue	If the field is associated with a pick applet, the new data from the pick applet will be available when the message fires. If the field is associated with a picklist or if data is entered by typing only, the new data will not be committed to the database when the event fires.

Command Objects

Command objects are configured in Tools. They have a simple configuration, similar to that of Runtime Events.

Table 47 displays the command object attributes.

Table 47. Command Object Attributes

Command Object Attribute	Value
Business Service	FINS Teller UI Navigation
Method	CallBusSvc
Method Argument	'Workflow Process Manager', 'RunProcess', 'ProcessName', 'FINS Teller Branch Connector', 'EventName', 'Branch Open', 'BusObj', 'FINS Teller Branch', 'PrimaryBusComp', 'FINS Teller Branch'

To initiate Teller messages using command objects, the command objects call the business service FINS Teller UI Navigation, which calls the method CallBusSvc. This method is a wrapper that takes a string of “name, value” pairs and then calls the named business service.

Table 48 provides a description of the method argument strings for the CallBusSvc method.

Table 48. CallBusSvc Method Arguments

Method Argument String	Description
'Workflow Process Manager'	The business service to call.
'RunProcess'	The method to call.
'ProcessName', 'FINS Teller Branch Connector'	A parameter to the business service, in this case the name of the workflow process to run.
'EventName', 'Branch Open'	Another parameter to the business service, in this case the value of a process property in the workflow process.

Table 48. CallBusSvc Method Arguments

Method Argument String	Description
'BusObj', 'FINS Teller Branch'	Name of the business object that is active. This is used to get the active row Id to set the Object Id process property value for running a workflow process.
'PrimaryBusComp', 'FINS Teller Branch'	Name of the business component that is active. This is used to get the active row Id to set the Object Id process property value for running a workflow process.

After configuring the command object, you can associate it in Tools with an applet method menu item to have it appear on the given applet. You can also assign it an accelerator key so that the command can be invoked using special keystrokes.

For more information on creating applet-level menu items, see *Siebel Tools Reference*.

Initial Setup

The Teller system depends on seed data for:

- Workflows
- Data Transformation Engine (DTE) maps
- Runtime events

This seed data is usually shipped in a deactivated state. Before running any of the integration functionality on a new database, you will need to activate the Workflows and Runtime Events, and set the correct Transport Manager connect information.

Activating Workflow Seed Data

To activate the workflows

1. Navigate to Siebel Workflow Administration > Workflow Processes.
2. Search for all the workflow processes identified in Table 50 on page 132.

   For a fresh database installation, there should only be one version of each workflow, and it should have a status of In Progress.
3. Activate the latest version of all these workflow processes.

Activating DTE Seed Data

No explicit setup is required to activate the DTE map seed data.

Activating Runtime Event Seed Data

By default, all the Teller runtime events should be deactivated in a fresh database installation.

To activate the Runtime Events

1. Navigate to Runtime Events Administration > Events.
2. Query for all events where the Action Set matches the pattern FINS Teller.

   The Conditional Expression value for all these action sets is 0. This indicates that the action set is disabled.
3. For each action set with the conditional expression set to 0 that you want to activate, erase the conditional expression value so the conditional expression is now null.

   This will enable the runtime event. You should only activate the runtime events that are necessary for your implementation.
4. Clear the runtime event cache by choosing Reload Personalization from the applet-level menu.

Setting the Transport Manager URL

If you are using the http transport to send and receive your Teller IFX XML messages, you will need to set the URL of the Web server used to receive these messages.

You can set the URL in one of two places:

- The business object-dependent workflow layer
- The workflow process FINS IFX XML Transport Manager

It is recommended that you set the connection URL in the business object-dependent workflow layer, and pass it to the FINS IFX XML Transport Manager layer.

If you hardcode the URL in the FINS IFX XML Transport Manager as a literal input value to the Transport Manager business service, all IFX and Teller messages that use the same workflow infrastructure will be forced to send their messages to the same URL.

The following procedure examines a sample workflow process.

To set the URL in a business object-dependent workflow

1. In the Workflow Process screen, search for the FINS Teller Session Connector workflow.
2. Navigate to the Process Properties view.
3. Scroll down until you see a process property called Transport Connect Information.

   This is the property that defines where the XML message will be sent. A default value is displayed here.
4. You can set the correct value here as a default value, or if you want to set a different URL for different messages, you can set it using the existing Wait steps in the workflow process.
5. Make sure each message that is sent out has its own wait step.
6. Set the properties for the wait step using output arguments.
7. Add a record to the Output Argument applet to set the Transport Connect Information process property to the value you want.

Teller Objects A

This appendix describes Siebel Connector for Teller messages, workflow processes and wizards.

Messages

From a message and a dispatcher map, you can identify the DTE map and the integration objects used to implement that message. For details on how to read the dispatcher map to identify the various components, see "Examining an Existing Dispatcher Map" on page 89.

The messages listed here are a subset of the complete Teller message set identified in the DTD and specification. These messages have all the necessary internal and external integration objects, DTE maps, and runtime events configured.

Table 49. Teller Messages

Service	Message	Dispatcher Map Integration Object Name
tllr.TlrSvc	tllr.BaitInqRq/Rs	FINSTlrDrwrDispMap
tllr.TlrSvc	tllr.BrnCloseRq/Rs	FINSTlrBranchDispMap
tllr.TlrSvc	tllr.BrnOpenRq/Rs	FINSTlrBranchDispMap
tllr.TlrSvc	tllr.BrnSettleRq/Rs	FINSTlrBranchDispMap
tllr.TlrSvc	tllr.BrnSettleSheetInqRq/Rs	FINSTlrBranchDispMap
tllr.TlrSvc	tllr.CashDrwrAssignRq/Rs	FINSTlrBranchDispMap
tllr.TlrSvc	tllr.CashDrwrModRq/Rs	FINSTlrDrwrDispMap
tllr.TlrSvc	tllr.CashDrwrInqRq/Rs	FINSTlrBranchDispMap FINSTlrDrwrDispMap
tllr.TlrSvc	tllr.CashDrwrUnassignRq/Rs	FINSTlrBranchDispMap
tllr.TlrSvc	tllr.HoldoverSchedAddRq/Rs	FINSTlrBranchDispMap
tllr.TlrSvc	tllr.HoldoverSchedCanRq/Rs	FINSTlrBranchDispMap
tllr.TlrSvc	tllr.HoldoverSchedInqRq/Rs	FINSTlrBranchDispMap
tllr.TlrSvc	tllr.SetHoldoverStRq/Rs	FINSTlrDrwrDispMap
tllr.TlrSvc	tllr.TrapInqRq/Rs	FINSTlrDrwrDispMap
tllr.TlrSvc	tllr.UserFrcSignoffRq/Rs	FINSTlrBranchDispMap
tllr.TlrSvc	tllr.BatchCloseRq/Rs	FINSTlrDrwrDispMap

Table 49. Teller Messages

Service	Message	Dispatcher Map Integration Object Name
tllr.TlrSvc	tllr.BatchInqRq/Rs	FINSTlrDrwrDispMap
tllr.TlrSvc	tllr.BinInqRq/Rs	FINSTlrBranchDispMap
tllr.TlrSvc	tllr.CashSettleRq/Rs	FINSTlrDrwrDispMap
tllr.TlrSvc	tllr.CashTrayInqRq/Rs	FINSTlrDrwrDispMap FINSTlrBranchDispMap
tllr.TlrSvc	tllr.EndOfDayRq/Rs	FINSTlrDrwrDispMap
tllr.TlrSvc	tllr.TlrExchReconcileRq/Rs	FINSTlrDrwrDispMap
tllr.TlrSvc	tllr.TlrExchStartRq/Rs	FINSTlrDrwrDispMap
tllr.TlrSvc	tllr.CashDrwrTrayXferRq/Rs	FINSTlrDrwrDispMap
tllr.TlrSvc	tllr.CustSessBeginRq/Rs	FINSTlrTranDispMap
tllr.TlrSvc	tllr.CustSessCanRq/Rs	FINSTlrTranDispMap
tllr.TlrSvc	tllr.CustSessEndRq/Rs	FINSTlrTranDispMap
tllr.TlrSvc	tllr.DepAddRq/Rs	FINSTlrTranDispMap
tllr.TlrSvc	tllr.EJTrnInqRq/Rs	FINSTlrTranDispMap
tllr.TlrSvc	tllr.XferAddRq/Rs	FINSTlrTranDispMap
BaseSvc	SvcAcctInqRq/Rs	FINSTlrTranDispMap
tllr.TlrSvc	tllr.TlrPmtAddRq/Rs	FINSTlrTranDispMap
tllr.TlrSvc	tllr.OverShortAddRq/Rs	FINSTlrDrwrDispMap
tllr.TlrSvc	tllr.TrnRvrsRq/Rs	FINSTlrTranDispMap
tllr.TlrSvc	tllr.WithdrwlAddRq/Rs	FINSTlrTranDispMap

Workflow Processes

The Workflow Processes shown in Table 50 are configured to support the Teller operations.

Table 50. Workflow Processes

Process Name	Description
FINS IFX Connector Outbound	IFX Connector workflow used by Teller
FINS IFX Connector Incoming	IFX Connector workflow used by Teller
FINS IFX Connector Outgoing	IFX Connector workflow used by Teller
FINS IFX Connector Transport Manager	IFX Connector workflow used by Teller
FINS Teller Session Connector	Manage messages initiated from the FINS Teller Session business object
FINS Teller Branch Connector	Manage messages initiated from the FINS Teller Branch business object
FINS Teller Drawer Connector	Manage messages initiated from the FINS Teller Drawer business object
FINS Teller Contact Connector	Manage messages initiated from the Contact business object
FINS Teller Employee Connector	Manage messages initiated from the FINS Teller Employee business object
FINS Teller Exchange Reconcile Connector	Manage messages initiated from the FINS Teller Reconcile Exchange business object
FINS Teller Login Connector	Manage IFX signon/signoff messages
FINS Teller Over/Short Post	Create a new over/short record during a cash settlement
FINS Teller Override Connector	Manage the navigation to the FINS Teller Override View and collection of override user name and password
FINS Teller Start Session	Used by the Go button on the FINS Contact Summary - Teller View

Table 50. Workflow Processes

Process Name	Description
FINS Teller VBC Connector	Used by all of the Teller messages that are based on VBCs.
FINS Teller Create Customer Session	Creates a new customer session and transaction. Used by the Go button on the FINS Contact Summary - Teller View.

Wizards

The IFX Wizards shown in Table 51 are configured to support Teller operations.

Table 51. IFX Wizards

Wizard Business Service Name	Dispatcher Map Updated	Tools Project for New Integration Objects
FINS Teller Integration Branch Wizard	FINSTlrBranchDispMap	FINS Teller Integration Branch
FINS Teller Integration Drawer Wizard	FINSTlrDrwrDispMap	FINS Teller Integration Drawer
FINS Teller Integration Transaction Wizard	FINSTlrTranDispMap	FINS Teller Integration Transaction