PeopleSoft®

PeopleTools 8.42 Integration Tools PeopleSoft Business Interlink Design-Time Plug-in Programming Guide

November 2002

PeopleTools 8.42
Integration Tools
PeopleSoft Business Interlink Design-Time Plug-in Programming Guide
SKU TOOLS842BID-B 1102

PeopleBooks Contributors: Teams from PeopleSoft Product Documentation and Development.

Contents

About This PeopleBook

PeopleBooks provide you with the information that you need to implement and use PeopleSoft applications.

This preface discusses:

- PeopleSoft application prerequisites.
- PeopleSoft application fundamentals.
- Related documentation.
- Typographical elements and visual cues.
- Comments and suggestions.
- Common elements in PeopleBooks.

Note. PeopleBooks document only page elements that require additional explanation. If a page element is not documented with the process or task in which it is used, then either it requires no additional explanation or it is documented with common elements for the section, chapter, PeopleBook, or product line. Elements that are common to all PeopleSoft applications are defined in this preface.

PeopleSoft Application Prerequisites

To benefit fully from the information that is covered in these books, you should have a basic understanding of how to use PeopleSoft applications.

See *Using PeopleSoft Applications.*

You might also want to complete at least one PeopleSoft introductory training course.

You should be familiar with navigating the system and adding, updating, and deleting information by using PeopleSoft windows, menus, and pages. You should also be comfortable using the World Wide Web and the Microsoft Windows or Windows NT graphical user interface.

These books do not review navigation and other basics. They present the information that you need to use the system and implement your PeopleSoft applications most effectively.

PeopleSoft Application Fundamentals

Each application PeopleBook provides implementation and processing information for your PeopleSoft database. However, additional, essential information describing the setup and design of your system appears in a companion volume of documentation called the application fundamentals PeopleBook. Each PeopleSoft product line has its own version of this documentation.

The application fundamentals PeopleBook consists of important topics that apply to many or all PeopleSoft applications across a product line. Whether you are implementing a single application, some combination of applications within the product line, or the entire product line, you should be familiar with the contents of this central PeopleBook. It is the starting point for fundamentals, such as setting up control tables and administering security.

Related Documentation

This section discusses how to:

- Obtain documentation updates.
- Order printed documentation.

Obtaining Documentation Updates

You can find updates and additional documentation for this release, as well as previous releases, on the PeopleSoft Customer Connection Website. Through the Documentation section of PeopleSoft Customer Connection, you can download files to add to your PeopleBook Library. You'll find a variety of useful and timely materials, including updates to the full PeopleSoft documentation that is delivered on your PeopleBooks CD-ROM.

Important! Before you upgrade, you must check PeopleSoft Customer Connection for updates to the upgrade instructions. PeopleSoft continually posts updates as the upgrade process is refined.

See Also

PeopleSoft Customer Connection Website, http://www.peoplesoft.com/corp/en/login.asp

Ordering Printed Documentation

You can order printed, bound volumes of the complete PeopleSoft documentation that is delivered on your PeopleBooks CD-ROM. PeopleSoft makes printed documentation available for each major release shortly after the software is shipped. Customers and partners can order printed PeopleSoft documentation by using any of these methods:

- Web
- Telephone
- Email

Web

From the Documentation section of the PeopleSoft Customer Connection Website, access the PeopleSoft Press Website under the Ordering PeopleBooks topic. The PeopleSoft Press Website is a joint venture between PeopleSoft and Consolidated Publications Incorporated (CPI), the book print vendor. Use a credit card, money order, cashier's check, or purchase order to place your order.

Telephone

Contact CPI at 800 888 3559.

Email

Send email to CPI at psoftpress@cc.larwood.com.

See Also

PeopleSoft Customer Connection Website, http://www.peoplesoft.com/corp/en/login.asp

Typographical Conventions and Visual Cues

This section discusses:

- Typographical conventions.
- Visual cues.

Typographical Conventions

The following table contains the typographical conventions that are used in PeopleBooks:

Typographical Convention or Visual Cue	Description
Bold	Indicates PeopleCode function names, method names, language constructs, and PeopleCode reserved words that must be included literally in the function call.
Italics	Indicates field values, emphasis, and PeopleSoft or other book-length publication titles. In PeopleCode syntax, italic items are placeholders for arguments that your program must supply. We also use italics when we refer to words as words or letters as letters, as in the following: Enter the number 0, not the letter *O*.
KEY+KEY	Indicates a key combination action. For example, a plus sign (+) between keys means that you must hold down the first key while you press the second key. For ALT+W, hold down the ALT key while you press W.
`Monospace font`	Indicates a PeopleCode program or other code example.
(quotation marks)	Indicate chapter titles in cross-references and words that are used differently from their intended meanings.

Typographical Convention or Visual Cue	Description
. . . (ellipses)	Indicate that the preceding item or series can be repeated any number of times in PeopleCode syntax.
{ } (curly braces)	Indicate a choice between two options in PeopleCode syntax. Options are separated by a pipe (|).
[ ] (square brackets)	Indicate optional items in PeopleCode syntax.
& (ampersand)	When placed before a parameter in PeopleCode syntax, an ampersand indicates that the parameter is an already instantiated object. Ampersands also precede all PeopleCode variables.
(ISO)	Information that applies to a specific country, to the U.S. federal government, or to the education and government market, is preceded by a three-letter code in parentheses. The code for the U.S. federal government is USF; the code for education and government is E&G, and the country codes from the International Standards Organization are used for specific countries. Here is an example: (GER) If you're administering German employees, German law requires you to indicate special nationality and citizenship information for German workers using nationality codes established by the German DEUEV Directive.
Cross-references	PeopleBooks provide cross-references either below the heading See Also or on a separate line preceded by the word *See.* Cross-references lead to other documentation that is pertinent to the immediately preceding documentation.

Visual Cues

PeopleBooks contain the following visual cues.

Notes

Notes indicate information that you should pay particular attention to as you work with the PeopleSoft system.

Note. Example of a note.

A note that is preceded by *Important!* is crucial and includes information that concerns what you must do for the system to function properly.

Important! Example of an important note.

Warnings

Warnings indicate crucial configuration considerations. Pay close attention to warning messages.

Warning! Example of a warning.

Comments and Suggestions

Your comments are important to us. We encourage you to tell us what you like, or what you would like to see changed about PeopleBooks and other PeopleSoft reference and training materials. Please send your suggestions to:

PeopleSoft Product Documentation Manager PeopleSoft, Inc. 4460 Hacienda Drive Pleasanton, CA 94588

Or send email comments to doc@peoplesoft.com.

While we cannot guarantee to answer every email message, we will pay careful attention to your comments and suggestions.

Common Elements in These PeopleBooks

As of Date	The last date for which a report or process includes data.
Business Unit	An ID that represents a high-level organization of business information. You can use a business unit to define regional or departmental units within a larger organization.
Description	Enter up to 30 characters of text.
Effective Date	The date on which a table row becomes effective; the date that an action begins. For example, to close out a ledger on June 30, the effective date for the ledger closing would be July 1. This date also determines when you can view and change the information. Pages or panels and batch processes that use the information use the current row.
Once, Always, and **Don't Run**	Select Once to run the request the next time the batch process runs. After the batch process runs, the process frequency is automatically set to Don't Run. Select Always to run the request every time the batch process runs. Select Don't Run to ignore the request when the batch process runs.

Report Manager	Click to access the Report List page, where you can view report content, check the status of a report, and see content detail messages (which show you a description of the report and the distribution list).
Process Monitor	Click to access the Process List page, where you can view the status of submitted process requests.
Run	Click to access the Process Scheduler request page, where you can specify the location where a process or job runs and the process output format.
Request ID	An ID that represents a set of selection criteria for a report or process.
User ID	An ID that represents the person who generates a transaction.
SetID	An ID that represents a set of control table information, or TableSets. TableSets enable you to share control table information and processing options among business units. The goal is to minimize redundant data and system maintenance tasks. When you assign a setID to a record group in a business unit, you indicate that all of the tables in the record group are shared between that business unit and any other business unit that also assigns that setID to that record group. For example, you can define a group of common job codes that are shared between several business units. Each business unit that shares the job codes is assigned the same setID for that record group.
Short Description	Enter up to 15 characters of text.

See Also

Using PeopleSoft Applications

PeopleSoft Process Scheduler

PeopleSoft Business Interlink Design-Time Plug-in Programming Guide Preface

This PeopleBook covers the concepts of Business Interlinks. It discusses how developers design the shape for a Business Interlink, and how they create Business Interlink XML Design-Time Plug-ins from that shape. The Business Interlink XML Design-Time Plug-in is written in the XML markup programming language. An in-depth knowledge of XML is not needed, but some knowledge of XML syntax could be helpful. This PeopleBook shows and explains the syntax that you need to write a Business Interlink XML Design-Time Plug-in.

The "About These PeopleBooks" preface contains general product line information, such as related documentation, common page elements, and typographical conventions. This preface also contains a glossary with useful terms that are used in PeopleBooks.

CHAPTER 1

Understanding Business Interlinks

This chapter discusses:

- Business Interlinks.
- Business Interlink architecture.
- Business Interlink actions.
- Business Interlink creation.
- An example of how Business Interlinks can be used.
- An example of an XML design-time plug-in.
- Common Business Interlink Terms.

Business Interlinks

Business Interlinks enable you to perform component-based, realtime integration from PeopleSoft to external systems. Business Interlinks do this by creating synchronous transactions that allow PeopleSoft applications to pass data to and receive data from the external system in real time. You can use Business Interlinks to integrate PeopleSoft with external systems, with another PeopleSoft application, and systems on the internet web.

A transaction consists of a named command with optional named and typed inputs and outputs. The associated external system or the Business Interlink Plug-in understands this command.

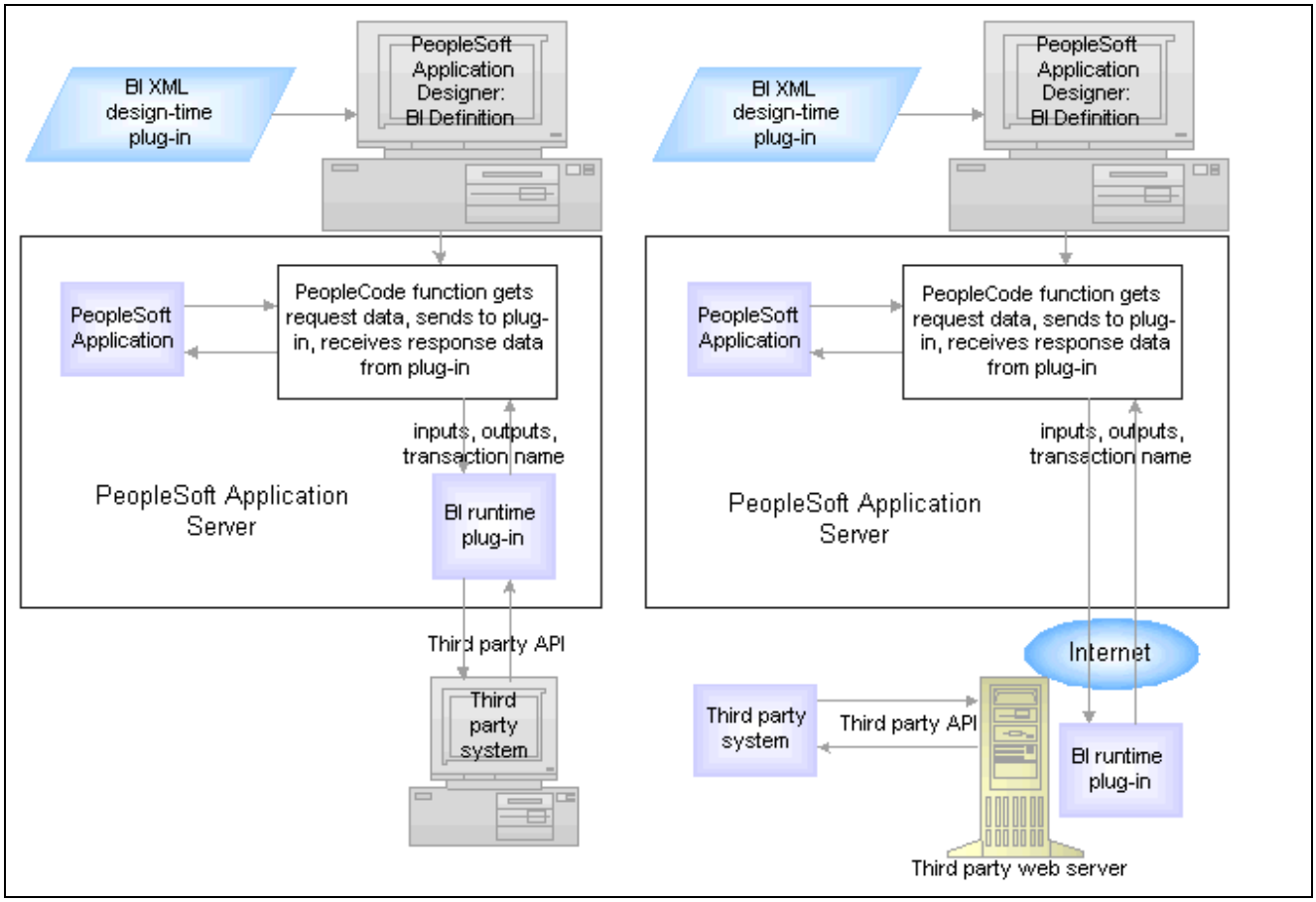
Business Interlink Architecture

The Business Interlink Architecture provides a plug-in framework for PeopleSoft applications to invoke third party APIs over the Internet. Different vendors support different methods for exposing their APIs, including object technologies such as COM, COBRA, EJB; programming language specific interfaces for C or C++; or interfaces based on HTTP and XML. The Business Interlink framework provides a consistent framework for application developers to invoke external applications across this wide variety of technologies.

When a business event triggers the execution of a Business Interlink, the Component Processor synchronously calls the Business Interlink Processor, which in turn invokes the appropriate Business Interlink plug-in. The plug-in provides a wrapper around the third party API and is designed to support any type of interface binding (COM, CORBA, EJB, XML) exposed by the third-party interface. The third-party software could be hosted on the same machine as the PeopleSoft Internet Application Server, or on a separate machine on the other side of the world, invoked over the Internet.

Business Interlink Actions

The following diagram shows the typical Business Interlink architecture, using an XML design-time plug-in and a runtime plug-in.



Business Interlink Architecture

When the Business Interlink Object is executed, the following actions take place:

1. A PeopleSoft Application triggers a PeopleCode event.

   For example, a user might, in a PeopleSoft page labeled “Shipping Time and Cost,” enter input values needed to calculate shipping time, and then press a button labeled “Calculate.”

2. The triggered event (pressing the “Calculate” button) passes the input values to a PeopleCode program, and then executes the PeopleCode program.

3. The PeopleCode program creates an Interlink Object, which contains the transaction name and inputs, and passes the Interlink Object to the Business Interlink runtime plug-in.

   The runtime plug-in can be located on:

   - A web server.
   - The same PeopleSoft Application Server as the PeopleSoft Application.

4. The Business Interlink runtime plug-in provides the implementation of the transactions and/or data operations.

It calls the external software system though its API, passing the input values from the Business Interlink Object. This external system can be located on:

- An external server.
- A web server.
- The same PeopleSoft Application Server as the PeopleSoft Application.

5. The third party system performs operations based on those input values, and returns output values thought its API to the runtime plug-in.

   These operations could be, for example, executing functions in the external system, or performing operations on a database in the external system.

6. The Business Interlink runtime plug-in assigns output values to the Business Interlink Object (if there are outputs).

   If there are outputs, the PeopleCode program can assign the output values to PeopleSoft variables. For example, in a PeopleSoft page labeled "Shipping Time and Cost," the output values could then be displayed upon that page.

Business Interlink Creation

To allow users to create and run Business Interlinks, developers must perform the following tasks. This manual describes the tasks that are *emphasized*.

1. *A developer designs the shape of a transaction(s) (inputs, outputs, name). For example, the action could be telling PSCustomer to calculate the cost of shipping.*
2. *A developer writes a Business Interlink design-time plug-in that implements the shape of the transaction(s).*

   This plug-in is written in the XML markup language.

3. A developer writes a Business Interlink runtime plug-in to implement the transaction: passing the inputs to an external system and receiving the outputs form the external system.
4. *The design-time plug-in* and the runtime plug-in *are deployed for use by PeopleSoft applications.*
5. A PeopleSoft Application Developer creates a Business Interlink Definition, which creates a specific shape for a particular PeopleSoft application.

   For example, the application might be created, or modified, to be able to connect to PSCustomer and calculate shipping costs.

6. A PeopleSoft Application Developer writes a PeopleCode program to call the Business Interlink object that is created for the Business Interlink definition.
7. A user can now use the PeopleSoft application to run the Business Interlink.

   For example, the use can now connect to PSCustomer and calculate shipping costs.

Example of Calculating Shipping Costs

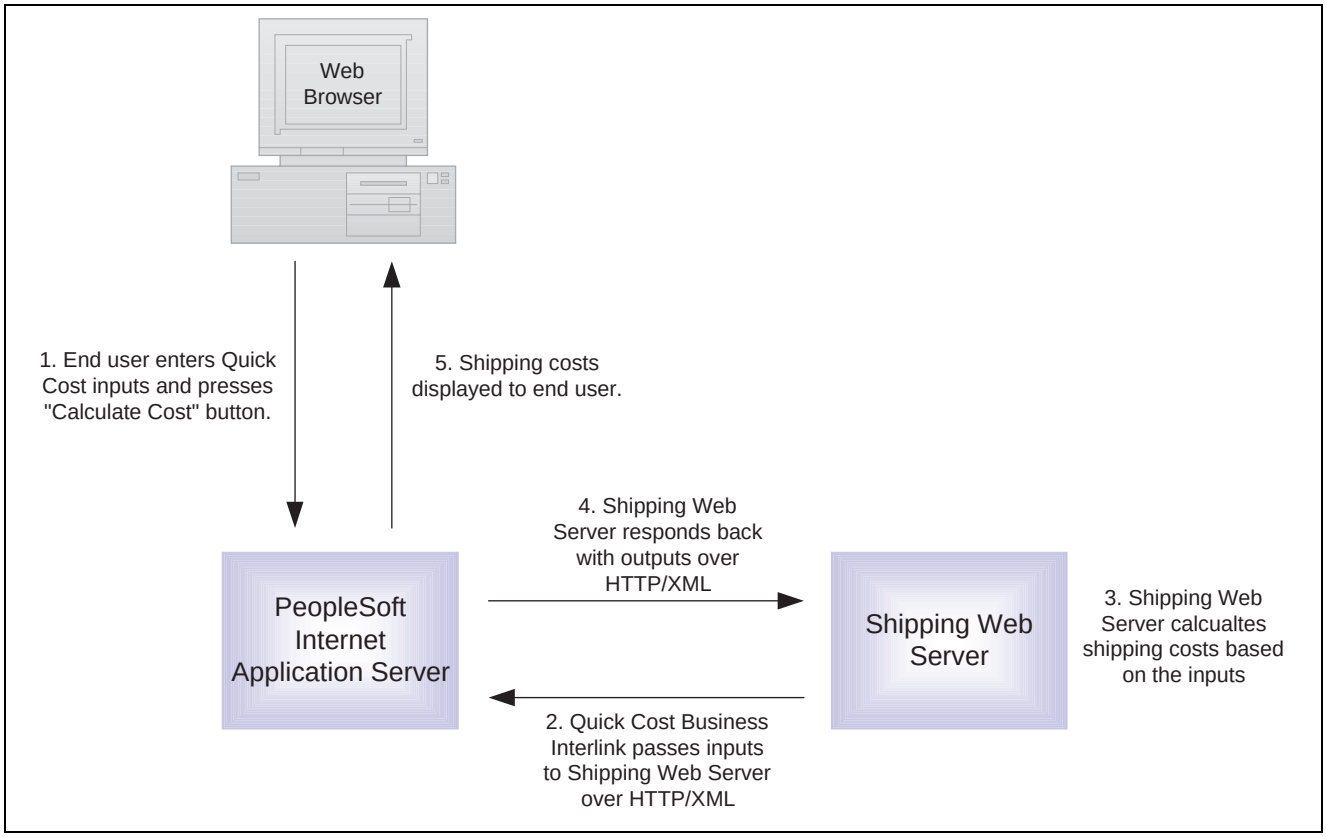
The Shipping Time & Cost example shows how Business Interlinks can be used.

In a Shipping and Time page, you can trigger two different Business Interlinks that call out to the PScustomer shipping and tracking Web site to calculate:

- Ground Time-in-Transit to determine how long it will take to deliver the package.
- Quick Cost to calculate how much it will cost to ship the package based on the shipping service type.

For Quick Cost, you first enter Origin Country and Postal Code, Destination Country, Postal Code, Packaging information, and then you press the Calculate button. At this point, the Business Interlink is invoked on the Application Server, and it issues an HTTP request out to the PScustomer Web site, which calculates the shipping costs and returns the information back to the PeopleSoft application.

This diagram explains the processing flow of this example in more detail.



Calculate Cost Business Interlink Processing Flow

In the PeopleTools Application Designer, you can view the Business Interlink Definition for the Quick Cost application. A Business Interlink consists of a set of inputs and outputs.

Example of Creating a Calculate XML Design-Time Plug-in

Following is a simplified XML design-time plug-in example for a transaction named Calculate Cost, with inputs of From, To, and Package_Info, and outputs of Service_Rate.

```
<?xml version="1.0" ?>
<interface_driver>

   <general_info>
      <description>PSCustomer services</description>
      <version>1</version>
      <comments>PSCustomer plug-in</comments>
      <image>PSCustomer.bmp</image>
   </general_info>

   <config_parameters>
      <URL>file://PSCustomer.dll</URL>
   </config_parameters>

   <trans_catalog>
      <category name="PSCustomer transactions">
         <transaction name="Calculate Cost">
            <input_list>
               <input name="From"
                  type="string"
                  required="true"/>
               <input name="To"
                  type="string"
                  required="true"/>
               <input name="Package_Info"
                  type="string"
                  required="true"/>
            </input_list>
            <output_list>
               <output name="Service_Rate"
                  type="string" />
            </output_list>
         </transaction>
      </category>
   </trans_catalog>
</interface_driver>
```

Common Business Interlink Terms

Application Designer: The integrated development environment used to develop PeopleSoft applications.

Basic Type: A data type such as an integer or string.

Business Interlink Definition: A definition encapsulating an external Transaction or Query and providing a set of generically typed input/outputs that can be assigned to PeopleCode variable or Record Fields at runtime. A Business Interlink Definition is added to the Application Designer's objects at the same level as Fields, Records, Pages, etc.

XML Design-Time Plug-in: An XML file that, when coded for an external system, encapsulate that external system and provide a catalog of Transactions, Classes and Criteria specific and meaningful to that external system. This functionality can also be written using a set of C++, Visual Basic, or other high-level language methods.

Business Interlink Framework: The framework for integrating any external system with PeopleTools application objects. It is composed of the following components: 1) An External System, 2) Generic definitions for a Transaction/Query command interfaces, 4) Business Interlink Definitions, 4) Business Interlink Plug-in.

Business Interlink Object: an instantiation based on a Business Interlink Definition. Actual data can be added to the inputs of the Business Interlink Objects once the appropriate bindings are provided. The Business Interlink Object can be executed to perform the external service. Once a Business Interlink Object is executed, the user of that object can retrieve the outputs of the external service. The Business Interlink Objects use buffers to receive input and send output. When a Business Interlink Object is executed, the transaction/query/class associated to the Business Interlink Object will be executed once per each row of the input buffers corresponding to the input Records. If there is only one row, after appropriate substitution by the driver, it is executed only once.

Runtime Plug-in: A set of C++, Visual Basic, or other high-level language methods that, when coded for an external system, encapsulate that external system and provide the execution methods to match the Business Interlink Design-Time Plug-in.

Catalog: the list of transactions, classes, and queries used to interface to the external system. Integration users are presented with this list when they pick the type of Business Interlink Plug-in they are going to use. There are four types of catalogs:

- Transaction catalog: lists transactions used to interface to the external system. See Transaction.
- Class catalog: lists classes used to interface to an external system. A class contains data members of basic types and/or objects that are typed after another class. A Class can also contain lists of basic types or objects.
- Operator catalog: lists query operators used to query the external system. These query operators are used to query the classes in the class catalog.
- Configuration parameter catalog: Used to configure an external system with PeopleSoft. For example, it might set up configuration and communication parameters for an external server.

External System: Any system that is not directly compiled with the PeopleTools servers.

PeopleCode: PeopleSoft's proprietary language; it is executed by the PeopleSoft Application Processor. PeopleCode generates results based upon specific actions, based upon existing data or the actions of a user. Business Interlink Objects are executed by calling the execute() method from PeopleCode. This makes external services available to all PeopleSoft applications wherever PeopleCode can be executed.

PeopleCode Event: An action that an end-user takes upon an object, usually a Record Field, that is referenced within a PeopleSoft page.

Query: a set of data members that are selected from a Class catalog (provided by the Business Interlink Plug-in) as well as a generic form of Criteria. The criteria are composed of <left-hand-side> <Relational Operator> <right-hand-side> statements that can be concatenated using a set of logical operators. All operators and class catalogs are dynamically provided through the Business Interlink Plug-in.

Transaction: a named command with optional named and typed inputs and outputs. The associated external system or the Business Interlink Plug-in understands this command. The types of inputs and outputs are based on a set of generic types.

Shape: For a transaction, the set of inputs and outputs for that transaction. For a class, the data members of that class.

CHAPTER 2

Designing Business Interlink Transactions and Classes

This chapter provides an overview of Business Interlink transactions and classes and discusses how to:

- List your configuration parameters
- List your Business Classes
- List your Business Transactions

Understanding Business Transactions and Classes

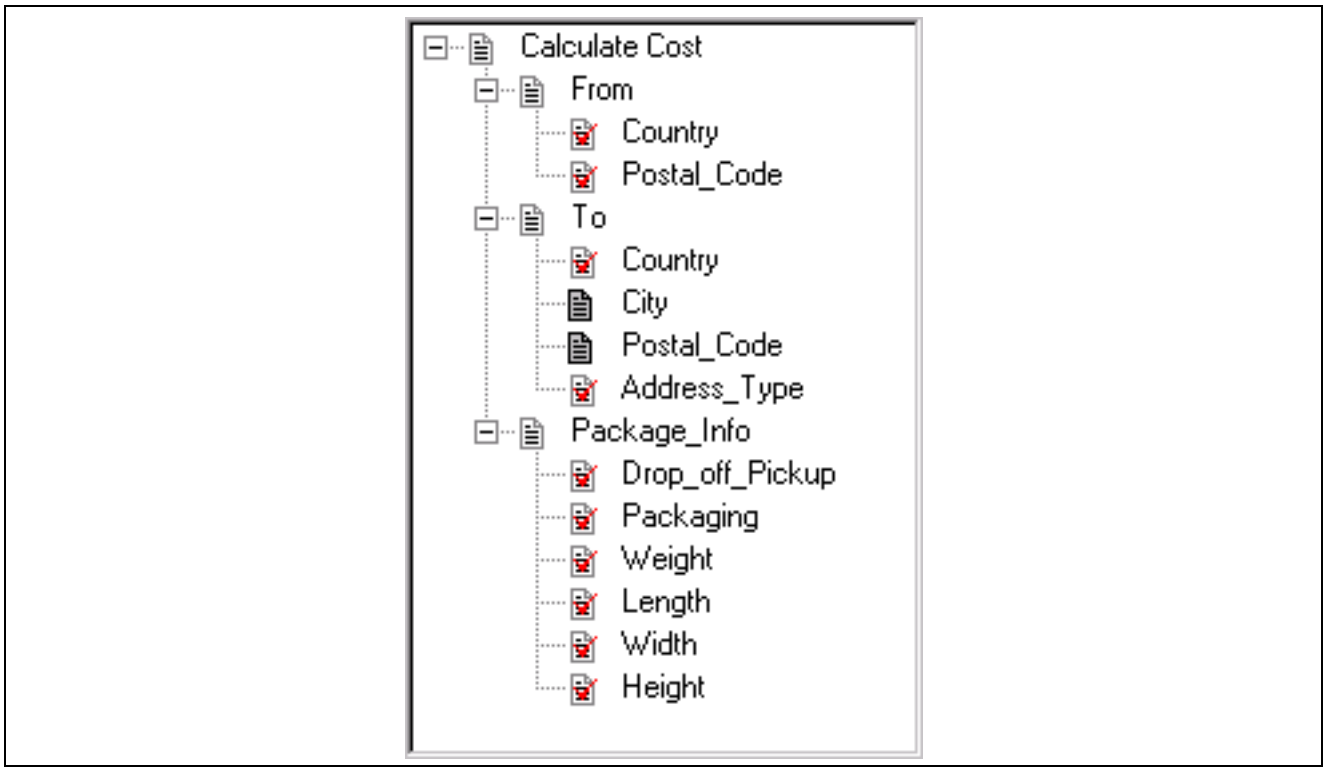
When you use Business Interlinks with your system, you must design a set of Business Interlink Transactions and/or Classes. A *Business Transaction*, or transaction, is a named command with optional named and typed inputs and outputs. A *Business Class*, or class, is a definition of a data structure. If you map the data structures in your external system to Business Classes, you can use the Business Interlink Framework to perform queries, adds, deletes, and updates on your data.

You should be familiar with the PeopleSoft components, pages, or batch programs that will be calling your Business Interlink Plug-in, and the input and output data they will need, in order to determine all the Business Transactions and Business Classes needed to be supplied by the plug-in.

You will determine the following:

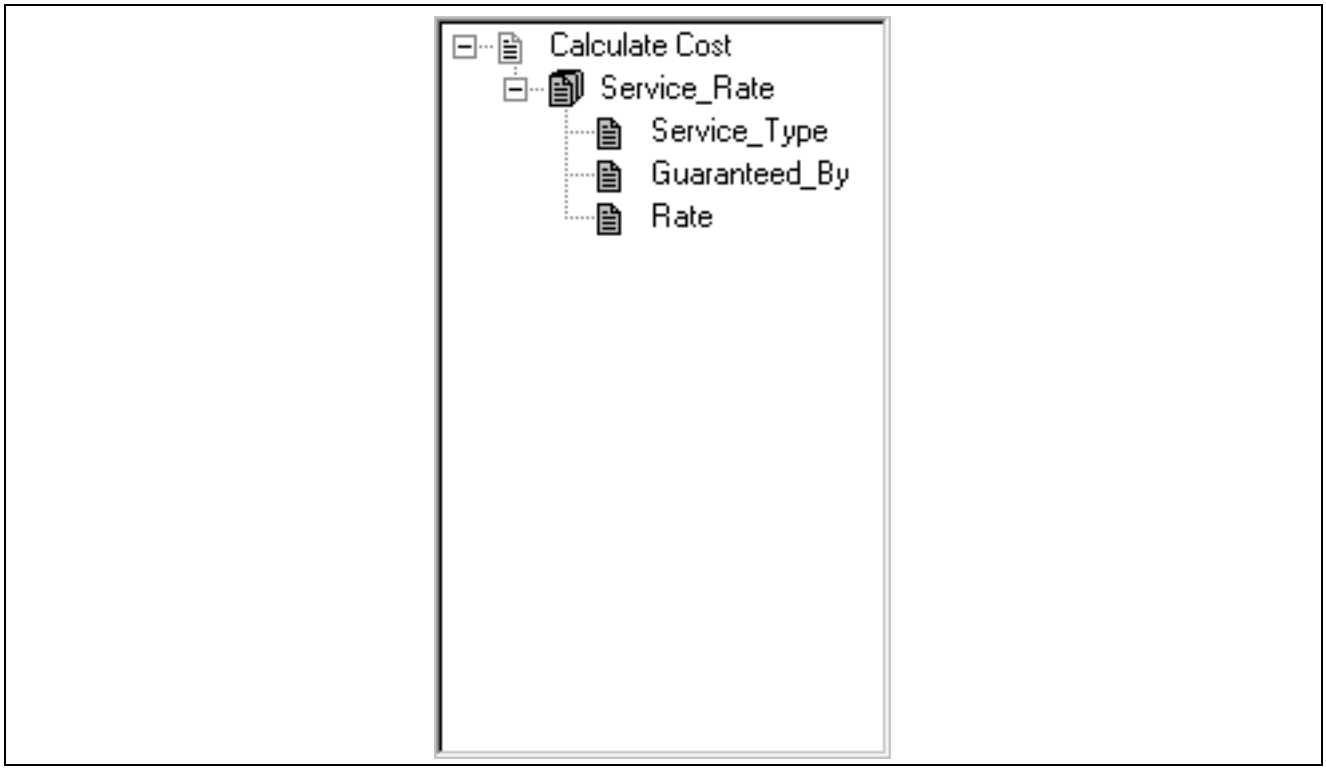
- If you need transactions, classes, or both.
- The list of transaction names, and the inputs and outputs for the transactions.
- The list of class names, and the data members for the classes.
- If you need a list of Configuration Parameters: global parameters that you can easily access within the execution methods of your Business Interlink Plug-in.

Here is a diagram of the Calculate Cost(Domestic) transaction inputs resulting from the designs in this chapter.



Design of Calculate Cost Transaction: Inputs

Here is a diagram of the Calculate Cost(Domestic) transaction outputs resulting from the designs in this chapter.



Design of Calculate Cost Transaction: Outputs

Listing Configuration Parameters

You might need to supply Configuration Parameters. Configuration parameters are used as global parameters that you can easily access within the execution methods of your Business Interlink Plug-in. For example, an email system would often need information allowing access to an email server, so its configuration parameters could be a mail server type (STMP or Pop3), a logon name, a password, and an email address.

There is one configuration parameter that is standard: the URL configuration parameter specifies where your Business Interlink runtime plug-in is located.

If you use configuration parameters other that the URL configuration parameter, determine the following information for each configuration parameter:

- The name.
- The data type.

Example Design for Configuration Parameters

The Freight Carrier example only uses the URL configuration parameter.

Listing Your Business Classes

You can use classes when you want to organize the input and output parameters for your transactions. For example, in a transaction for Freight Carrier that calculates the cost of sending a package, you may use classes to organize the package information: its origin, destination, type, and account information.

You can also use classes to map to data in your external system so that you can perform one or more of the following methods on that data: query, add, update, and delete. For example, if you have a database with Customer records, you may want to be able to use all four methods: query for customer information, update a customer's information, add a customer to the database, and delete a customer from the database. Or, if your external system provides access to its application objects through a run-time object engine, or a relational storage, you want to define a set of classes, each with their data members, some of which can point to other classes.

If you use Business Classes, determine the following information for each class:

- The class name.
- The names and types of the data members for that class.

Design for Freight Carrier Classes

The Freight Carrier classes are used to help organize the input and output parameters for the Freight Carrier transactions.

Origin, the class containing information about the origin location of a Freight Carrier package, can be designed as follows.

Data Member Name	Data Type
Country	Enumeration. The enumerated types are: United States, Puerto Rico
Postal_Code	String

Destination, the class containing information about the destination location of a Freight Carrier package, can be designed as follows.

Data Member Name	Data Type
Country	Enumeration. The enumerated types are: Argentina,Australia,Aruba,Brazil,Bosnia,Canada,China,Costa Rica,Finland,France,Germany,Greece,Hong Kong,Iran,Italy,Israel,Japan,Korea,Mexico,Russia,Spain,Taiwan,United Kingdom,United States,Zambia
City	String
Postal_Code	String
Address_Type	Enumeration. The enumerated types are: Commercial, Residential

Package_Information, the class containing information about a Freight Carrier package, can be designed as follows.

Data Member Name	Data Type
Drop_off_Pickup	Enumeration. The enumerated types are: Regular Daily Pickup,On Call Air,One Time Pickup,Letter Center,Customer Counter
Packaging	Enumeration. The enumerated types are: Your Packaging, Letter Envelop, Express Box, Worldwide 25KG Box, Worldwide 10KG Box
Weight	Integer
Length	Integer

Data Member Name	Data Type
Width	Integer
Height	Integer

Service Rate, the class containing information about the Freight Carrier service rates, can be designed as follows.

Data Member Name	Data Type
Service_Type	String
Guarnteed_By	String
Rate	String

Listing Your Business Transactions

If the purpose of your external system is to perform functions (other than update, add, delete, query of structured data), you will likely use Business Transactions.

If your external system already provide a set of transactions with well-defined interfaces, you are almost done with this step. You will only need to organize a list of transaction names, each with a set of (optionally hierarchical) named and typed inputs and outputs. On the other hand, if there are no well-defined transactions for interfacing, you may need to define a set of logical transactions that wrap around the lower level functions you are trying to expose.

If you will use Business Transactions, decide on the following information for each transaction:

- The transaction name
- What the transaction does
- What input and output parameters the transaction uses, and their data types

Design For Freight Carrier Transactions

Calculate Cost, the transaction that calculates the cost of sending a Freight Carrier package, can be designed as follows.

Input Parameter Name	Data Type
From	Origin class

Input Parameter Name	Data Type
To	Destination class
Package_Info	Package_Information class

Output Parameter Name	Data Type
Service_Rate	Service Rate class

Time-in-Transit, the transaction that calculates the time taken to ship an item, can be designed as follows.

Input Parameter Name	Data Type
From	Origin class
To	Destination class

Output Parameter Name	Data Type
From	Origin class
To	Destination class
Transit_Time	String

Tracking, the transaction that tracks a shipped item, can be designed as follows.

Input Parameter Name	Data Type
Tracking_Number	String

Output Parameter Name	Data Type
Tracking_Number	String
Status	String
Date	Date

CHAPTER 3

Writing an XML Design-Time Plug-In

This chapter provides an overview of XML Design-Time Plug-ins and discusses how to:

- Use the XML template file to create your XML Design-Time Plug-In.
- Write the tags in an XML Design-Time Plug-In.
- Use data types.

Understanding XML Design-Time Plug-ins

Once you have designed your transactions and classes by naming the transactions/classes and their parameters, you introduce your transactions and classes to the Business Interlink framework. Before you can manifest your transactions as Business Interlink Definitions, you introduce them to the PeopleTools design-time environment. This is accomplished by supplying a an XML design-time plug-in. This plug-in is used to define and save Business Interlink Definitions. It can also be used to test the Business Interlink Definition by simulating the execution of the corresponding Business Interlink Objects, if you supply default output values.

After you have decided what services that your system should support, check the services to see if their catalogs are static. A static catalog means that the transactions and classes are predefined; the input/output parameters for a transaction, or the data members for a class, are not changeable in data type or number of parameters/members.

When you use static catalogs, you can supply an XML design-time plug-in. The XML design-time plug-in uses tags for expressing all the information required for defining the interface to your transactions.

Note. For comparison, a dynamic catalog would mean that the transactions and classes are changeable; a plug-in to a database would allow the members of a class to be changed, also a new class can be added or deleted. The *PeopleSoft Business Interlink Runtime Plug-in Programming Guide* contains a chapter that describes how to write design time functionality when you do not write an XML design-time plug-in.

Using The XML Template File To Create Your XML Design-Time Plug-In

When you write your XML design-time plug-in, you can copy the file template.xml, and edit that copy to create your own XML design-time plug-in. The template.xml file contains most of the structure that you need for your XML design-time plug-in. Copy the template.xml file from the following directory, and then use a text editor to create your XML design-time plug-in.

The template is stored at the following location:

```
<PS_HOME>\SDK\src\C++\TEMPLATES
```

For examples of completed XML design-time plug-ins, search the following directories for file names ending with .xml:

```
<PS_HOME>\SDK\src\C++\Samples
```

This is an example of an XML design-time plug-in for a Freight Carrier plug-in. It contains four classes (Origin, Destination, Package_Information, Service_Rate) and three transactions (Calculate Cost, Tracking, Time-in-Transit). Note that the classes are used as inputs and outputs for the transactions.

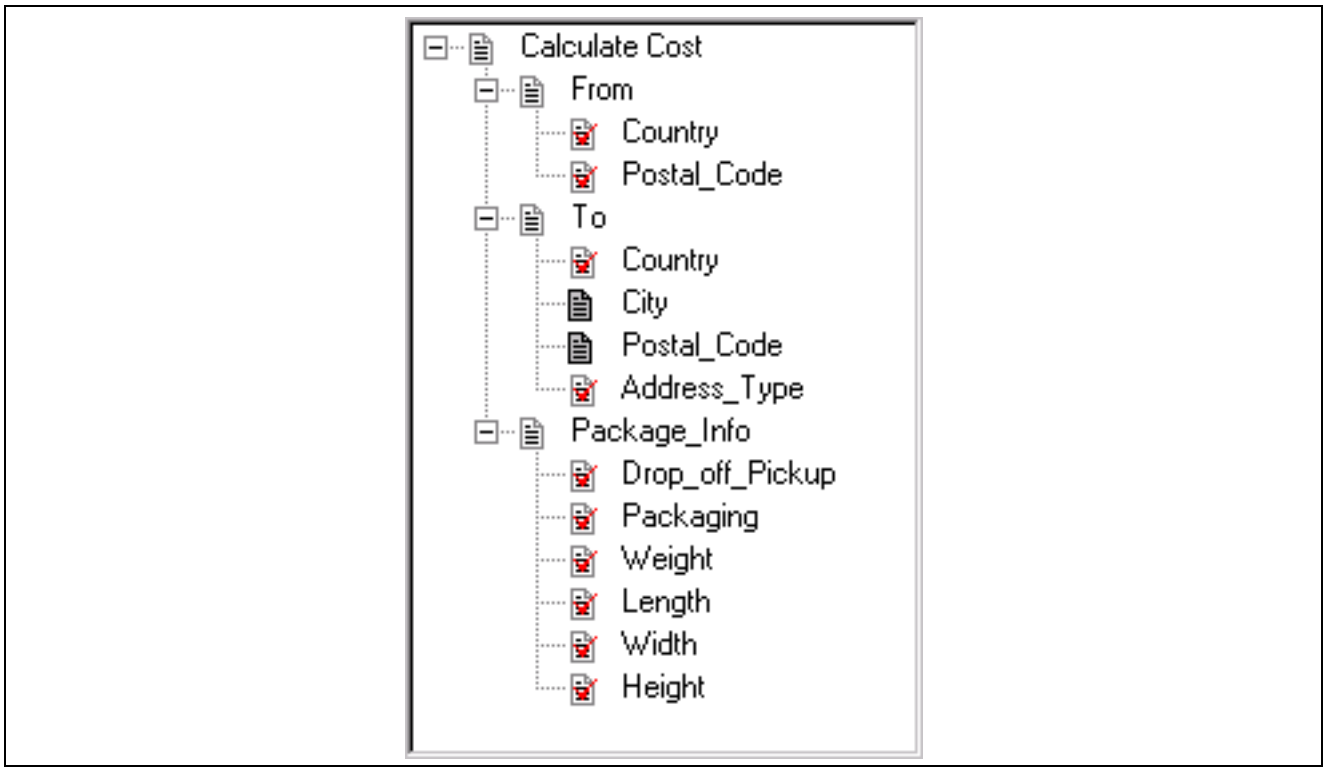
```
<?xml version="1.0" ?>
<interface_driver>

   <general_info>
      <description>PSCustomer services</description>
      <version>1</version>
      <comments>PSCustomer plug-in</comments>
      <image>PSCustomer.bmp</image>
   </general_info>


   <driver_settings>
      <option type="static_catalog" supported="true"/>
      <option type="transaction" supported="true"/>
      <option type="input_class_expandable" supported="true"/>

      <relational_op>
      </relational_op>

      <logical_op>
      </logical_op>
   </driver_settings>


   <config_parameters>
      <URL>file://PSCustomer.dll</URL>
   </config_parameters>


   <class_catalog>
      <category name="Internal Class">

         <class name="Origin">
            <member name="Country"
               type="enum(United States,Puerto Rico)"
               default="United States"
               required="true"/>
            <member name="Postal_Code"
               type="string"
               default="94588"
```

```
                required="true"/>
          </class>

          <class name="Destination">
             <member name="Country" type="enum(Argentina,Australia,Aruba,Brazil,?
Bosnia,Canada,China,Costa Rica,Finland,France,Germany,Greece,?
Hong Kong,Iran,Italy,Israel,Japan,Korea,Mexico,Russia,Spain,?
Taiwan,United Kingdom,United States,Zambia)"
                default="United States"
                required="true"/>
             <member name="City"
                type="string"
                default="New York"
                required="false"/>
             <member name="Postal_Code"
                type="string"
                default="10200"
                required="false"/>
             <member name="Address_Type"
                type="enum(Commercial,Residential)"
                default="Commercial"
                required="true"/>
          </class>

          <class name="Package_Information"

             <member name="Drop_off_Pickup" type="enum(Regular Daily Pickup,?
On Call Air,One Time Pickup,Letter Center,Customer Counter)"
                default="One_Time_Pickup" required="true"/>
             <member name="Packaging" type="enum(Your Packaging,?
Letter Envelop,Tube,Express Box,?
Worldwide 25KG Box,Worldwide 10KG Box )"
                default="Express_Box" required="true"/>
             <member name="Weight"
                type="int"
                default="1"
                required="true"/>
             <member name="Length"
                type="int"
                default="4"
                required="true"/>
             <member name="Width"
                type="int"
                default="4"
                required="true"/>
             <member name="Height"
                type="int"
                default="4"
                required="true"/>
          </class>
```

```
      <class name="Service Rate">
         <member name="Service_Type"
            type="string"
            default="Next Day Air Early AM"/>
         <member name="Guaranteed_By"
            type="string"
            default="8:00 AM Next Day"/>
         <member name="Rate"
            type="string"
            default="50.00"/>
      </class>
   </category>
</class_catalog>

<trans_catalog>
   <category name="PSCustomer transactions">

      <transaction name="Tracking">
         <input_list>
            <input name="Tracking_Number"
               type="string"
               required="true"/>
         </input_list>
         <output_list>
            <output name="Tracking_Number"
               type="string"
               default="1Z897X430397192061"/>
            <output name="Status"
               type="string"
               default="Delivered"/>
            <output name="Date"
               type="date"
               default="01/05/1998 16:12:00"/>
         </output_list>
      </transaction>

      <transaction name="Time-in-Transit">
         <input_list>
            <input name="From"
               type="string"
               required="true"/>
            <input name="To"
               type="string"
               required="true"/>
         </input_list>
         <output_list>
            <output name="From"
               type="string"
               default="Pleasanton"/>
```

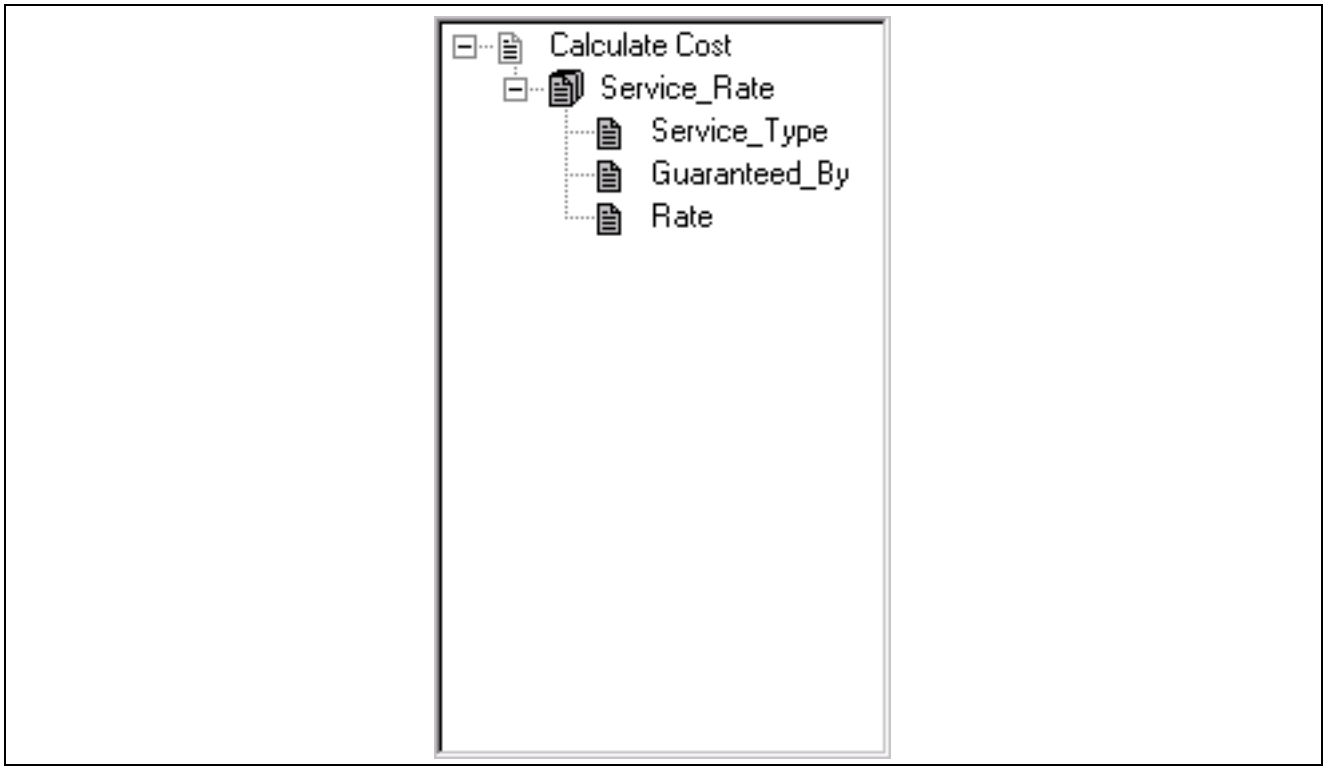
```
            <output name="To"
               type="string"
               default="San Jose"/>
            <output name="Transit_Time"
               type="string"
               default="1"/>
         </output_list>
      </transaction>

      <transaction name="Calculate Cost">
         <input_list>
            <input name="From"
               type="object"
               classname="Origin"/>
            <input name="To"
               type="object"
               classname="Destination"/>
            <input name="Package_Info"
               type="object"
               classname="Package_Information"/>
         </input_list>
         <output_list>
            <output name="Service_Rate"
               type="list_object"
               classname="Service Rate"/>
         </output_list>
      </transaction>

   </category>

</trans_catalog>
</interface_driver>
```

Here is a diagram of the Calculate Cost transaction inputs resulting from the above XML design-time plug-in.



Design of Calculate Cost Transaction Inputs

Here is a diagram of the Calculate Cost transaction outputs resulting from the above XML design-time plug-in.



Design of Calculate Cost Transaction Outputs

Writing The Tags in an XML Design-Time Plug-In

This section discusses how to:

- Write the main tag <interface driver>.
- List the supported Business Interlink types <driver_settings>.
- List the configuration parameters <config_parameters>.
- Write the class catalog <class_catalog>.
- Write the transaction catalog <trans_catalog>.

Writing the main tag <interface_driver>

The main tag for the Business Interlink XML design-time plug-in is **<interface_driver>**. It contains the **<general_info>**, **<driver_settings>**, **<config_parameters>**, **<class_catalog>**, and **<trans_catalog>** tags.

Writing the General Information for the Plug-in <general_info>

In the **<general_info>** tag, write a description of your plug-in and the version number for your plug-in, and optionally the last time the XML design-time plug-in was updates and any comments for this XML design-time plug-in.

Here is an example of how the **<general_info>** tag is coded for a Freight Carrier plug-in. The description is “PSCustomer services”, and the version number is 1.

```
<general_info>
   <description>PSCustomer services</description>
   <version>1</version>
   <comments>PSCustomer plug-in</comments>
   <image>PSCustomer.bmp</image>
</general_info>
```

This section discusses how to:

- Write the plug-in description <description>.
- Write the plug-in version number <version>.
- Name a graphic file for the Business Interlink property dialog box <image>.

Write the Plug-in Description <description>

In the **<description>** tag, write a short description of your XML design-time plug-in. The **<description>** tag must be placed within a **<general_info>** tag.

Here is an example of the **<description>** tag. The description is “PSCustomer services”.

```
<description>PSCustomer services</description>
```

Within the Application Designer, the description is displayed within the New Business Interlink page, and at the top of each Interlink page within the Application Designer.

Write the Plug-in Version Number <version>

In the **<version>** tag, write the version number for your XML Design-Time Plug-In. The **<version>** tag must be placed within a **<general_info>** tag. The version number is used by the Business Interlink Framework to determine the version of your plug-in. If there is more than one version of your plug-in, this number is needed to identify the plug-in.

Here is an example of the **<version>** tag. The version number is 1.

```
<version>1</version>
```

The version number allows you to write future versions of your XML design-time plug-in, and to uniquely identify each version with a number. The version number for the XML design-time plug-in will be compared to the version number for the C++ Business Interlink runtime plug-in that you also write; there can be more than one set of C++ Business Interlink runtime plug-ins per XML design-time plug-in.

See *PeopleSoft Business Interlink Runtime Plug-in Programming Guide*, "Writing the Version Methods for a Business Interlink Runtime Plug-In," Writing IsVersionCompatible for Your Business Interlink Plug-in Class.

Name a Graphic File for the Business Interlink Property Dialog Box <image>

In the **<image>** tag, name a graphic file that you want to use with this Business Interlink. The graphic appears in the Business Interlink property box in the Application Designer.

This graphic file must be stored in the same location as where the XML design-time plug-in is deployed.

Here is an example of how the **<image>** tag could be coded.

```
<image>PSCustomer.bmp</image>
```

Listing The Supported Business Interlink Types <driver_settings>

In the **<driver_settings>** tag, list the Business Interlink types that your plug-in will support, and list any relational operators and logical operators if your plug-in supports them.

Here is an example of how the **<driver_settings>** tag could be coded. The supported Business Interlink type is transaction.

```
<driver_settings>
   <option type="static_catalog" supported="true"/>
   <option type="transaction" supported="true"/>
   <option type="input_class_expandable" supported="true"/>

   <relational_op>
   </relational_op>

   <logical_op>
   </logical_op>
</driver_settings>
```

This section discusses how to:

- Write the Business Interlink type <option>.
- List the query or update relational operators <relational_op>.
- List the query or update logical operators <logical_op>.
- List an operator <operator>.

Write A Business Interlink Type <option>

Write an **<option>** tag for each Business Interlink type that your plug-in supports. The **<option>** tag must be placed within a **<driver_settings>** tag.

Here are examples of **<option>** tags that support the Business Interlink types of transaction, static catalog (static catalog being required in the XML design-time plug-in), and input_class_expandable, so that inputs can be classes.

```
<option type="static_catalog" supported="true"/>
<option type="transaction" supported="true"/>
<option type="input_class_expandable" supported="true"/>
```

The valid values for *type* are:

type	Definition
static_catalog	When you write an XML design-time plug-in, you have static catalogs. You must include the following **<option>** tag with every XML design-time plug-in for a Business Interlink Plug-in: `<option type="static_catalog" supported="true"/>`
transaction	Your plug-in supports transaction Business Interlinks.
object_query	Your plug-in supports query Business Interlinks.
object_add	Your plug-in supports add Business Interlinks.
object_update	Your plug-in supports update Business Interlinks.
object_delete	Your plug-in supports delete Business Interlinks.
order_by_desc	If you use the object_query type, you can use order_by_desc for descending order. This type allows a query Business Interlink Object to list query results in descending order. The type order_by_desc is only used with query Business Interlink types.

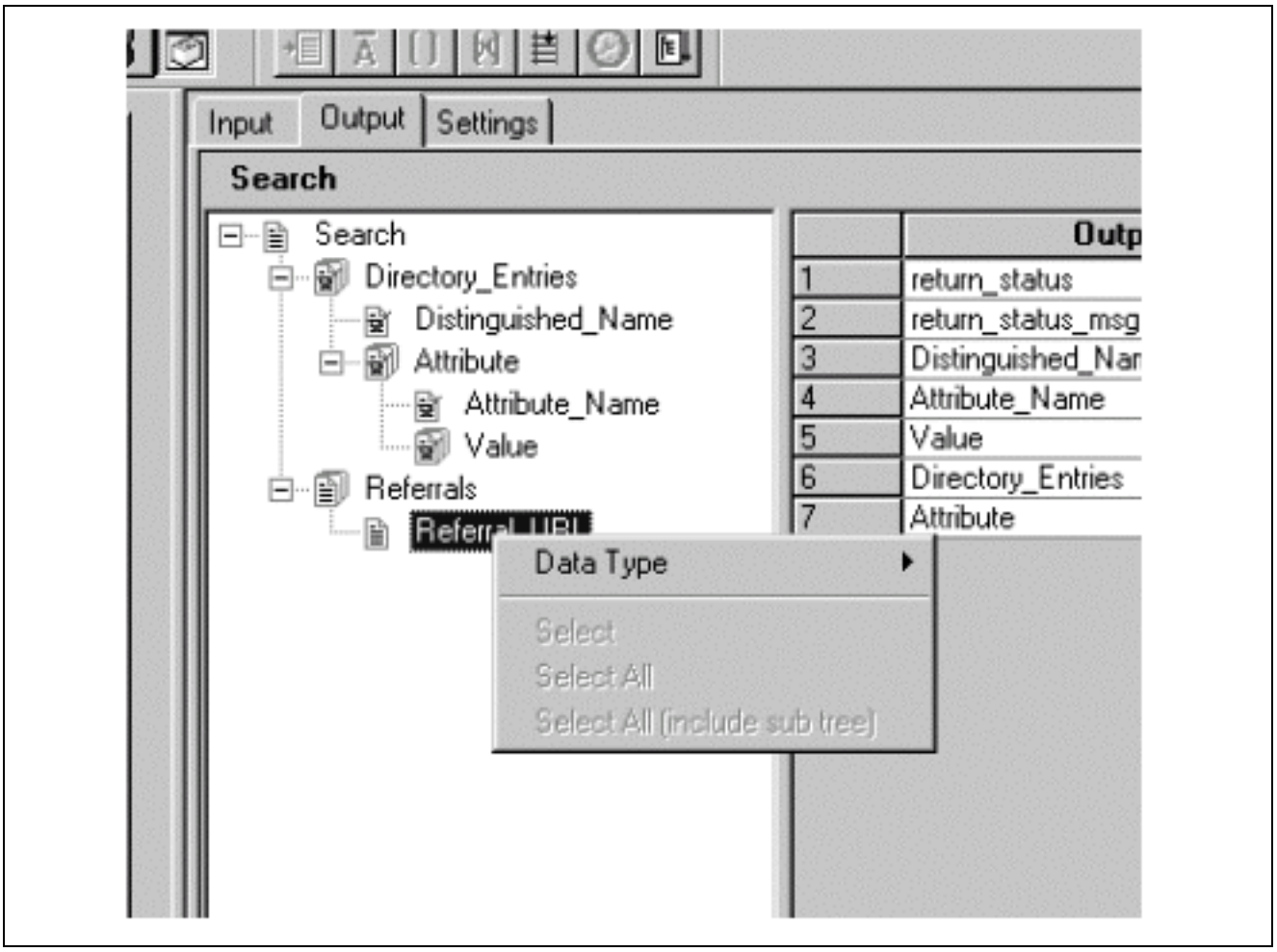
type	Definition
order_by_asc	If you use the object_query type, you can use order_by_asc for ascending order. This type allows a query Business Interlink Object to list query results in ascending order. The type order_by_asc is only used with query Business Interlink types.
input_class_expandable	You must include the following **<option>** tag with every XML design-time plug-in that contains, within a **<transaction>** tag, an **<input>** tag that is of type object. The object type allows inputs to be classes by pointing to a **<class>** tag. `<option type="input_class_expandable" ? supported="true"/>`
output_class_expandable	You must include the following **<option>** tag with every XML design-time plug-in that contains, within a **<transaction>** tag, an **<output>** tag that is of type object. The object type allows outputs to be classes by pointing to a **<class>** tag. `<option type="output_class_expandable"? supported="true"/>`
hierarchical_model	To support Business Interlink Object Lists, set the hierarchical_model object type.

To support Business Interlink Object Lists, you should support the hierarchical_model object type in the <driver_settings> section of your XML Design-Time Plug-In:

```
<option type="hierarchical_model" supported="true"/>
```

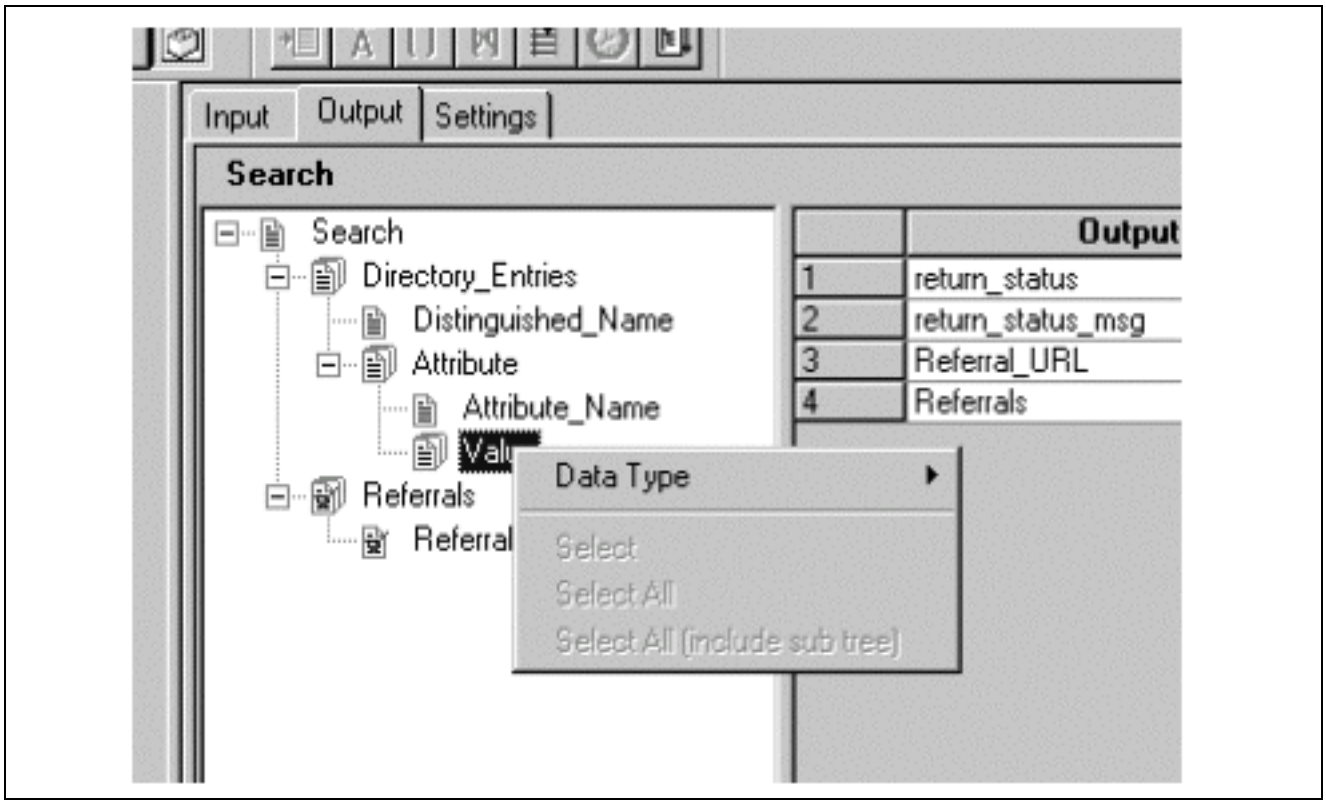
Within the Business Interlink Definition created from this XML Design-Time Plug-In, this setting allows you to select the children of two object lists that are at the same level of a Business Interlink Definition hierarchy.

For example, the following Business Interlink Definition contains object lists at the same level.



Referrals Object List Not Selected

Once a child of Directory_Entries is selected, Referral_URL can not be selected unless the XML Design-Time Plug-in supports the hierarchical_model object type.



Directory_Entries Not Selected

Likewise, if Referral_URL is selected first, Directory_Entries can not be selected unless the XML Design-Time Plug-in supports the hierarchical_model object type.

List The Query or Update Relational Operators <relational_op>

If you use object_query or object_update types, then you can write relational operators by using the **<relational_op>** tags. Use the **<relational_op>** tag only if you have relational operators. The **<relational_op>** tag must be placed within a **<driver_settings>** tag. The **<relational_op>** tag is only valid with the Interlink types of object query or object update.

The following example shows how you would set relational operators for + and !=.

```
<relational_op>
   <operator name="=" />
   <operator name="!=" />
</relational_op>
```

List The Query or Update Logical Operators <logical_op>

If you use object_query or object_update types, then you can write logical operators by using the **<logical_op>** tags. In these tags, include the name of the operator. Use the **<logical_op>** tag only if you have logical operators. The **<logical_op>** tag must be placed within a **<driver_settings>** tag. The **<logical_op>** tag is only valid with the Interlink types of object query or object update.

The following example shows how you would set logical operators for OR and AND.

```
<logical_op>
   <operator name="OR" />
```

```
   <operator name="AND" />
</logical_op>
```

List an operator <operator>

Write an **<operator>** tag for each relational operator and logical operator that your plug-in supports. The **<operator>** tag must be placed within a **<relational_op>** tag or a **<logical_op>** tag.

The following example shows how you would set a relational operator for !=.

```
<operator name="!=" />
```

Listing The Configuration Parameters <config_parameters>

In the **<config_parameters>** tag, write the configuration parameters for your plug-in. Configuration parameters contain data that can be used in a variety of ways, depending upon your system. A common use is for the configuration parameters to be a set of global variables that you would use when you write the execution code for your plug-in. Another common use for configuration parameters is data that helps connect your external service to PeopleSoft.

Here is an example of how the **<config_parameters>** tag is coded for a Freight Carrier plug-in.

```
<config_parameters>
   <URL>file://PSCustomer.dll</URL>
</config_parameters>
```

Note. Whenever you specify a UNIX directory name as a configuration parameter, you must end the directory name with a backslash /.

Here is an example of how the **<config_parameters>** tag could be coded for an email plug-in. This example shows how two configuration parameters are coded. The configuration parameter SMTP_MAIL_SERVER is of type string, has a default value of st-sun13.peoplesoft.com, and is a required configuration parameter. The configuration parameter LOGIN_NAME is of type string, has a default value of certora, and is a required configuration parameter.

```
<config_parameters>
   <parameter name="SMTP_MAIL_SERVER"
      type="string"
      default="st-sun13.peoplesoft.com"
      required="true"/>
   <parameter name="LOGIN_NAME"
      type="string"
      default="certora"
      required="true"/>
</config_parameters>
<URL>file//:myplugin.dll</URL>
```

This section discusses how to:

- Write a configuration parameter <parameter>.
- Error check BI documents <BiDocValidate>.
- Specify the Business Interlink runtime plug-in file location <URL>.

Write A Configuration Parameter <parameter>

Write a **<parameter>** tag for each configuration parameter that your plug-in uses. The **<parameter>** tag must be placed within a **<config_parameters>** tag. A **<parameter>** tag contains a name and type, and an optional default value.

Here is a **<parameter>** tag for a required configuration parameter that is a character string named LOGIN NAME.

```
<parameter name="LOGIN_NAME"
   type="string"
   default="certora"
   required="true"/>
```

See Chapter 3, "Writing an XML Design-Time Plug-In," Using Data Types, page 35.

<BiDocValidate> Error Check BI Documents

When set to ON, the PeopleCode program that runs a BIDocs object is checked to see if it exists before adding/getting values from it. A BIDocs object is the most common method used by PeopleCode to access the input and output data for Business Interlinks.

For example, if the BIDoc object "Rate" does not exist, and BIDocValidate is set to ON, then the following line of PeopleCode will cause an error:

```
&ret = &biDocs.GetValue("Rate", &RATE);
```

The default value is ON.

<URL> Specify Business Interlink Runtime Plug-in File Location

Write a **<URL>** tag if you want to specify where the Business Interlink runtime plug-in is located. Here is an example of how the **<URL>** tag is coded for a Freight Carrier plug-in, telling that the runtime plug-in is a file named PSCustomer.dll.

```
<URL>file://PSCustomer.dll</URL>
```

If you supply a **<URL>** tag, and the plug-in is not found in the location specified by the **<URL>** tag, an error message will occur when a user attempts to use this plug-in. The plug-in can be specified as follows:

```
<URL>file://runtime_plug-in_name.dll</URL>
```

runtime_plug-in_name the name of the Business Interlink runtime plug-in.

This will find a runtime plug-in named *runtime_plug-in_name.dll* in the InterfaceDrivers directory. The InterfaceDrivers directory is located in the PeopleTools installation.

```
<URL>file://path/runtime_plug-in_name.dll</URL>
```

path the path to your Business Interlink runtime plug-in file.

runtime_plug-in_name the name of your runtime plug-in file.

This will find the runtime plug-in named *runtime_plug-in_name.dll* at the specified path on the computer where PeopleTools is installed.

```
<URL>http://webserver/servletname</URL>
```

webserver the URL where the Business Interlink installation is located. (Windows NT only.)

servletpath the path to a servlet on the Business Interlink installation.

servletname varies depending upon the Business Interlink installation for the web server. For Microsoft IIS, the default name is:

```
BusInterlink.asp
```

For Apache Web Server, the default name and path is:

```
Servlets\BusInterlinkServlet
```

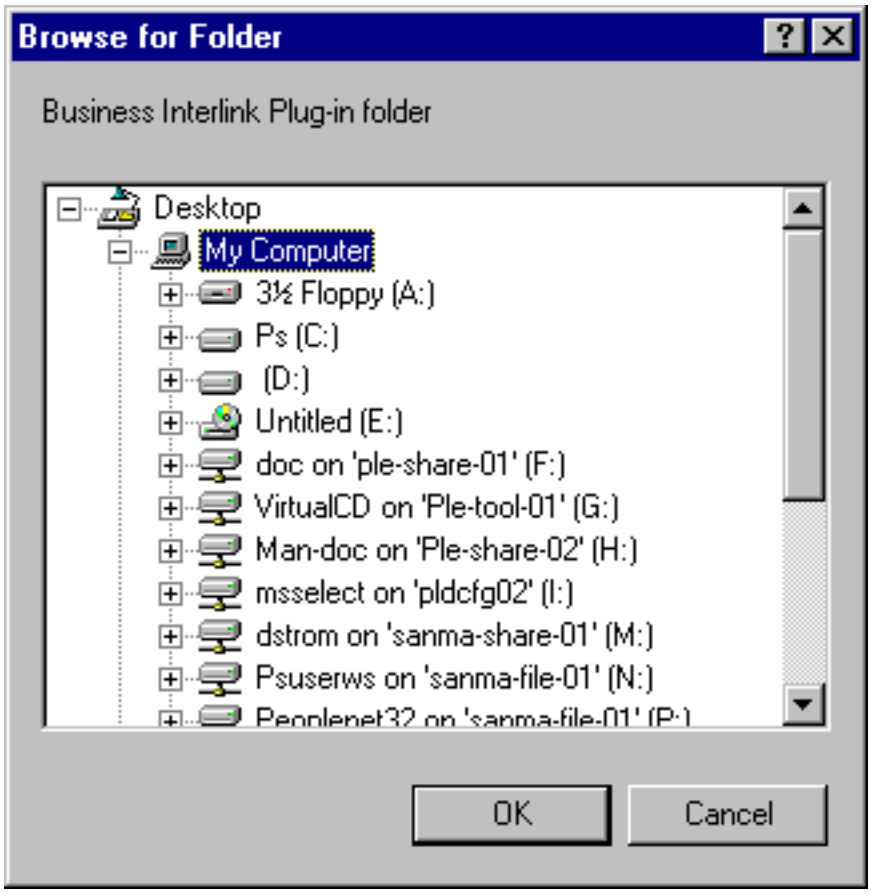
For *servletname* to work, the runtime plug-in must use the same name as the name of the XML design-time plug-in. For example, if runtime is fcarrier.dll, the design-time must be fcarrier.xml.

The method that the Business Interlink Framework uses to find your runtime plug-in DLL file is:

1. It looks in the location specified by the **<URL>** tag.
2. If there is no **<URL>** tag, it looks in the location specified by the Browse For Folder page, which is accessed by pressing the Location button in the New Business Interlink page.
3. If the search location button is not used, it looks in the InterfaceDrivers directory:

   ```
   PeopleTools_directory\ptdvl\bin\client\win86\InterfaceDrivers
   ```

   where *PeopleTools_directory* is the directory where PeopleTools is installed.



Browse For Folder Page

See Also

PeopleSoft Business Interlink Runtime Plug-in Programming Guide

Writing the Class Catalog <class_catalog>

In the **<class_catalog>** tag, write the class catalog that you want to use with your plug-in. The class catalog provides the categories for the classes, the names of the classes, the names of the data members for each class, the data type of each data member, and optional default values for the data members.

```
<class_catalog>
   <category name="the category name">

   <!-- Write the class tags here (code not shown) -->

   </category>
</class_catalog>
```

This section discusses how to:

- Write the categories for the classes.
- Write a class tag <class>.
- Write a data member for a class <member>.

<category> Write The Categories For The Classes

You organize the classes into categories with the **<category>** tag. The <*category*> tag must be placed within a **<category>** tag or a **<transaction_catalog>** tag.

For classes, you can write the **<category>** tag in two ways: named and unnamed.

Put classes into a **<category>** tag when you want to use the class as a transaction input or output parameter or a class data member, and you do not use that class with the Business Interlink types object_query, object_add, object_update, and object_delete.

Put classes into a **<category>** tag when you define within the **<driver_settings>** tag one of the Business Interlink types that operate on classes: object_query, object_add, object_update, object_delete. Business Interlink Objects of those types can then operate on those classes. When a user, within the Application Designer, uses the Business Interlink Search page to search for the class from which to create that Business Interlink Definition, the user can select a category to list the classes in that category, which were listed in the **<category>** tag in the XML design-time plug-in.

```
<category name="the category name">
   <!-- Write the class tags here (code not shown) -->

</category>
```

You can put classes into an unnamed **<category>** tag when you want to use the class only as a transaction input or output parameter or a class data member.

```
<category>

   <!-- Write the class tags here (code not shown) -->

</category>
```

Note. When you place a class into an unnamed **<category>** tag, it will not appear in the Business Interlink Search page in the PeopleSoft Application Designer.

<class> Write A Class

Write a **<class>** tag for each class that your plug-in will use. The **<class>** tag must be placed within a **<category>** tag. Classes are data structures, which consist of members. Members can be basic types such as integers and strings, members can be lists of basic types, and members can be other classes. Classes are used in two ways:

- Application Developers create Business Interlink Definitions/Objects that will perform queries, adds, updates, or deletes upon these classes.
- Classes can be used as members of other classes and as transaction input or output parameters.

Here is an example of how a **<class>** tag is coded for a Freight Carrier plug-in. It creates a class named “Origin”.

```
<class name="Origin">
   <member name="Country"
      type="enum(United States,Puerto Rico)"
      default="United States"
      required="true"/>
   <member name="Postal_Code"
```

```
      type="string"
      default="94588"
      required="true"/>
</class>
```

<member> Write Members For A Class

Write a **<member>** tag for every data member that a class uses. The **<member>** tag must be placed within a **<class>** tag. A **<member>** tag contains a name and type, and an optional default value. It can also contain a trim, which causes all leading and trailing spaces to be trimmed from the data member; any spaces within, such as in "value 1", are not trimmed.

Note. Whenever you specify a UNIX directory name as a data member, you must end the directory name with a backslash /.

Here is an example of how a **<member>** tag is coded for a Freight Carrier plug-in. It creates a data member named "OriginPostal_Code".

```
<member name="OriginPostal_Code"
   type="string"
   default="94588"
   required="false"/>
   trim="true"/>
```

When you use trim, and you are building XML tags, you might need to add a space when calling AddValue in your PeopleCode program to control the output. For example, for the following class:

```
<class name="EndDate">
 <member name="Date" type="date" required="false"/>
 <member name="CurrentFlag" type="string" required="false" trim="true"/>
 <member name="SummaryText" type="object" classname="SummaryText" required="false"/>
</class>
```

Within PeopleCode, you can add a space with the member that has the trim:

```
&date = "abc";
&EndDateDoc.AddNextDoc();
If &date <> "" Then
   &ret = &EndDateDoc.AddValue("Date", "2001-8-10");
Else
   &ret = &EndDateDoc.AddValue("CurrentFlag", " ");
End-If;
```

Without the space, you get the following tag:

```
<CurrentFlag/>
```

With the space, you get the following tag:

```
<CurrentFlag> <CurrentFlag/>
```

See Chapter 3, "Writing an XML Design-Time Plug-In," Using Data Types, page 35.

Writing the Transaction Catalog <trans_catalog>

In the **<trans_catalog>** tag, write the transaction catalog that you want to use with your plug-in. The transaction catalog provides the names of the transactions, the names of the input and output parameters for each transaction, the data type of each parameter, and optional default values for the input and output parameters.

```
<trans_catalog>
   <category name="the category name">

   <!-- Write the transaction tags here (code not shown) -->

   </category>
</trans_catalog>
```

This section discusses how to:

- Write the categories for the transactions <category>.
- Write a transaction tag <transaction>.
- List the input parameters <input_list>.
- Write an input parameter <input>.
- List the output parameters <output_list>.
- Write an output parameter <output>.

Write The Categories For The Transactions <category>

Organize the transactions into named **<category>** tags. When a user, within the Application Designer, uses the Business Interlink Search page to search for the transaction from which to create that Business Interlink Definition, the user can select a category to list the transactions in that category, which were listed in the **<category>** tag in the XML design-time plug-in. The **<category>** tag must be placed within a **<transaction_catalog>** tag or a **<class_catalog>** tag.

For example, the following XML code shows a **<category>** tag named PUS transactions.

```
<category name="the category name">

   <!-- Write the transaction tags here (code not shown) -->

</category>
```

<transaction> Write A Transaction

Write a **<transaction>** tag for each transaction that your plug-in will use. A transaction is a named command that can have inputs and outputs. The **<transaction>** tag must be placed within a **<category>** tag.

Here is an example of how a **<transaction>** tag is coded for a Freight Carrier plug-in. It creates a transaction named Calculate Cost.

```
<transaction name="Calculate Cost">
   <input_list>
      <input name="From"
         type="object"
```

```
            classname="Origin"/>
         <input name="To"
            type="object"
            classname="Destination"/>
         <input name="Package_Info"
            type="object"
            classname="Package_Information"/>
      </input_list>
      <output_list>
         <output name="Service_Rate"
            type="list_object"
            classname="Service Rate"/>
      </output_list>
   </transaction>
```

List the Input Parameters <input_list>

If the transaction has input parameters, list each one within an **<input_list>** tag. The **<input_list>** tag must be placed within a **<transaction>** tag.

Write an Input Parameter for a Transaction <input>

Write an **<input>** tag for every input parameter that a transaction uses. The **<input>** tag must be placed within an **<input_list>** tag. An **<input>** tag contains a name and type, and an optional default value.

Note. Whenever you specify a UNIX directory name as an input parameter, you must end the directory name with a backslash /.

Here is an example of how an **<input>** tag is coded for a Freight Carrier plug-in. It creates an input parameter named “From”, which in turn uses the **<class>** named Origin. The data members of Origin are used as input parameters for this transaction.

```
<input name="From"
   type="object"
   classname="Origin"/>
```

Here is an **<input>** tag for an input parameter that is a floating point variable named dollar.

```
<input name="dollar" type="float" default="1.0" required="true"/>
```

See Chapter 3, “Writing an XML Design-Time Plug-In,” Using Data Types, page 35.

List the Output Parameters <output_list>

If the transaction has output parameters, list each one within an **<output_list>** tag. The **<output_list>** tag must be placed within a **<transaction>** tag.

Write an Output Parameter for a Transaction <output>

Write an **<output>** tag for every output parameter that a transaction uses. The **<output>** tag must be placed within an **<output_list>** tag. An **<output>** tag contains a name and type, and an optional default value. It can also contain a trim, which causes all leading and trailing spaces to be trimmed from the data member; any spaces within, such as in “value 1”, are not trimmed.

Note. Whenever you specify a UNIX directory name as an output parameter, you must end the directory name with a backslash /.

Here is an example of how an **<output>** tag is coded for a Freight Carrier plug-in. It creates an output parameter named "Service_Rate", which in turn uses the **<class>** named Service_Rate. The data members of Service_Rate are used as output parameters for this transaction.

```
<output name="Service_Rate"
   type="object"
   classname="Service Rate"/>
```

Here is an **<output>** tag for an output parameter that is a date variable named overdue.

```
<output name="overdue" type="date" default="12/31/99" />
```

See Chapter 3, "Writing an XML Design-Time Plug-In," Using Data Types, page 35.

See Chapter 3, "Writing an XML Design-Time Plug-In," Writing the Class Catalog <class_catalog>, page 30.

Using Data Types

The types you can use for configuration parameters, input parameters, output parameters, and data members are:

Type	Definition and format
Int	An integer.
String	A character string.
Float	A floating point variable.
Boolean	A Boolean value of TRUE or FALSE.
Date	A date. The format is YYYY-MM-DD, where YYYY is the year, MM is the month, and DD is the day.
Time	A time. The format is HH:MM:SS, where HH is the hour, MM is the minutes, and SS is the seconds.
Datetime	A date and time. The format is YYYY-MM-DD HH:MM:SS, where YYYY is the year, MM is the month, DD is the day, HH is the hour, MM is the minutes, and SS is the seconds.

Type	Definition and format
Enum	An enumeration. The format is enum(*name1*, *name2*,…) where *name1*, *name2*, … are the comma delimited names for this enumeration. Following is an example of an enumeration named Address with the values of Commercial and Residential. `<member name="Address"? type="enum(Commercial, Residential)">`
Object	A **<class>** in the **<class_catalog>** tag. If you set the type as object, you must also provide the classname for that **<class>** Following is an example of an object as an input for a transaction; for this example, there must be a **<class>** named Origin also defined in the XML design-time plug-in. `<input name="From" type="object"? classname="Origin"/>`
Password	A password. This is a character string.
Lists of the above types (not used with configuration parameters)	Each of the above types except for password can be supplied as a list. The list types are list_integer, list_string, list_float, list_boolean, list_date, list_time, list_datetime, and list_object. List types can not be used with relational or logical operators. When you set default values for a list type, you will set it in the same way that you set defaults for the non-list types; you set only one default value in each list.

Escaping XML Restricted Characters

Anywhere you pass in text or data in the XML Design-Time Plug-In, all XML restricted characters (such as < and >) need to be escaped. For example, < must be %60, > must be%62, " must be %34.

CHAPTER 4

Deploying your XML Design-Time Plug-In

Once you have written your XML design-time plug-in, place it into the following directory:

PSHome\bin\client\winx86\interfacedrivers

Where *PSHome* is the directory where PeopleTools is installed.

Glossary of PeopleSoft Terms

absence entitlement	This element defines rules for granting paid time off for valid absences, such as sick time, vacation, and maternity leave. An absence entitlement element defines the entitlement amount, frequency, and entitlement period.
absence take	This element defines the conditions that must be met before a payee is entitled to take paid time off.
account	You use an account code to record and summarize financial transactions as expenditures, revenues, assets, or liabilities balances. The use of this delivered PeopleSoft ChartField is typically defined when you implement PeopleSoft General Ledger.
accounting class	In PeopleSoft Enterprise Performance Management, the accounting class defines how a resource is treated for generally accepted accounting practices. The Inventory class indicates whether a resource becomes part of a balance sheet account, such as inventory or fixed assets, while the Non-inventory class indicates that the resource is treated as an expense of the period during which it occurs.
accounting date	The accounting date indicates when a transaction is recognized, as opposed to the date the transaction actually occurred. The accounting date and transaction date can be the same. The accounting date determines the period in the general ledger to which the transaction is to be posted. You can only select an accounting date that falls within an open period in the ledger to which you are posting. The accounting date for an item is normally the invoice date.
accounting entry	A set of related debits and credits. An accounting entry is made up of multiple accounting lines. In most PeopleSoft applications, accounting entries are always balanced (debits equal credits). Accounting entries are created to record accruals, payments, payment cancellations, manual closures, project activities in the general ledger, and so forth, depending on the application.
accounting split	The accounting split method indicates how expenses are allocated or divided among one or more sets of accounting ChartFields.
accumulator	You use an accumulator to store cumulative values of defined items as they are processed. You can accumulate a single value over time or multiple values over time. For example, an accumulator could consist of all voluntary deductions, or all company deductions, enabling you to accumulate amounts. It allows total flexibility for time periods and values accumulated.
action reason	The reason an employee's job or employment information is updated. The action reason is entered in two parts: a personnel action, such as a promotion, termination, or change from one pay group to anotherand a reason for that action. Action reasons are used by PeopleSoft Human Resources, PeopleSoft Benefits Administration, PeopleSoft Stock Administration, and the COBRA Administration feature of the Base Benefits business process.
activity	In PeopleSoft Enterprise Learning Management, an instance of a catalog item delivery methodit may also be called a class. The activity defines such things as meeting times and locations, instructors, reserved equipment and materials, and detailed costs that are associated with the offering, enrollment limits and deadlines, and waitlisting capacities.
allocation rule	In PeopleSoft Enterprise Incentive Management, an expression within compensation plans that enables the system to assign transactions to nodes and participants. During transaction allocation, the allocation engine traverses the compensation structure

	from the current node to the root node, checking each node for plans that contain allocation rules.
alternate account	A feature in PeopleSoft General Ledger that enables you to create a statutory chart of accounts and enter statutory account transactions at the detail transaction level, as required for recording and reporting by some national governments.
application agent	An application agent is an online agent that is loaded into memory with a PeopleSoft page. It detects when a business rule has been triggered and determines the appropriate action.
asset class	An asset group used for reporting purposes. It can be used in conjunction with the asset category to refine asset classification.
attachment	In PeopleSoft Enterprise Learning Management, nonsystem-defined electronic material that supplements a learning resource, such as an equipment items user handbook or the site map of a large facility.
background process	In PeopleSoft, background processes are executed through process-specific COBOL programs and run outside the Windows environment.
benchmark job	In PeopleSoft Workforce Analytics, a benchmark job is a job code for which there is corresponding salary survey data from published, third-party sources.
branch	A tree node that rolls up to nodes above it in the hierarchy, as defined in PeopleSoft Tree Manager.
budgetary account only	An account used by the system only and not by users; this type of account does not accept transactions. You can only budget with this account. Formerly called system-maintained account.
budget check	In commitment control, the processing of source transactions against control budget ledgers, to see if they pass, fail, or pass with a warning.
budget control	In commitment control, budget control ensures that commitments and expenditures don't exceed budgets. It enables you to track transactions against corresponding budgets and terminate a document's cycle if the defined budget conditions are not met. For example, you can prevent a purchase order from being dispatched to a vendor if there are insufficient funds in the related budget to support it.
budget period	The interval of time (such as 12 months or 4 quarters) into which a period is divided for budgetary and reporting purposes. The ChartField allows maximum flexibility to define operational accounting time periods without restriction to only one calendar.
business event	In PeopleSoft Sales Incentive Management, an original business transaction or activity that may justify the creation of a PeopleSoft Enterprise Incentive Management event (a sale, for example).
catalog item	In PeopleSoft Enterprise Learning Management, a specific topic that a learner can study and have tracked. For example, Introduction to Microsoft Word. A catalog item contains general information about the topic and includes a course code, description, categorization, keywords, and delivery methods.
category	In PeopleSoft Enterprise Learning Management, a way to classify catalog items so that users can easily browse and search relevant entries in the learning catalog. Categories can be hierarchical.
ChartField	A field that stores a chart of accounts, resources, and so on, depending on the PeopleSoft application. ChartField values represent individual account numbers, department codes, and so forth.
ChartField balancing	You can require specific ChartFields to match up (balance) on the debit and the credit side of a transaction.

ChartField combination edit The process of editing journal lines for valid ChartField combinations based on user-defined rules.

ChartKey One or more fields that uniquely identify each row in a table. Some tables contain only one field as the key, while others require a combination.

child In PeopleSoft Tree Manager trees, a child is a node or detail on a tree linked to another, higher-level node (referred to as the parent). Child nodes can be rolled up into the parent. A node can be a child and a parent at the same time depending on its location within the tree.

Class ChartField A ChartField value that identifies a unique appropriation budget key when you combine it with a fund, department ID, and program code, as well as a budget period. Formerly called *sub-classification.*

clone In PeopleCode, to make a unique copy. In contrast, to *copy* may mean making a new reference to an object, so if the underlying object is changed, both the copy and the original change.

collection To make a set of documents available for searching in Verity, you must first create at least one collection. A collection is set of directories and files that allow search application users to use the Verity search engine to quickly find and display source documents that match search criteria. A collection is a set of statistics and pointers to the source documents, stored in a proprietary format on a file server. Because a collection can only store information for a single location, PeopleSoft maintains a set of collections (one per language code) for each search index object.

compensation object In PeopleSoft Enterprise Incentive Management, a node within a compensation structure. Compensation objects are the building blocks that make up a compensation structure's hierarchical representation.

compensation structure In PeopleSoft Enterprise Incentive Management, a hierarchical relationship of compensation objects that represents the compensation-related relationship between the objects.

configuration parameter catalog Used to configure an external system with PeopleSoft. For example, a configuration parameter catalog might set up configuration and communication parameters for an external server.

configuration plan In PeopleSoft Enterprise Incentive Management, configuration plans hold allocation information for common variables (not incentive rules) and are attached to a node without a participant. Configuration plans are not processed by transactions.

content reference Content references are pointers to content registered in the portal registry. These are typically either URLs or iScripts. Content references fall into three categories: target content, templates, and template pagelets.

context In PeopleSoft Enterprise Incentive Management, a mechanism that is used to determine the scope of a processing run. PeopleSoft Enterprise Incentive Management uses three types of context: plan, period, and run-level.

corporate account Equivalent to the Account ChartField. Distinguishes between the chart of accounts typically used to record and report financial information for management, stockholders, and the general public, as opposed to a chart of statutory (alternate) accounts required by a regulatory authority for recording and reporting financial information.

cost profile A combination of a receipt cost method, a cost flow, and a deplete cost method. A profile is associated with a cost book and determines how items in that book are valued, as well as how the material movement of the item is valued for the book.

cost row A cost transaction and amount for a set of ChartFields.

data acquisition In PeopleSoft Enterprise Incentive Management, the process during which raw business transactions are acquired from external source systems and fed into the operational data store (ODS).

data elements Data elements, at their simplest level, define a subset of data and the rules by which to group them.

For Workforce Analytics, data elements are rules that tell the system what measures to retrieve about your workforce groups.

data row Contains the entries for each field in a table. To identify each data row uniquely, PeopleSoft applications use a key consisting of one or more fields in the table.

data validation In PeopleSoft Enterprise Incentive Management, a process of validating and cleansing the feed data to resolve conflicts and make the data processable.

DAT file This text file, used with the Verity search engine, contains all of the information from documents that are searchable but not returned in the results list.

delivery method In PeopleSoft Enterprise Learning Management, identifies a learning activitys delivery method type. An activity can have one or more delivery methods.

delivery method type In PeopleSoft Enterprise Learning Management, specifies a method that your organization uses to deliver learning activities, for example, scheduled or self-paced learning.

distribution The process of assigning values to ChartFields. A distribution is a string of ChartField values assigned to items, payments, and budget amounts.

double byte character If youre working with Japanese or other Asian employees, you can enter the employee's name using double-byte characters. The standard double byte character set name format in PeopleSoft applications is: [last name] space [first name].

dynamic tree A tree that takes its detail valuesdynamic detailsdirectly from a table in the database, rather than from a range of values entered by the user.

edit table A table in the database that has its own record definition, such as the Department table. As fields are entered into a PeopleSoft application, they can be validated against an edit table to ensure data integrity throughout the system.

effective date A method of dating information in PeopleSoft applications. You can predate information to add historical data to your system, or postdate information in order to enter it before it actually goes into effect. By using effective dates, you don't delete values; you enter a new value with a current effective date.

EIM job Abbreviation for *Enterprise Incentive Management job.* In PeopleSoft Enterprise Incentive Management, a collection of job steps that corresponds to the steps in an organizations compensation-related business process. An EIM job can be stopped to allow manual changes or corrections to be applied between steps, and then resumed from where it left off, continuing with the next step. A run can also be restarted or rolled back.

EIM ledger Abbreviation for *Enterprise Incentive Management ledger.* In PeopleSoft Enterprise Incentive Management, an object to handle incremental result gathering within the scope of a participant. The ledger captures a result set with all of the appropriate traces to the data origin and to the processing steps of which it is a result.

equipment In PeopleSoft Enterprise Learning Management, resource items that can be assigned to a training facility, to a specific training room, or directly to an activity session. Equipment items are generally items that are used (sometimes for a fee) and returned after the activity is complete.

event Events are predefined points either in the application processor flow or in the program flow. As each point is encountered, the event activates each component, triggering any PeopleCode program associated with that component and that event. Examples of events are FieldChange, SavePreChange, and OnRouteSubscription. In PeopleSoft Human Resources, *event* also refers to incidents that affect benefits eligibility.

event propagation process In PeopleSoft Sales Incentive Management, a process that determines, through logic, the propagation of an original PeopleSoft Enterprise Incentive Management event and creates a derivative (duplicate) of the original event to be processed by other objects. Sales Incentive Management uses this mechanism to implement splits, roll-ups, and so on. Event propagation determines who receives the credit.

external system In PeopleSoft, any system that is not directly compiled with PeopleTools servers.

fact In PeopleSoft applications, facts are numeric data values from fields from a source database as well as an analytic application. A fact can be anything you want to measure your business by, for example, revenue, actual, budget data, or sales numbers. A fact is stored on a fact table.

filter In PeopleSoft applications, a filter creates a subset of information. Filters are used in templates to limit your information from a pick list of attribute values.

generic process type In PeopleSoft Process Scheduler, process types are identified by a generic process type. For example, the generic process type SQR includes all SQR process types, such as SQR process and SQR report.

group Any set of records associated under a single name or variable in order to run calculations in PeopleSoft business processes. In PeopleSoft Time and Labor, for example, employees are placed in groups for time reporting purposes.

homepage Users can personalize the homepage, or the page that first appears when they access the portal.

incentive object In PeopleSoft Enterprise Incentive Management, the incentive-related objects that define and support the PeopleSoft Enterprise Incentive Management calculation process and results, such as plan templates, plans, results data, user interaction objects, and so on.

incentive rule In PeopleSoft Sales Incentive Management, the commands that act on transactions and turn them into compensation. A rule is one part in the process of turning a transaction into compensation.

key One or more fields that uniquely identify each row in a table. Some tables contain only one field as the key, while others require a combination.

learner group In PeopleSoft Enterprise Learning Management, a group of learners within the same learning environment that share the same attributes, such as department or job code.

learning activity See *activity.*

learning history In PeopleSoft Enterprise Learning Management, a self-service repository for all of a learner's completed learning activities.

learning plan In PeopleSoft Enterprise Learning Management, a self-service repository for all of a learner's planned and in-progress learning activities.

ledger mapping You use ledger mapping to relate expense data from general ledger accounts to resource objects. Multiple ledger line items can be mapped to one or more resource IDs. You can also use ledger mapping to map dollar amounts (referred to as *rates*) to business units. You can map the amounts in two different ways: an actual amount that represents actual costs of the accounting period, or a budgeted amount that can be used to calculate the capacity rates as well as budgeted model results. In PeopleSoft Enterprise Warehouse, you can map general ledger accounts to the EW Ledger table.

level	A section of a tree that organizes groups of nodes.
library section	In PeopleSoft Enterprise Incentive Management, a section that is defined in a plan (or template) and that is available for other plans to share. Changes to a library section are reflected in all plans that use it.
linked section	In PeopleSoft Enterprise Incentive Management, a section that is defined in a plan template but appears in a plan. Changes to linked sections propagate to plans using that section.
linked variable	In PeopleSoft Enterprise Incentive Management, a variable that is defined and maintained in a plan template and that also appears in a plan. Changes to linked variables propagate to plans using that variable.
load	The feature that initiates a process to automatically load information into a PeopleSoft applicationfor example, populating the PeopleSoft Benefits database with plan-level election information.
local functionality	In PeopleSoft HRMS, the set of information that is available for a specific country. You can access this information when you click the appropriate country flag in the global window, or when you access it by a local country menu.
location	Locations enable you to indicate the different types of addressesfor a company, for example, one address to receive bills, another for shipping, a third for postal deliveries, and a separate street address. Each address has a different location number. The primary locationindicated by a *1*is the address you use most often and may be different from the main address.
market template	In PeopleSoft Enterprise Incentive Management, additional functionality that is specific to a given market or industry and is built on top of a product category.
material	In PeopleSoft Enterprise Learning Management, a resource item that can be assigned to the sessions of an activity. Material items are generally consumed during the duration of an activity and not returned, and they may have an associated cost.
message definition	An object definition specified in PeopleSoft Application Designer that contains message information for PeopleSoft Application Messaging.
meta-SQL	Meta-SQL constructs expand into platform-specific SQL substrings. They are used in functions that pass SQL strings, such as in SQL objects, the SQLExec function, and PeopleSoft Application Engine programs.
metastring	Metastrings are special expressions included in SQL string literals. The metastrings, prefixed with a percent (%) symbol, are included directly in the string literals. They expand at run time into an appropriate substring for the current database platform.
multibook	Processes in PeopleSoft applications that can create both application entries and general ledgers denominated in more than one currency.
multicurrency	The ability to process transactions in a currency other than the business unit's base currency.
objective	In PeopleSoft Enterprise Learning Management, an individual's learning goal. An example of a learning goal is a competency gap.
override	In PeopleSoft Enterprise Incentive Management, the ability to make a change to a plan that applies to only one plan context.
pagelet	Each block of content on the homepage is called a pagelet. These pagelets display summary information within a small rectangular area on the page. The pagelet provide users with a snapshot of their most relevant PeopleSoft and non-PeopleSoft content.

parent node A tree node linked to lower-level nodes or details that roll up into it. A node can be a parent and a child at the same time, depending on its location within the tree.

participant In PeopleSoft Enterprise Incentive Management, participants are recipients of the incentive compensation calculation process.

participant object Each participant object may be related to one or more compensation objects.

See also *participant object.*

payout In PeopleSoft Enterprise Incentive Management, the resulting incentive plan computation that is provided to payroll.

PeopleCode PeopleCode is a proprietary language, executed by the PeopleSoft application processor. PeopleCode generates results based upon existing data or user actions. By using business interlink objects, external services are available to all PeopleSoft applications wherever PeopleCode can be executed.

PeopleCode event An action that a user takes upon an object, usually a record field, that is referenced within a PeopleSoft page.

PeopleSoft Internet Architecture The fundamental architecture on which PeopleSoft 8 applications are constructed, consisting of an RDBMS, an application server, a Web server, and a browser.

performance measurement In PeopleSoft Enterprise Incentive Management, a variable used to store data (similar to an aggregator, but without a predefined formula) within the scope of an incentive plan. Performance measures are associated with a plan calendar, territory, and participant. Performance measurements are used for quota calculation and reporting.

period context In PeopleSoft Enterprise Incentive Management, because a participant typically uses the same compensation plan for multiple periods, the period context associates a plan context with a specific calendar period and fiscal year. The period context references the associated plan context, thus forming a chain. Each plan context has a corresponding set of period contexts.

per seat cost In PeopleSoft Enterprise Learning Management, the cost per learner, based on the total activity costs divided by either minimum attendees or maximum attendees. Organizations use this cost to price PeopleSoft Enterprise Learning Management activities.

plan In PeopleSoft Sales Incentive Management, a collection of allocation rules, variables, steps, sections, and incentive rules that instruct the PeopleSoft Enterprise Incentive Management engine in how to process transactions.

plan context In PeopleSoft Enterprise Incentive Management, correlates a participant with the compensation plan and node to which the participant is assigned, enabling the PeopleSoft Enterprise Incentive Management system to find anything that is associated with the node and that is required to perform compensation processing. Each participant, node, and plan combination represents a unique plan contextif three participants are on a compensation structure, each has a different plan context. Configuration plans are identified by plan contexts and are associated with the participants that refer to them.

plan section In PeopleSoft Enterprise Incentive Management, a segment of a plan that handles a specific type of event processing.

plan template In PeopleSoft Enterprise Incentive Management, the base from which a plan is created. A plan template contains common sections and variables that are inherited by all plans that are created from the template. A template may contain steps and sections that are not visible in the plan definition.

portal registry In PeopleSoft applications, the portal registry is a tree-like structure in which content references are organized, classified, and registered. It is a central repository that

	defines both the structure and content of a portal through a hierarchical, tree-like structure of folders useful for organizing and securing content references.
private view	A user-defined view that is available only to the user who created it.
process	See *Batch Processes.*
process definition	Process definitions define each run request.
process instance	A unique number that identifies each process request. This value is automatically incremented and assigned to each requested process when the process is submitted to run.
process job	You can link process definitions into a job request and process each request serially or in parallel. You can also initiate subsequent processes based on the return code from each prior request.
process request	A single run request, such as an SQR, a COBOL program, or a Crystal report that you run through PeopleSoft Process Scheduler.
process run control	A PeopleTools variable used to retain PeopleSoft Process Scheduler values needed at runtime for all requests that reference a run control ID. Do not confuse these with application run controls, which may be defined with the same run control ID, but only contain information specific to a given application process request.
product category	In PeopleSoft Enterprise Incentive Management, indicates an application in the Enterprise Incentive Management suite of products. Each transaction in the PeopleSoft Enterprise Incentive Management system is associated with a product category.
publishing	In PeopleSoft Enterprise Incentive Management, a stage in processing that makes incentive-related results available to participants.
record definition	A logical grouping of data elements.
record field	A field within a record definition.
record group	A set of logically and functionally related control tables and views. Record groups help enable TableSet sharing, which eliminates redundant data entry. Record groups ensure that TableSet sharing is applied consistently across all related tables and views.
record input VAT flag	Abbreviation for *record input value-added tax flag.* Within PeopleSoft Purchasing, Payables, and General Ledger, this flag indicates that you are recording input VAT on the transaction. This flag, in conjunction with the record output VAT flag, is used to determine the accounting entries created for a transaction and to determine how a transaction is reported on the VAT return. For all cases within Purchasing and Payables where VAT information is tracked on a transaction, this flag is set to Yes. This flag is not used in PeopleSoft Order Management, Billing, or Receivables, where it is assumed that you are always recording only output VAT, or in PeopleSoft Expenses, where it is assumed that you are always recording only input VAT.
record output VAT flag	Abbreviation for *record output value-added tax flag.* See *record input VAT flag.*
reference data	In PeopleSoft Sales Incentive Management, system objects that represent the sales organization, such as territories, participants, products, customers, channels, and so on.
reference object	In PeopleSoft Enterprise Incentive Management, this dimension-type object further defines the business. Reference objects can have their own hierarchy (for example, product tree, customer tree, industry tree, and geography tree).
reference transaction	In commitment control, a reference transaction is a source transaction that is referenced by a higher-level (and usually later) source transaction, in order to

	automatically reverse all or part of the referenced transaction's budget-checked amount. This avoids duplicate postings during the sequential entry of the transaction at different commitment levels. For example, the amount of an encumbrance transaction (such as a purchase order) will, when checked and recorded against a budget, cause the system to concurrently reference and relieve all or part of the amount of a corresponding pre-encumbrance transaction, such as a purchase requisition.
relationship object	In PeopleSoft Enterprise Incentive Management, these objects further define a compensation structure to resolve transactions by establishing associations between compensation objects and business objects.
results management process	In PeopleSoft Sales Incentive Management, the process during which compensation administrators may review processing results, manually change transactions, process draws, update and review payouts, process approvals, and accumulate and push payments to the EIM ledger.
role user	A PeopleSoft Workflow user. A person's role user ID serves much the same purpose as a user ID does in other parts of the system. PeopleSoft Workflow uses role user IDs to determine how to route worklist items to users (through an email address, for example) and to track the roles that users play in the workflow. Role users do not need PeopleSoft user IDs.
role	Describes how people fit into PeopleSoft Workflow. A role is a class of users who perform the same type of work, such as clerks or managers. Your business rules typically specify what user role needs to do an activity.
roll up	In a tree, to roll up is to total sums based on the information hierarchy.
routing	Connects activities in PeopleSoft Workflow. Routings specify where the information goes and what form it takesemail message, electronic form, or worklist entry.
run control	A run control is a type of online page that is used to begin a process, such as the batch processing of a payroll run. Run control pages generally start a program that manipulates data.
run control ID	A unique ID to associate each user with his or her own run control table entries.
run-level context	In PeopleSoft Enterprise Incentive Management, associates a particular run (and batch ID) with a period context and plan context. Every plan context that participates in a run has a separate run-level context. Because a run cannot span periods, only one run-level context is associated with each plan context.
search query	You use this set of objects to pass a query string and operators to the search engine. The search index returns a set of matching results with keys to the source documents.
section	In PeopleSoft Enterprise Incentive Management, a collection of incentive rules that operate on transactions of a specific type. Sections enable plans to be segmented to process logical events in different sections.
security event	In commitment control, security events trigger security authorization checking, such as budget entries, transfers, and adjustments; exception overrides and notifications; and inquiries.
self-service application	Self-service refers to PeopleSoft applications that are accessed by end users with a browser.
session	In PeopleSoft Enterprise Learning Management, a single meeting day of an activity (that is, the period of time between start and finish times within a day). The session stores the specific date, location, meeting time, and instructor. Sessions are used for scheduled training.
session template	In PeopleSoft Enterprise Learning Management, enables you to set up common activity characteristics that may be reused while scheduling a PeopleSoft Enterprise

	Learning Management activitycharacteristics such as days of the week, start and end times, facility and room assignments, instructors, and equipment. A session pattern template can be attached to an activity that is being scheduled. Attaching a template to an activity causes all of the default template information to populate the activity session pattern.
setup relationship	In PeopleSoft Enterprise Incentive Management, a relationship object type that associates a configuration plan with any structure node.
sibling	A tree node at the same level as another node, where both roll up into the same parent. A node can be a sibling, parent, and child all at the same time, depending on its location in the tree.
single signon	With single signon, users can, after being authenticated by a PeopleSoft application server, access a second PeopleSoft application server without entering a user ID or password.
source transaction	In commitment control, any transaction generated in a PeopleSoft or third-party application that is integrated with commitment control and which can be checked against commitment control budgets. For example, a pre-encumbrance, encumbrance, expenditure, recognized revenue, or collected revenue transaction.
SpeedChart	A user-defined shorthand key that designates several ChartKeys to be used for voucher entry. Percentages can optionally be related to each ChartKey in a SpeedChart definition.
SpeedType	A code representing a combination of ChartField values. SpeedTypes simplify the entry of ChartFields commonly used together.
SQR	See *Structured Query Report (SQR)*.
statutory account	Account required by a regulatory authority for recording and reporting financial results. In PeopleSoft, this is equivalent to the Alternate Account (ALTACCT) ChartField.
step	In PeopleSoft Sales Incentive Management, a collection of sections in a plan. Each step corresponds to a step in the job run.
Structured Query Report (SQR)	A type of printed or displayed report generated from data extracted from a PeopleSoft SQL-based relational database. PeopleSoft applications provide a variety of standard SQRs that summarize table information and data. You can use these reports as is, customize them, or create your own.
Summary ChartField	You use summary ChartFields to create summary ledgers that roll up detail amounts based on specific detail values or on selected tree nodes. When detail values are summarized using tree nodes, summary ChartFields must be used in the summary ledger data record to accommodate the maximum length of a node name (20 characters).
summary ledger	An accounting feature used primarily in allocations, inquiries, and PS/nVision reporting to store combined account balances from detail ledgers. Summary ledgers increase speed and efficiency of reporting by eliminating the need to summarize detail ledger balances each time a report is requested. Instead, detail balances are summarized in a background process according to user-specified criteria and stored on summary ledgers. The summary ledgers are then accessed directly for reporting.
summary tree	A tree used to roll up accounts for each type of report in summary ledgers. Summary trees enable you to define trees on trees. In a summary tree, the detail values are really nodes on a detail tree or another summary tree (known as the *basis* tree). A summary tree structure specifies the details on which the summary trees are to be built.

table The underlying PeopleSoft data format, in which data is stored by columns (fields) and rows (records, or instances).

TableSet sharing Specifies control table data for each business unit so that redundancy is eliminated.

target currency The value of the entry currency or currencies converted to a single currency for budget viewing and inquiry purposes.

template A template is HTML code associated with a Web page. It defines the layout of the page and also where to get HTML for each part of the page. In PeopleSoft, you use templates to build a page by combining HTML from a number of sources. For a PeopleSoft portal, all templates must be registered in the portal registry, and each content reference must be assigned a template.

territory In PeopleSoft Sales Incentive Management, hierarchical relationships of business objects, including regions, products, customers, industries, and participants.

TimeSpan A relative period, such as year-to-date or current period, that can be used in various PeopleSoft General Ledger functions and reports when a rolling time frame, rather than a specific date, is required. TimeSpans can also be used with flexible formulas in PeopleSoft Projects.

transaction allocation In PeopleSoft Enterprise Incentive Management, the process of identifying the owner of a transaction. When a raw transaction from a batch is allocated to a plan context, the transaction is duplicated in the PeopleSoft Enterprise Incentive Management transaction tables.

transaction loading process In PeopleSoft Enterprise Incentive Management, the process during which transactions are loaded into Sales Incentive Management. During loading, the source currency is converted to the business unit currency while retaining the source currency code. At the completion of this stage, the transaction is in the first state.

transaction state In PeopleSoft Enterprise Incentive Management, a value assigned by an incentive rule to a transaction. Transaction states enable sections to process only transactions that are at a specific stage in system processing. After being successfully processed, transactions may be promoted to the next transaction state and picked up by a different section for further processing.

transaction type In PeopleSoft Enterprise Incentive Management, a way to categorize transactions to identify specific transaction types (for example, shipment, order, opportunity, and so on). Plan sections process only one type of transaction type. Transaction types can be defined based on a company's specific processes model.

Translate table A system edit table that stores codes and translate values for the miscellaneous fields in the database that do not warrant individual edit tables of their own.

tree The graphical hierarchy in PeopleSoft systems that displays the relationship between all accounting units (for example, corporate divisions, projects, reporting groups, account numbers) and determines roll-up hierarchies.

unclaimed transaction In PeopleSoft Enterprise Incentive Management, a transaction that is not claimed by a node or participant after the allocation process has completed, usually due to missing or incomplete data. Unclaimed transactions may be manually assigned to the appropriate node or participant by a compensation administrator.

uniform resource locator (URL) In PeopleSoft, the term URL refers to the entire query string. The following is an example of a URL: http://serverx/InternetClient /InternetClientServlet?ICType=Script&ICScriptProgramName=WEBLIB_BEN_ 401k.PAGES.FieldFormula.iScript_Home401k

universal navigation header Every PeopleSoft portal includes the universal navigation header, intended to appear at the top of every page as long as the user is signed on to the portal. In addition to

	providing access to the standard navigation buttons (like Home, Favorites, and signoff) the universal navigation header can also display a welcome message for each user.
URL	See *uniform resource locator (URL).*
user interaction object	In PeopleSoft Sales Incentive Management, used to define the reporting components and reports that a participant can access in his or her context. All Sales Incentive Management user interface objects and reports are registered as user interaction objects. User interaction objects can be linked to a compensation structure node through a compensation relationship object (individually or as groups).
variable	In PeopleSoft Sales Incentive Management, the intermediate results of calculations. Variables hold the calculation results and are then inputs to other calculations. Variables can be plan variables that persist beyond the run of an engine or local variables that exist only during the processing of a section.
warehouse	A PeopleSoft data warehouse that consists of predefined ETL maps, data warehouse tools, and DataMart definitions.
worksheet	A way of presenting data through a PeopleSoft Business Analysis Modeler interface that enables users to do in-depth analysis using pivoting tables, charts, notes, and history information.
workflow	The background process that creates a list of administrative actions based on selection criteria and specifies the procedure associated with each action.
worklist	The automated to-do list that PeopleSoft Workflow creates. From the worklist, you can directly access the pages you need to perform the next action, and then return to the worklist for another item.
zero-rated VAT	Abbreviation for *zero-rated value-added tax.* A VAT transaction with a VAT code that has a tax percent of zero. Used to track taxable VAT activity where no actual VAT amount is charged.

Index