

PeopleSoft®

EnterpriseOne 8.93
Interoperability
PeopleBook

May 2004

EnterpriseOne 8.93
Interoperability PeopleBook
SKU TT893IG0504

Copyright© 2004 PeopleSoft, Inc. All rights reserved.

All material contained in this documentation is proprietary and confidential to PeopleSoft, Inc. ("PeopleSoft"), protected by copyright laws and subject to the nondisclosure provisions of the applicable PeopleSoft agreement. No part of this documentation may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, including, but not limited to, electronic, graphic, mechanical, photocopying, recording, or otherwise without the prior written permission of PeopleSoft.

This documentation is subject to change without notice, and PeopleSoft does not warrant that the material contained in this documentation is free of errors. Any errors found in this document should be reported to PeopleSoft in writing.

The copyrighted software that accompanies this document is licensed for use only in strict accordance with the applicable license agreement which should be read carefully as it governs the terms of use of the software and this document, including the disclosure thereof.

PeopleSoft, PeopleTools, PS/nVision, PeopleCode, PeopleBooks, PeopleTalk, and Vantive are registered trademarks, and Pure Internet Architecture, Intelligent Context Manager, and The Real-Time Enterprise are trademarks of PeopleSoft, Inc. All other company and product names may be trademarks of their respective owners. The information contained herein is subject to change without notice.

Open Source Disclosure

This product includes software developed by the Apache Software Foundation (<http://www.apache.org/>). Copyright (c) 1999-2000 The Apache Software Foundation. All rights reserved. THIS SOFTWARE IS PROVIDED "AS IS" AND ANY EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE APACHE SOFTWARE FOUNDATION OR ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

PeopleSoft takes no responsibility for its use or distribution of any open source or shareware software or documentation and disclaims any and all liability or damages resulting from use of said software or documentation.

Table of Contents

About These EnterpriseOne PeopleBooks Preface	1
EnterpriseOne Application Prerequisites	1
Obtaining Documentation Updates	1
Typographical Conventions and Visual Cues	2
Typographical Conventions.....	2
Visual Cues	3
Comments and Suggestions	3
Acronyms	4
Interoperability Overview	6
Interoperability Features	6
Benefits	7
Interoperability Models and Capabilities	7
Interoperability Capabilities	8
Interoperability Models	10
Choosing an Interoperability Model	15
Other Industry Standard Support	16
Business Function Calls	17
Finding the Right Business Function.....	18
Reviewing Online API and Business Function Documentation	18
Creating Business Function Documentation	19
Using an Application as a Model.....	19
XML Solution	21
Formatting an XML Document	22
Type Element	23
Establish Session.....	23
Expire Session	24
Explicit Transaction	24
Implicit Transaction	24
Prepare/Commit/Rollback	25
Terminate Session	25
Standards for Creating XML Documents	25
Decimal and Comma Separators	26
Date Usage	26
Configure the System Environment for XML	26
UNIX.....	27
iSeries	27
WIN32	27

XML Transformation Service.....	28
XTS Processing.....	28
Build a Custom Selector.....	33
XTS APIs.....	33
Configure the jde.ini File for XTS	41
XML Dispatch.....	43
XML Dispatch Process.....	43
XML Dispatch Recognizers.....	44
XML Dispatch Transports.....	44
Configure the jde.ini File for XML Dispatch.....	44
XML Dispatch Error Handling.....	46
XML CallObject	47
XML CallObject Templates	47
XML CallObject Process	49
XML CallObject Elements	50
Configure the jde.ini File for XML CallObject	55
Return Codes for CallObject	58
XML Transaction	59
XML Transaction Update Process	59
XML Transaction Data Request Process.....	60
Configure the jde.ini File for XML Transaction.....	61
XML List	63
List-Retrieval Engine Table Conversion Wrapper.....	63
XML List Process	64
XML List Requests	65
Configure the jde.ini File for the List-Retrieval Engine.....	72
Configure the jde.ini File for XML List.....	72
Troubleshooting: XML Kernel.....	73
Z Transactions	74
Processing Z Transactions.....	74
Name the Transaction.....	75
Add Records to the Inbound Interface Table	75
Run an Update Process.....	75
Check for Errors	78
Confirm the Update	78
Purge the Data from the Interface Table.....	79
Flat Files	80
Flat File Formats	81
Setup Requirements for Flat File Conversion	81
Converting Flat Files	82
Flat File Conversion Program	82
Import Flat Files Using a Business Function.....	86
Flat File Conversion Error Messages.....	87
Flat File APIs.....	87

Events	90
Z Events	92
Enabling Z Event Processing	94
Setting Up Flat File Cross-Reference	95
Setting Up Data Export Controls	95
Processing Log Table	96
Verifying that the Subsystem Job is Running	96
Purging Data from the Interface Table	97
Z Event Sequencing	97
Configure the jde.ini File for Z Events	97
Z File Events XML Document Format	98
Vendor-Specific Outbound Functions	98
Real-Time Events	99
Event Unique ID	101
Journaling	101
Real-Time Event APIs	103
Generating Real-Time Events	107
XAPI	111
XAPI Outbound Events	113
XAPI Inbound Response	119
XAPI PeopleSoft EnterpriseOne to PeopleSoft EnterpriseOne	125
jde.ini File Configurations for Events	146
Defining Events	148
Setting Up Subscriber Information	157
Reliable Event Delivery	162
Configure Your System for Reliable Event Delivery	163
Events Self-Diagnostic Utility Tool	163
Events Self-Diagnostic Utility Tool Process Overview	164
Events Self-Diagnostic Utility Tool Components	165
Setting Up Database Tables for Self-Diagnostic Events Generation	166
Executing the Event Self-Diagnostic Tool	166
Batch Interfaces	171
Interface Tables	171
Interface Table Structure	171
Inbound Interface Table Processing	173
Outbound Interface Table Processing	174
Interface Table Maintenance	175
Interoperability Interface Table Information	176
Electronic Data Interface (EDI)	179
Table Conversion	179
Output Stream Access (OSA) UBEs	179
Advanced Planning Agent (APAg)/Integration	180

Open Data Access (ODA)	181
Hardware and Software Requirements	181
Hardware Requirements	181
Software Requirements.....	182
ODBC Component Files.....	182
Open Data Access (ODA) Driver Architecture	182
Adding an ODA Data Source	183
Adding a File Data Source	184
Adding a System Data Source	186
Modifying a Data Source	187
Deleting a Data Source	188
Using Keywords in the Connection String	188
Working with ODA.....	189
Running a Query Using Microsoft Excel	191
ODA Error Messages	192
XML Format Examples (All Parameters)	197
Example: Inbound Sales Order XML Format (All Parameters)	197
Example: Outbound Customer Create XML Request and Response Format (All Parameters)	208
XML Format Examples (Default Values)	212
Example: Inbound Sales Order XML Format	212
XML Format Examples (Events)	214
Example: Z Events XML Format	214
Example: Real-Time Events Template.....	231
EnterpriseOne PeopleBooks Glossary	237
Index	271

About These EnterpriseOne PeopleBooks

Preface

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

This preface discusses:

- EnterpriseOne application prerequisites
- Obtaining documentation updates
- Typographical elements and visual cues
- Comments and suggestions

Note

EnterpriseOne PeopleBooks document only fields that require additional explanation. If a field 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.

EnterpriseOne Application Prerequisites

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

See the *Foundation Guide*.

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

You should be familiar with navigating the system and adding, updating, and deleting information by using EnterpriseOne menus and forms. 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 EnterpriseOne applications most effectively.

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 can find a variety of useful and timely materials, including updates to the full PeopleSoft documentation that is delivered on your PeopleBooks CD-ROM.

Note

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.jsp>

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 EnterpriseOne PeopleBooks:

Typographical Convention or Visual Cue	Description
<i>Italics</i>	Indicates emphasis, topic titles, and titles of PeopleSoft or other book-length publications. Also used in code to indicate variable values.
Key+Key	A plus sign (+) between keys means that you must hold down the first key while you press the second key. For example, Alt+W means hold down the Alt key while you press W.
Monospace font	Indicates a PeopleCode program or other code example.
“ ” (quotation marks)	Indicates an adjective that is used in a way that might not be readily understood without the quotation marks, for example "as of" date, "as if" currency, "from" date, and "thru" date.
Cross-references	EnterpriseOne PeopleBooks provide cross-references either below the heading "See Also" or preceded by the word See. Cross-references lead to other documentation that is pertinent to the immediately preceding documentation.

Visual Cues

EnterpriseOne PeopleBooks contain the following visual cues:

- Notes
- Cautions

Notes

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

Note

Example of a note.

Cautions

Text that is preceded by *Caution* is crucial and includes information that concerns what you must do for the system to function properly.

Caution

Example of a caution.

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 you can send e-mail comments to doc@peoplesoft.com.

While we cannot guarantee an answer to every e-mail message, we will pay careful attention to your comments and suggestions.

Acronyms

The following acronyms are commonly used in PeopleSoft EnterpriseOne and might appear in the *Interoperability Guide*.

APAg	Advanced Planning Agent
API	Application Programming Interface
APPL	Application
BDA	Business View Design Aid
BSFN	Business Function
BSVW	Business View
COM	Component Object Model
CRP	Conference Room Pilot
DBMS	Database Management System
DCOM	Distributed Component Object Model
DD	Data Dictionary
DLL	Dynamic Link Library
DS or DSTR	Data Structure
EDI	Electronic Data Interchange
ER	Event Rules
FDA	Form Design Aid
IDL	Interface Definition Language
NER	Named Event Rules
ODA	Open Data Access
ODBC	Open Database Connectivity
OCM	Object Configuration Manager
OL	Object Librarian
OSA	Output Stream Access
QBE	Query by Example
RDA	Report Design Aid

SAR	Software Action Request
Specs	Specifications
SQL	Structured Query Language
TBLE	Table
TC	Table Conversion
TDA	Table Design Aid
TER	Table Event Rules
UBE	Universal Batch Engine
WF	Workflow
XAPI	Extended Application Programming Interface
XML	Extensible Markup Language
XPI	Extended Process Integration
XTS	XML Transformation Service

Interoperability Overview

Interoperability is most often associated with software as a way to allow disparate software applications to work together. For example, interoperability makes it possible for a company to use applications from different vendors as if they were from a single vendor. Seamless sharing of function and information becomes possible.

Interoperability reduces or eliminates the problems of islands of automation. It allows business processes to flow from one application to another. Interoperability allows one system to work with another, in a near real-time fashion, to share critical business information. Interoperability options become the "glue" between systems and applications.

Interoperability Features

Full interoperability among systems makes the flow of data among the systems seamless to the user. PeopleSoft EnterpriseOne provides a framework to mask the complexity of interoperability with external systems, and to simplify interfacing with third-party packages.

The interoperability solution for EnterpriseOne meets the following three important business objectives:

- Flexibility, Options, and Choice

PeopleSoft gives you several possibilities in the types of applications and information you want to work with EnterpriseOne—legacy, best-of-breed, customer management, reporting tools, and many more. The developer can make the right choice for your particular environment and needs.

- Investment Preservation

If you have existing applications you want to continue using, or if you have applications you are thinking about adopting, you can interface them to EnterpriseOne. You can use industry standard methods if the existing or new technologies support them, or you can use EnterpriseOne business logic to create this interoperability. Also, you will benefit from our ongoing upgrades and improvements to that architecture.

- Manageability

EnterpriseOne is designed to make the interoperability process easily manageable.

Benefits

Interoperability offers the following benefits:

- Businesses can bring together applications and systems across an enterprise, irrespective of vendors.
- Collaborations can occur between trading partners to lower the cost of doing business or to increase competitiveness.
- Multiple systems can be linked together to share information in a real-time manner, delivering time-sensitive information to those who need it.
- Disparate solutions encountered because of mergers or acquisitions can be quickly incorporated into the enterprise's information technology solution.

The PeopleSoft EnterpriseOne interoperability strategy includes wide ranging models and capabilities.

Interoperability Models and Capabilities

The following matrix provides an overview of interoperability models that are supported by PeopleSoft EnterpriseOne. The matrix shows the models, which are further divided into types and into the capabilities that can be used with each model type. The model and model types are listed in the left-hand column. Capabilities, which are ways to access EnterpriseOne data, are shown as columns in the matrix. For each model type, you can read across the table to see what capabilities can be used with that model type. PeopleSoft provides both interactive and batch capabilities. The capabilities are grouped by inbound, outbound, and batch. An inbound capability is a request for data or a transaction initiated outside of EnterpriseOne. An outbound capability originates inside of EnterpriseOne.

PeopleSoft EnterpriseOne Interoperability									
Capability Model	Inbound to EnterpriseOne				Outbound from EnterpriseOne			Batch	
	BSFN Calls	XML CallObj., XML List, XML Trans.	Z Trans.	Flat Files	Events		Generate XML Output	Flat Files	Recommended for Batch
					Real-Time and XAPI Events	Z Events			
XPI									
Enterprise XPI	Y	List	N	Y	Y	Y	Y	Y	N
Inter-Enterprise XPI	Y	List	N	Y	Y	Y	Y	Y	N
Connectors									
Dynamic Java Connector (Java Connector)	Y	CO, Trans, List	N	N	Y	Y	Y	N	N

PeopleSoft EnterpriseOne Interoperability									
	Inbound to EnterpriseOne				Outbound from EnterpriseOne				Batch
JCA Resource Adapter	Y	CO, Trans, List	N	N	N	N	N	N	N
COM Connector	Y	CO, Trans, List	N	N	Y	Y	Y	N	N
EnterpriseOne Messaging Adapters									
Adapter for MQSeries	Y	CO, Trans	Y	N	Y	Y	Y	N	Y
Adapter for MSMQ	Y	CO, Trans	Y	N	Y	Y	Y	N	Y
Batch Interfaces									
Interface Tables	Y	N	Y	Y	N	N	N	N	Y
EnterpriseOne EDI	Y	N	Y	Y	N	N	N	Y	Y
Table Conversions	Y	N	Y	Y	N	N	N	Y	Y
OSA (UBE)	N	N	N	N	N	N	Y	N	Y
APAg/ Integration	N	N	Y	N	Y	N	Y	N	Y
Open Data Access	Supports business view and table inquiries				N				N

Interoperability Capabilities

A capability is a way to transfer information into PeopleSoft EnterpriseOne or to retrieve information from EnterpriseOne. The interoperability matrix shows inbound and outbound capabilities and identifies capabilities that are appropriate for batch processing. Inbound capabilities let you inquire about data and update (add, change, or delete) data. With inquiry capabilities, you retrieve data for information purposes only. For example, you might want to see prices or availability of an item. Update capabilities can be performed on an individual transaction basis or in a batch process, which consists of groups of transactions. An individual transaction update involves updating a single record (for example, adding a purchase order or creating an invoice). Batch processes (groups of transactions) typically involve updating multiple records, are usually scheduled to occur at a specific time, and are non-interactive. Examples of batch processing are to upload 10,000 orders to the database at the end of the day or to obtain all of the pricing information that has changed and send that information to a Web site at the end of the day.

The capabilities available for transferring information into and retrieving information from EnterpriseOne are described briefly in the following paragraphs. Each capability is discussed in further detail in other areas of this guide.

Business Function Calls

Business function calls are core to the PeopleSoft EnterpriseOne interoperability. Business functions encapsulate transaction logic to perform specific tasks, such as journal entry transactions, depreciation calculations, and sales order transactions.

EnterpriseOne uses two types of business functions: regular business functions and master business functions. A regular business function performs simple tasks, such as tax calculation or account number validation. A master business function (MBF) performs complex tasks, and can call several regular business functions to perform those tasks.

See Also

- ❑ *Business Functions* in the *Development Tools Guide* for more information about business functions

XML

XML provides a flexible, standards-based way of sharing information and moving data among systems. XML allows you to extend enterprise applications and collaborate with business partners and customers. You can use XML CallObject and XML Transaction to update or retrieve PeopleSoft EnterpriseOne data. You can use XML List to create an XML Data file in the EnterpriseOne system repository and then to retrieve the data in small chunks to avoid network traffic. EnterpriseOne output is an XML document.

Z Transactions

Z transactions provide inbound capability to PeopleSoft EnterpriseOne that allows you to update EnterpriseOne data. EnterpriseOne provides interface tables (Z tables) that support Z transaction capability. You also can create your own Z tables.

Flat Files

Flat files (also known as user-defined formats) are text files that are usually stored on your workstation or server. Flat files do not have relationships defined for them, and they typically use the Unicode character set. Data in a flat file is stored as one continuous string of information. You can use flat files to import or export data from applications that have no other means of interaction. For example, you might want to share information between PeopleSoft EnterpriseOne and another system.

Events

Events are notifications to third-party applications or end users that a PeopleSoft EnterpriseOne business transaction has occurred. EnterpriseOne supports three kinds of events: Z events, real-time events, and XAPI events. Event data is represented as an XML document.

Z events use interface tables and a batch process to retrieve transaction information, and then use a Z event generator and the data export subsystem to manage the flow of the outbound data.

Real-time events can be generated from a server or a client. System calls (from a server) and client business function calls (from a client) retrieve transaction information. The transaction information is distributed to subscribers.

XAPI events are real-time events that require a response. A XAPI event is created in the same manner as a real-time event, with additional data structure information for invoking a business function when the response XML document is received.

Extended Process Integration (XPI)

XPI is the Extended Process Integration suite. It consists of two major products: Enterprise XPI and Inter-Enterprise XPI.

- Enterprise XPI provides the infrastructure for internal collaboration within the enterprise.
- Inter-Enterprise XPI provides the infrastructure for business-to-business collaboration beyond the enterprise.

Some benefits of using Enterprise XPI are:

- Many adapters already exist.
- A common access point is provided for security and permissions.
- Scalability is achieved by adding brokers and adapters.
- Message delivery is guaranteed.
- Graphic tools exist for creating integrations and workflows.
- Predefined extended business processes (XBPs) and canonicals provide easy integration with PeopleSoft data.

Some benefits of using Inter-Enterprise XPI are:

- It provides interoperability to external sources, such as vendors and trading partners.
- It provides guaranteed delivery and security.
- It uses industry formats, such as standard XML, EDI, OAG, RosettaNet, BizTalk, eXML, and ebXML.
- It provides workflow and mapping functionality.
- Packaged transactions for RosettaNet are available.

For more information about using XPI, see the Enterprise XPI and Inter-Enterprise XPI documentation. Available documentation includes information about Enterprise foundation, Inter-enterprise foundation, adapters, and XBPs.

Interoperability Models

A model is a way for third parties to connect to or access the PeopleSoft EnterpriseOne system. EnterpriseOne supports the following five basic interoperability models:

- Extended Process Integration (XPI)
- Connectors
- Messaging Adapters
- Batch Interfaces
- Open Data Access (ODA)

These models can be further categorized by type. Each model type supports one or more of the capabilities for sending information into or retrieving information from the EnterpriseOne database. The Interoperability Models and Capabilities matrix introduced previously shows the model types and the capabilities that each model type supports. An overview of each of the model types is provided in the following paragraphs.

Connectors

Connectors are point-to-point component-based models that allow third-party applications and PeopleSoft EnterpriseOne to share logic and data. EnterpriseOne connector architecture includes Java and COM connectors. The connectors accept inbound XML requests and expose business functions for reuse. Output from the connectors is in the form of an XML document.

- Java – The EnterpriseOne dynamic Java and Java connectors support real-time event processing. Java is a portable language, so you can easily tie EnterpriseOne functionality to Java applications.
- COM – The EnterpriseOne COM connector solution is fully compliant with the Microsoft component object model. You can easily tie EnterpriseOne functionality to Visual Basic and VC++ applications. The COM connector also supports real-time event processing.

Some benefits of using connectors are as follows:

- Scalable
- Designed to be multi-threaded
- Allow concurrent users

See Also

- *Connectors* and other topics in the *Connectors Guide* for more information about PeopleSoft EnterpriseOne connectors

Messaging Adapters

PeopleSoft EnterpriseOne provides messaging support for MQSeries and Microsoft Message Queuing (MSMQ). MQSeries and MSMQ handle message queuing, message delivery, and transaction monitoring. EnterpriseOne can use these messaging systems to handle and pass requests for logic and data between EnterpriseOne and third-party systems.

Some of the benefits of using messaging adapters are as follows:

- Reliable connection
- Guaranteed delivery
- Operations acknowledgement

See Also

- *Adapter Overview* and other topics in the *Messaging Adapter Programmer's Guide* for more information about using the messaging adapters

Batch Interfaces

Batch implies processing multiple transactions at the same time and usually involves movement of bulk information. Batch processing is often scheduled and is non-interactive. PeopleSoft EnterpriseOne provides several model types for batch processing, and each model type has one or more capabilities that allow you to access EnterpriseOne data. The model types are as follows:

- Interface Tables
- Electronic Data Exchange (EDI)
- Table Conversions
- UBEs
- APAg/Integration

Interface Tables

Interface tables provide point-to-point interoperability solutions for importing and exporting data. Interface tables are also called Z tables. Interface tables are working files into which you place transaction information to be processed into or out of PeopleSoft EnterpriseOne. In addition to the interface tables provided by EnterpriseOne, you can build your own interface tables. If you use interfaces tables to update EnterpriseOne data, the format of the data must be presented in the format defined by EnterpriseOne. If you use interface tables to retrieve EnterpriseOne data, you use a batch process that extracts the data from the applications tables.

Some of the benefits of using interface tables are as follows:

- Defined data structure
- Identifiable fields
- Customizable interface tables

Electronic Data Interchange (EDI)

EDI provides a point-to-point interoperability solution for importing and exporting data. EDI is the paperless computer-to-computer exchange of business transactions, such as purchase orders and invoices, in a standard format with standard content. As such, it is an important part of an electronic commerce strategy.

When computers exchange data using EDI, the data is transmitted in EDI standard format so it is recognizable by other systems using the same EDI standard format. Companies that use EDI must have translator software to convert the data from the EDI standard format to the format of their computer system.

The PeopleSoft EnterpriseOne Data Interface for Electronic Data Interchange system acts as an interface between the EnterpriseOne system data and the translator software. In addition to exchanging EDI data, this data interface also can be used for general interoperability and electronic commerce needs where a file-based interface meets the business requirements.

Some benefits of using the Data Interface for Electronic Data Interchange system are as follows:

- Shorter fulfillment cycle
- Increased information integrity through reduced manual data entry
- Reduced manual clerical work

EDI is particularly effective at sending information to multiple applications simultaneously.

See Also

- ❑ *Overview for Data Interface for Electronic Data Interchange System* and other topics in the *Data Interface for Electronic Data Guide* for information about using electronic data interchange for interoperability

Table Conversion

Table conversion provides a point-to-point interoperability solution for importing and exporting data. Table conversion is a special form of Universal Batch Engine (UBE) that allows you to do high-speed manipulation of data in tables. PeopleSoft EnterpriseOne has a table conversion utility that you can use to gather, format, import, and export data. The table conversion tool allows you to transfer and copy data. You can also delete records from tables. Table conversion allows you to use a non-EnterpriseOne table to process, call direct business functions, and give an output. For example, you might want to run a UBE that reads from an EnterpriseOne master file to populate a non-EnterpriseOne table.

The table conversion utility can make use of any EnterpriseOne table, business view, and text file, or any table that is not an EnterpriseOne table but resides in a database that is supported by EnterpriseOne, such as Oracle, Access, iSeries, or SQL Server. These non-EnterpriseOne tables are commonly referred to as foreign tables.

See Also

- ❑ *Table Conversions Overview* and other topics in the *Table Conversion Guide* for more information about table conversions

Output Stream Access (OSA)

OSA provides a point-to-point interoperability solution for exporting data from UBEs. OSA allows you to set up an interface for PeopleSoft EnterpriseOne to pass data to another software package, such as Microsoft® Excel, for processing.

The benefits for using OSA are as follows:

- Eliminates the manual task of manually formatting output
- Can employ the processing power of the target software program

See Also

- ❑ *Output Stream Access* in the *EnterpriseOne Report Writing Guide* for more information about using Output Stream Access (OSA)

Advanced Planning Agent (APAg)/Integration

The PeopleSoft EnterpriseOne Advanced Planning Agent (APAg) is a tool for batch extraction, transforming, and loading enterprise data. APAg supports access to data sources in the form of relational databases, flat file format, and other data or message encoding such as XML. APAg also moves data from one place to another and initiates tasks related to the movement of the data.

Benefits of using the APAg tool are as follows:

- Copies massive amounts of table data
- Efficiently and effectively accomplishes initial data loads

See Also

- ❑ *APAg Integrates Your System* and other topics in the *Advanced Planning Agent 3_3_3 User Guide* for information about the APAg tool

Open Data Access (ODA)

ODA provides the capability for you to extract PeopleSoft EnterpriseOne data (using SQL statements) so that you can summarize information and generate reports. You can use ODA with any of the following desktop applications:

- Microsoft Query
- Microsoft Access
- Microsoft Excel
- ODBCTEST
- Crystal Report
- Microsoft Analysis Service

ODA sits between the front-end query and reporting applications and the EnterpriseOne-configured ODBC drivers.

The EnterpriseOne database contains object and column names, specific data types, and security rules that must be converted or applied so that the data is presented correctly. The specific data types and rules include decimal shifting, Julian date, currency, media object, security, and user defined codes. In some instances, ODA modifies the SQL SELECT statement, as well as the data, so that it appears correctly within the selected application.

Some of the benefits of using ODA are as follows:

- Read-only access to all EnterpriseOne data, including the entire data dictionary
- Use of the same security rules that you established for EnterpriseOne
- Ability to extract EnterpriseOne data easily

See Also

- ❑ *Open Data Access (ODA)* in the *Interoperability Guide* for more information about using ODA

Choosing an Interoperability Model

Choosing an interoperability model depends on your business needs. The following matrix can help you determine which interoperability model best supports your interoperability requirements.

PeopleSoft EnterpriseOne Interoperability					
Model	Platforms (Windows, UNIX, iSeries)	Integration Model	Best Fit Programming Languages	Critical Technical Skills for Creating Inbound Transaction	Critical Technical Skills for Creating Outbound Transactions
XPI					
Enterprise XPI	SUN, AIX, Windows	Broker	Java	XPI Toolset	XPI Toolset, Real- Time Events, XAPI Events
Inter-Enterprise XPI	SUN, AIX, Windows	Broker	Java	XPI Toolset	XPI Toolset, Real- Time Events, XAPI Events
Connectors					
Java Connector	All	Point-to- Point	Java	Java APIs, GenJava	Real-Time Events, XAPI Events
COM Connector	Windows	Point-to- Point	C/C++/VB	COM, GenCOM	N/A
Messaging Adapters					
Adapter for MQSeries	All	Broker	HTML, C/C++, Java	MQSeries, XML	Z-Tables, Subsystem Processing (includes R00460, Data Export Controls, and so on)
Adapter for MSMQ	Windows	Broker	C/C++	MSMQ, XML	Z-Tables, Subsystem Processing (includes R00460, Data Export Controls, and so on)
Batch Interfaces					
Interface Tables	All	Point-to- Point	Any	Z-Tables, UBEs	Custom Code
EnterpriseOne EDI	All	Point-to- Point	Any, Flat Files	Z-Tables, UBEs	Custom Code
Table Conversions	All	Point-to- Point	ERP TC	Table Conversions	Table Conversion Director/RDA
OSA (UBE)	All	Point-to- Point	HTML, C/C++	N/A	RDA, Custom Code

PeopleSoft EnterpriseOne Interoperability					
Considerations Model	Platforms (Windows, UNIX, iSeries)	Integration Model	Best Fit Programming Languages	Critical Technical Skills for Creating Inbound Transaction	Critical Technical Skills for Creating Outbound Transactions
APAg/ Integration	UNIX, Windows	Point-to- Point	Own Language	APAg Tool, Z Tables	APAg Tool, Z-Tables
Open Data Access	All	Point-to- Point	VB	Custom code or third-party application (queries only)	

Other Industry Standard Support

PeopleSoft EnterpriseOne has a media object function that supports other industry standard functions, such as:

- Object Linking and Embedding (OLE) for the exchange of different data types
- Dynamic Data Exchange (DDE) for static and dynamic links across applications
- Binary Large Object (BLOB) for media object attachments within applications
- Extended Messaging API (MAPI) for message exchange across differing mail and groupware applications

See Also

- ❑ *Media Object Attachments* in the *Foundation Guide*
- ❑ *Media Objects and Imaging* in the *System Administration Guide*
- ❑ *Messages and Queues* in the *Foundation Guide*

Business Function Calls

A business function is an encapsulated set of business rules and logic that can be reused by multiple applications. Business functions provide a common way to access the PeopleSoft EnterpriseOne database. A business function accomplishes a specific task. Master business functions provide the logic and database calls necessary to extend, edit, and commit the full transaction to the database. Third-party applications can use master business functions for full EnterpriseOne functionality, data validation, security, and data integrity.

You can use master business functions to update master files (such as Address Book Master and Item Master) or to update transaction files (such as sales orders and purchase orders). Generally, master file master business functions, which access tables, are simpler than transaction file master business functions, which are specific to a program. Transaction master business functions provide a common set of functions that contain all of the necessary default values and editing for a transaction file. Transaction master business functions contain logic that ensures the integrity of the transaction being inserted, updated, or deleted from the database.

For interoperability, master file master business functions can be used instead of table input/output, which allows you to perform updates to related tables using the master business function instead of table event rules. Multiple records are not used; instead, all edits and actions are performed with one call.

The following models support business function calls:

- XPI
- Connectors
 - Dynamic Java Connector
 - Java Connector
 - COM Connector
- Messaging Adapters
 - Adapter for MQSeries
 - Adapter for MSMQ
- Batch Interfaces
 - Interface Tables
 - Electronic Data Interchange (EDI)
 - Table Conversions

See Also

- *Business Functions* in the *Development Tools Guide* for information about creating and using business functions

Finding the Right Business Function

Business functions are core for interoperability with PeopleSoft EnterpriseOne. If you build custom integrations to interoperate with EnterpriseOne, you will need to know which business functions to call and how to call those business functions. You can use existing business functions, modify an existing business function, or create a custom business function. If you are creating a custom business function, PeopleSoft suggests that you find an existing business function that is similar to what you want to accomplish and use the existing business function as a model. To find an existing business function, try the following:

- Review the online business function and API documentation.
- Create business function documentation.
- Use the following EnterpriseOne tools:
 - Object Management Workbench
 - Cross Reference Facility
 - Autopilot Analyzer
 - Debug Application

Caution

When an update or ESU affects business functions, you might be required to modify your custom integration.

Reviewing Online API and Business Function Documentation

The online reference guide provides detailed information about PeopleSoft EnterpriseOne business functions and APIs that you can use in your custom integrations. The online reference guide groups business functions as follows:

Master Business Functions	A collection of business functions that provide the logic and database calls necessary to extend, edit, and commit the full transaction to the database. The design of master business functions allows them to be called asynchronously and to send coded error messages back to calling applications.
Major Business Functions	Components that encapsulate reusable logic common to many applications, such as date editing routines and common multicurrency functions.
Minor Business Functions	Components that perform complex logic for a specific instance or single application. Minor business functions are used in EnterpriseOne for processing that cannot be accomplished efficiently in event rules or for logic that might be needed in multiple places within a single application.

See Also

- ❑ The *Tools APIs Reference* in the online *Technical Reference Documentation* on the Knowledge Garden for business functions and APIs

Creating Business Function Documentation

Business function documentation explains what individual business functions do and how to use each business function. You can generate information for all business functions, groups of business functions, or individual business functions. The documentation for a business function includes information such as:

- Purpose
- Parameters (the data structure used)
- Explanation of each individual parameter that indicates input/output required and explanation of return values
- Related tables (the table accessed)
- Related business functions (business functions called from within the functions itself)
- Special handling instructions

See Also

- ❑ *Business Function Documentation* in the *Development Tools Guide* for information about creating, generating, and viewing business function documentation

Using an Application as a Model

If you can find a PeopleSoft EnterpriseOne application that is similar to what you need to do, you can use that application as a model. The EnterpriseOne Cross Application Development Tools menu (GH902) provides several tools that you can use to determine what business functions an EnterpriseOne application uses and how the business function is used in the application. From the Cross Application Development Tools menu, you can access the following:

- Object Management Workbench
- Cross Reference Facility
- AutoPilot Analyzer
- Debug Application

Object Management Workbench

You can use the Object Management Workbench (OMW) to search for the business function object and then review the C code.

See Also

- ❑ *Object Management Workbench* in the *Development Tools Guide* for more information about using the Object Management Workbench

Cross Reference Facility

You can use the Cross Reference Facility to identify each instance for which a business function is used. The Cross Reference program (P980011) is on the Cross Application Development Tools menu (GH902).

See Also

- ❑ *Cross Reference Facility* in the *Development Tools Guide* for more information about searching for objects, including business functions

AutoPilot Analyzer

You can use the PeopleSoft EnterpriseOne AutoPilot Analyzer tool to capture and display information about the inputs and outputs of every business function and database API call made for an EnterpriseOne application. To use the AutoPilot Analyzer tool, you must first capture the data, and then you must import the data to the AutoPilot Analyzer tool.

See Also

The following topics in the *Analyzer Tool Guide*:

- ❑ *Capturing Data for Analyzer Tool* for information about capturing data
- ❑ *Importing Test Results* for information about displaying data that you previously captured

Debug Application

Another option that you might consider for understanding a PeopleSoft EnterpriseOne application is to run an EnterpriseOne debugger. You can run the Event Rules Debugger to obtain named event rule and table event rule information for an EnterpriseOne application. You can use Microsoft Visual C++ to debug business functions that are written in C. You can use these two tools together.

See Also

See the following topics in the *Development Tools Guide*:

- ❑ *Working with the Event Rules Debugger*
- ❑ *Debugging Business Functions Using Microsoft Visual C++*

XML Solution

The PeopleSoft EnterpriseOne XML solution supports well-formed XML documents. This XML solution supports UTF8 and UTF16 Unicode standards for inbound and outbound information. Outbound information is supported in UTF8 Unicode. Inbound and outbound information is supported in UTF16 Unicode. The EnterpriseOne XML solution includes:

XMLDispatch	Provides a single point of entry for all XML documents coming into EnterpriseOne and for EnterpriseOne responses.
XML Transformation System (XTS)	Transforms an XML document that is not in the EnterpriseOne format into an XML document that can be processed by EnterpriseOne, and then transforms the EnterpriseOne response back to your XML format.
XML CallObject	Allows you to call business functions.
XML Transaction	Allows you to use a predefined transaction type (such as JDEPOIN) to send information to or request information from EnterpriseOne. XML transaction uses EnterpriseOne interface table functionality.
XML List Kernel	Allows you to request and receive EnterpriseOne database information in chunks.
XML Service Kernel	Allows you to request events from one EnterpriseOne system and receive a response from another EnterpriseOne system.

Some of the benefits of using XML are as follows:

- Models using XML are scalable, allowing multiple connections to be opened.
- XML can be used with EnterpriseOne messaging adapters, providing a reliable connection and acknowledging operations.
- XML exposes business functions and interface tables.
- XML can be used to aggregate business function calls into one document, which reduces network traffic.
- XML can manage session creation, validation, and tracking.

If you can create XML documents on your interoperability server, you can use XML for your interoperability solution. XML callObject, XML List, and XML transactions capabilities can be used with the following EnterpriseOne models:

- XPI
- EnterpriseOne Connectors
 - Dynamic Java Connector
 - Java Connector
 - COM Connector

- EnterpriseOne Messaging Adapters
 - EnterpriseOne Adapter for MQSeries
 - EnterpriseOne Adapter for MSMQ

See Also

- ❑ *World Wide Web Consortium (W3C)* XML home page <http://www.w3.org/XML/> for general information about XML

The following topics in the *Interoperability Guide*:

- ❑ *Interoperability Overview* for information about the PeopleSoft EnterpriseOne models and capabilities that support these models
- ❑ *Business Function Calls* for information about choosing which business function to use
- ❑ *XML* for information about XML CallObject, XML Transaction, and XML List features
- ❑ *Events* for information about Z events, real-time events, and XAPI events

Formatting an XML Document

When you send an XML document to PeopleSoft EnterpriseOne for processing, the document must be in the XML format defined by EnterpriseOne. Once the document reaches the EnterpriseOne server, the system processes the document based on the document type. All XML documents must contain the following elements:

- One of the following types:
 - jdeRequest Type
 - jdeResponseType
- Establish Session
- Expire Session
- Terminate Session

In addition, you can use the following optional elements:

- Explicit Transaction
- Implicit Transaction
- Prepare/Commit/Rollback

Type Element

The type element, which can be `jdeRequest` or `jdeResponse`, is the root element for all request documents coming into the XML infrastructure. This element contains basic information about the execution environment. The following attributes form the `jdeRequest` and `jdeResponse` type element:

Type	Specifies the type of XML document request. Depending on the operation to be performed, the <code>jdeRequest</code> type can be one of the following: • <code>Callmethod</code> • <code>List</code> • <code>Trans</code> • <code>xapicallmethod</code> The <code>jdeResponse</code> type indicates an XML document coming from another PeopleSoft EnterpriseOne system. The operation for <code>jdeResponse</code> is <code>realTimeEvent</code>. Note The <code>xapicallmethod</code> and <code>realTimeEvent</code> types are discussed in the Events section of this document.
User	Specifies the user name for user identification and validation.
Pwd	Specifies the user password for user identification and validation.
Environment	Specifies the system environment.
Session	Specifies the session ID. This attribute is optional.
Sessionidle	Specifies the session timeout time. This attribute is optional.

Establish Session

You establish a session by setting the session attribute of the standard `jdeRequest` element. When the session attribute is an empty string, a new session is started. On the server, the `SessionManager` singleton class creates a new instance of a session object given the user name, password, and environment name. The session can be reused before it expires to avoid the overhead of session initialization. You can specify the session ID in the session attribute for an already established session in an earlier request.

```
<?xml version='1.0' ?>
<jdeRequest type='callmethod' user='steve' pwd='xyz'
environment='prod' session='' sessionidle='1800'>
...
</jdeRequest>
```

Note

If you do not want to start a new session, then remove the `session=''` tag. The above example is for starting a new session.

Expire Session

Session expiration is addressed by the sessionidle attribute of the standard jdeRequest element. This attribute, when given on a session creation request, specifies the amount of time in seconds that this session is allowed to be idle. If the SessionManager determines that a session has not had any requests processed in this amount of time, it terminates the session and frees all associated resources. The session idle default is 30 minutes. The session idle time is defined in the XML document.

```
<?xml version='1.0'?>
<jdeRequest type='callmethod' user='steve' pwd='xyz'
environment='prod' session='' sessionidle='1800'>
...
</jdeRequest>
```

Explicit Transaction

Explicit database transactions are supported by another element, the startTransaction tag. The startTransaction tag specifies whether transactions are to be manually or automatically committed. The startTransaction tag element is an empty element, meaning that all of its information is in the attributes.

```
<?xml version='1.0'?>
<jdeRequest type='callmethod' user='steve' pwd='xyz' environment='prod' session=''>
<startTransaction trans='t1' type='manual'/>
</jdeRequest>
```

Implicit Transaction

An XML request is included in a transaction set when the name of a transaction set is referenced in its trans attribute. Implicit start transactions can be included in the request by specifying the name of a transaction set that has not previously been created. For an implicit start, the transaction set will be a manual commit set.

```
<?xml version='1.0'?>
<jdeRequest type='callmethod' user='steve' pwd='xyz' environment='prod' session=''>
<callMethod name='myfunc' app='P42101' trans='t1'>
<params>
<param name='CostCtr'> 1001</param>
</params>
</callMethod>
</jdeRequest>
```

Prepare/Commit/Rollback

Manual transaction sets can be committed or rolled back. As part of a two-phase commit, they can be prepared to commit. Prepare, commit, and rollback requests to the database are made by using the `endTransaction` element. The transaction set is identified by the *trans* attribute. The *action* attribute indicates the action to take on the transaction set. The value can be *prepare*, *commit*, or *rollback*. This element is always an empty element, as shown by the forward slash.

PeopleSoft recommends that you manage the session ID when doing manual commits and terminate the session after the transaction is complete.

```
<?xml version='1.0'?>
<jdeRequest type='callmethod' user='steve' pwd='xyz' environment='prod' session=''>
<endTransaction trans='t1' action='commit'/>
</jdeRequest>
```

Note

If `startTransaction` and `endTransaction` are in separate documents, one of the following occurs:

- The session attribute is not sent in the second document. In this case, the user, password, and environment are used to match the previous session.
 - The session number from the response of the first document is sent in the session attribute of the documents associated with the same transaction.
-

Terminate Session

Session termination is done by submitting an XML document to explicitly terminate the session. You must specify the session to be terminated in the `jdeRequest` element tag.

```
<?xml version='1.0' ?>
<jdeRequest type='callmethod' user='steve' pwd='xyz' environment='prod'
session=5665.931961929.454'>
<endSession/>
</jdeRequest>
```

Standards for Creating XML Documents

In addition to ensuring that your XML documents have the required format elements (`jdeRequest` or `jdeResponse` Type, Establish Session, Expire Session, and Terminate Session), PeopleSoft EnterpriseOne has standards for XML documents that are different from industry standards.

Decimal and Comma Separators

PeopleSoft EnterpriseOne uses the decimal and thousands separators differently than XML industry standards. The decimal and thousands separators do not depend on use profile settings, jde.ini settings, or regional settings for the computer. When you write XML documents to interface with EnterpriseOne, the decimal character "." (dot) should always be used as a decimal separator, and a comma "," should be used as a thousands separator. The purpose of the separator standards is to achieve consistent interoperability policy and to prevent data corruption.

Date Usage

Different components of the XML foundation use different format codes and APIs to format the following dates:

- to XML date
- from XML date
- to JDEDATE
- from JDEDATE

The following table explains the formats used by each XML component that is supported by PeopleSoft EnterpriseOne:

Component	Inbound		Outbound	
	Format	Outcome	Format	Outcome
XMLCallObject	F	YYYYMD	ESOSA	YYYY/MM/DD
XMLTransaction	F*	User Preference	ESOSA	YYYY/MM/DD
XMLList	B*	User Preference	NULL	User Preference

* Component ignores the format code

Configure the System Environment for XML

Before you can use XML with PeopleSoft EnterpriseOne, you must ensure that the ICU_DATA system environment variable is correctly defined on your EnterpriseOne system. If the ICU_DATA variable is not correctly defined, EnterpriseOne produces the following error message:

The default Unicode converter could not be found within the jdenet_n.log on the OneWorld Enterprise Server.

For EnterpriseOne, the ICU conversion table, icu_data.dat, is generally located in system/locale/xml. Settings for different platforms are discussed in the following paragraphs. Use the appropriate setting for your platform.

UNIX

For UNIX systems, the ICU_DATA path is based on the ICU_DATA environment variable. The UNIX PeopleSoft EnterpriseOne user login script sets the ICU_DATA environment variable to the directory location of the ICU resource file, icudata.dat. If the user login script does not set the ICU_DATA environment variable, you must define the ICU_DATA variable with a trailing slash, as follows:

```
Export ICU_DATA=$SYSTEM/locale/xml/
```

Where \$SYSTEM represents your EnterpriseOne install directory.

iSeries

For iSeries systems, the ICU_DATA path is set when the ICU 1.6 conversion function is first called by the system. The system looks up Data Area BUILD_VER in the system library for the System Directory setting. For example:

System Directory: B9_S

The system appends locale/xml to the path specified in the BUILD_VER, and then uses this path as the ICU_DATA path. You must ensure the BUILD_VER data area is properly set to reflect the system directory setting.

WIN32

For WIN32 systems, the ICU_DATA path is set when the ICU 1.6 conversion function is first called. The following logic is used on WIN32:

1. The system looks up the environment variable JDE_B9_ICU_DATA. If this environment is found, it becomes the path for the conversion files.
2. The system looks for the following section in the jde.ini file:

```
[XML]
```

```
ICUPath=<<install>>/system/locale/xml
```

If the ICUPath setting is found, it becomes the path for the conversion files.

3. If the system cannot find the ICUPath setting in the jde.ini file, the ICU_Path defaults to:

```
EXECUTABLE_DIRECTORY/./system/locale/xml
```

The EXECUTABLE_DIRECTORY must be <<install>>/system/bin32.

Based on the above logic, you usually do not need to set the JDE_B9_ICU_DATA ENVIRONMENT variable or the jde.ini file. You need to set the jde.ini ICUPath only when the location of the icudata.dat is different from system/locale/xml.

Note

The PeopleSoft EnterpriseOne client install sets the environment variable JDE_B9_ICU_DATA.

XML Transformation Service

The PeopleSoft EnterpriseOne XML transformation system (XTS) uses extensible stylesheet language (XSL) to transform XML documents to the format required by EnterpriseOne. XTS also transforms EnterpriseOne response XML documents back to the XML format of the original request.

XTS is a multi-threaded Java process that runs as an EnterpriseOne kernel process. Upon system startup, the XTS kernel library loads a Java virtual machine (JVM). Once the JVM is loaded, the server proxy is started. If you do not have JVM available on your server, you must install one. EnterpriseOne supports Java JVM version 1.3 and later.

XTS is available on all platforms that EnterpriseOne supports.

XTS Processing

When the PeopleSoft EnterpriseOne XML Dispatch kernel receives an XML document that it does not recognize, it sends the document to XTS for transformation. XTS reads the XSL, transforms the document to a format that is compatible with EnterpriseOne, and sends the document back to the XML Dispatch kernel for EnterpriseOne processing. When the EnterpriseOne response comes into XML Dispatch, XML Dispatch remembers that the document needs to be transformed from the EnterpriseOne XML format and sends the document to XTS for transformation. XTS transforms the EnterpriseOne XML document back to your original XML format and sends the document to XML Dispatch for distribution to you.

Native XML format is the XML format that is defined by EnterpriseOne and is documented in this guide. All XML documents coming into EnterpriseOne must be in native XML format. The EnterpriseOne kernel processes (such as, XML CallObject, XML trans, XML list, and so on) can only process XML documents that are in native format. As part of the XTS solution, EnterpriseOne provides a selector that determines whether a non-EnterpriseOne XML document can be transformed. A selector is code that looks at an XML document to see if it recognizes the document. If the selector recognizes the XML document, the selector is able to associate the XML document with a provided stylesheet for transformation. The selector is able to transform Version 1 XML format into EnterpriseOne native XML format. Version 1 XML format is XML format that is defined by EnterpriseOne but has been modified to be tool friendly. Native XML format uses a field name that is preceded by param name. Version 1 XML format uses just the field name.

Example: PeopleSoft EnterpriseOne Native XML Format

The following sample shows PeopleSoft EnterpriseOne native XML format:

```
<?xml version="1.0" encoding="UTF-8" ?>
<jdeRequest type="callmethod" user="JDE" pwd="JDE" environment="PRD733" session="">
  <callMethod name="GetLocalComputerId" app="MSMQ" runOnError="no">
    <params>
      <param name="szMachineKey" id="2" />
    </params>
    <onError abort="yes" />
  </callMethod>
  - <callMethod name="F4211FSBeginDoc" app="MSMQ" runOnError="no">
    <params>
      <param name="mnCMJobNumber" id="1" />
      <param name="cCMDocAction">A</param>
```

```

    <param name="cCMProcessEdits">1</param>
    <param name="szCMComputerID" idref="2" />
    <param name="cCMUpdateWriteToWF">2</param>
    <param name="szCMPProgramID">MSMQ</param>
    <param name="szCMVersion">MSMQ</param>
    <param name="szOrderType">SQ</param>
    <param name="szBusinessUnit">M30</param>
    <param name="mnAddressNumber">4242</param>
    <param name="szReference">2</param>
    <param name="cApplyFreightYN">Y</param>
    <param name="szCurrencyCode">CAD</param>
    <param name="cWKSsourceOfData" />
    <param name="cWKProcMode">1</param>
    <param name="mnWKSsuppressProcess">0</param>
  </params>
  <onError abort="yes">
    <callMethod name="F4211ClearWorkFile" app="MSMQ" runOnError="yes">
      <params>
        <param name="mnJobNo" idref="1" />
        <param name="szComputerID" idref="2" />
        <param name="mnFromLineNo">0</param>
        <param name="mnThruLineNo">0</param>
        <param name="cClearHeaderWF">2</param>
        <param name="cClearDetailWF">2</param>
        <param name="szProgramID">MSMQ</param>
        <param name="szCMVersion">ZJDE0001</param>
      </params>
    </callMethod>
  </onError>
</callMethod>
<callMethod name="F4211FSEditLine" app="MSMQ" runOnError="yes">
  <params>
    <param name="mnCMJobNo" idref="1" />
    <param name="cCMLineAction">A</param>
    <param name="cCMProcessEdits">1</param>
    <param name="cCMWriteToWFFlag">2</param>
    <param name="szCMComputerID" idref="2" />
    <param name="mnLineNo">1</param>
    <param name="szItemNo">1001</param>
    <param name="mnQtyOrdered">5</param>
    <param name="cSalesTaxableYN">N</param>
    <param name="szTransactionUOM">EA</param>
    <param name="szCMPProgramID">1</param>
    <param name="szCMVersion">ZJDE0001</param>
    <param name="cWKSsourceOfData" />
  </params>
  <onError abort="no" />
</callMethod>
<callMethod name="F4211FSEndDoc" app="MSMQ" runOnError="no">
  <params>
    <param name="mnCMJobNo" idref="1" />
    <param name="szCMComputerID" idref="2" />
    <param name="szCMPProgramID">MSMQ</param>
    <param name="szCMVersion">ZJDE0001</param>
    <param name="cCMUseWorkFiles">2</param>
  </params>

```

```

    <param name="mnSalesOrderNo" id="3" />
    <param name="szKeyCompany" id="4" />
    <param name="mnOrderTotal" id="5" />
  </params>
  <onError abort="no">
    <callMethod name="F4211ClearWorkFile" app="MSMQ" runOnError="yes">
      <params>
        <param name="mnJobNo" idref="1" />
        <param name="szComputerID" idref="2" />
        <param name="mnFromLineNo">0</param>
        <param name="mnThruLineNo">0</param>
        <param name="cClearHeaderWF">2</param>
        <param name="cClearDetailWF">2</param>
        <param name="szProgramID">MSMQ</param>
        <param name="szCMVersion">ZJDE0001</param>
      </params>
    </callMethod>
  </onError>
</callMethod>
<returnParams failureDestination="error" runOnError="yes"
successDestination="success">
  <param name="mnOrderNo" idref="3" />
  <param name="szOrderCo" idref="4" />
  <param name="mnWKOrderTotal" idref="5" />
</returnParams>
<onError abort="yes">
  <callMethod name="F4211ClearWorkFile" app="MSMQ" runOnError="yes">
    <params>
      <param name="mnJobNo" idref="1" />
      <param name="szComputerID" idref="2" />
      <param name="mnFromLineNo">0</param>
      <param name="mnThruLineNo">0</param>
      <param name="cClearHeaderWF">2</param>
      <param name="cClearDetailWF">2</param>
      <param name="szProgramID">MSMQ</param>
      <param name="szCMVersion">ZJDE0001</param>
    </params>
  </callMethod>
</onError>
</jdeRequest>

```

Example: PeopleSoft EnterpriseOne Version 1 XML Format

The following sample shows Version 1 XML format:

```

<?xml version="1.0" ?>
<intBPAPI>
  <dsControl>
    <dsLogin>
      <User>JDESVR</User>
      <Password>JDESVR</Password>
      <Environment>ADEVNIS2</Environment>
      <Session />
    </dsLogin>
  </dsControl>
</intBPAPI>

```

```

</dsLogin>
<dsAPI>
  <Noun>jdeSalesOrder</Noun>
  <Verb>Create</Verb>
  <Version>1.1</Version>
</dsAPI>
<dsTranslation>
  <InMap />
  <OutMap />
</dsTranslation>
</dsControl>
<dsData>
  <callMethod_GetLocalComputerId app="NetComm" runOnError="no">
    <szMachineKey id="2" />
    <onError_GetLocalComputerId abort="yes" />
  </callMethod_GetLocalComputerId>
  <callMethod_F4211FSBeginDoc app="NetComm" runOnError="no">
    <mnCMJobNumber id="1" />
    <cCMDocAction>A</cCMDocAction>
    <cCMPProcessEdits>1</cCMPProcessEdits>
    <szCMComputerID idref="2" />
    <cCMUpdateWriteToWF>2</cCMUpdateWriteToWF>
    <szCMPProgramID>NetComm</szCMPProgramID>
    <szCMVersion>NetComm</szCMVersion>
    <szOrderType>SQ</szOrderType>
    <szBusinessUnit>M30</szBusinessUnit>
    <mnAddressNumber>4242</mnAddressNumber>
    <szReference>2</szReference>
    <cApplyFreightYN>Y</cApplyFreightYN>
    <szCurrencyCode>CAD</szCurrencyCode>
    <cWKSSourceOfData />
    <cWKProcMode>1</cWKProcMode>
    <mnWKSuppressProcess>0</mnWKSuppressProcess>
    <onError_F4211FSBeginDoc abort="yes">
      <callMethod_F4211ClearWorkFile app="NetComm" runOnError="yes">
        <mnJobNo idref="1" />
        <szComputerID idref="2" />
        <mnFromLineNo>0</mnFromLineNo>
        <mnThruLineNo>0</mnThruLineNo>
        <cClearHeaderWF>2</cClearHeaderWF>
        <cClearDetailWF>2</cClearDetailWF>
        <szProgramID>NetComm</szProgramID>
        <szCMVersion>ZJDE0001</szCMVersion>
      </callMethod_F4211ClearWorkFile>
    </onError_F4211FSBeginDoc>
  </callMethod_F4211FSBeginDoc>
  <callMethod_F4211FSEditLine app="NetComm" runOnError="yes">
    <mnCMJobNo idref="1" />
    <cCMLineAction>A</cCMLineAction>
    <cCMPProcessEdits>1</cCMPProcessEdits>
    <cCMWriteToWFFlag>2</cCMWriteToWFFlag>
    <szCMComputerID idref="2" />
    <mnLineNo>1</mnLineNo>
    <szItemNo>1001</szItemNo>
    <mnQtyOrdered>5</mnQtyOrdered>
  </callMethod_F4211FSEditLine>
</dsData>

```

```

        <cSalesTaxableYN>N</cSalesTaxableYN>
        <szTransactionUOM>EA</szTransactionUOM>
        <szCMPProgramID>1</szCMPProgramID>
        <szCMVersion>ZJDE0001</szCMVersion>
        <cWKSsourceOfData />
        <onError_F4211FSEditLine abort="no" />
    </callMethod_F4211FSEditLine>
    <callMethod_F4211FSEndDoc app="NetComm" runOnError="no">
        <mnCMJobNo idref="1" />
        <szCMComputerID idref="2" />
        <szCMPProgramID>NetComm</szCMPProgramID>
        <szCMVersion>ZJDE0001</szCMVersion>
        <cCMUseWorkFiles>2</cCMUseWorkFiles>
        <mnSalesOrderNo id="3" />
        <szKeyCompany id="4" />
        <mnOrderTotal id="5" />
        <onError_F4211FSEndDoc abort="no">
            <callMethod_F4211ClearWorkFile app="NetComm" runOnError="yes">
                <mnJobNo idref="1" />
                <szComputerID idref="2" />
                <mnFromLineNo>0</mnFromLineNo>
                <mnThruLineNo>0</mnThruLineNo>
                <cClearHeaderWF>2</cClearHeaderWF>
                <cClearDetailWF>2</cClearDetailWF>
                <szProgramID>NetComm</szProgramID>
                <szCMVersion>ZJDE0001</szCMVersion>
            </callMethod_F4211ClearWorkFile>
        </onError_F4211FSEndDoc>
    </callMethod_F4211FSEndDoc>
    <returnParams failureDestination="error" successDestination="success"
runOnError="yes">
        <mnOrderNo idref="3" />
        <szOrderCo idref="4" />
        <mnWKOrderTotal idref="5" />
    </returnParams>
    <onError abort="yes">
        <callMethod_F4211ClearWorkFile app="NetComm" runOnError="yes">
            <mnJobNo idref="1" />
            <szComputerID idref="2" />
            <mnFromLineNo>0</mnFromLineNo>
            <mnThruLineNo>0</mnThruLineNo>
            <cClearHeaderWF>2</cClearHeaderWF>
            <cClearDetailWF>2</cClearDetailWF>
            <szProgramID>NetComm</szProgramID>
            <szCMVersion>ZJDE0001</szCMVersion>
        </callMethod_F4211ClearWorkFile>
    </onError>
</dsData>
</intBPAPI>

```

Build a Custom Selector

You can build a selector to transform your XML format into PeopleSoft EnterpriseOne native XML format. If you write a custom selector, include both request and response extensible stylesheet language transformation (XSLT) documents.

Inside the Java file, two APIs are used to select templates. Use the boolean fetchTemplates API to fetch the appropriate XSLT document for the request document. Public boolean fetchTemplates throws IXTSMTemplateSelector.TemplateFetchException, XTSEXMLParseException. The following sample shows how to use this API:

```
fetchTemplates(XTSDocument inXML, IXTSMSelectionInfo info)
```

Use the Public void fetchTemplates to fetch the appropriate XSLT document for the response document. Public void fetchTemplates throws IXTSMTemplateSelector.TemplateFetchException.

```
fetchTemplates(IXTSMSelectionInfo info)
```

Sample code showing how PeopleSoft wrote a selector is provided below.

Note

Ensure that your custom selector is accessible in the ClassPath.

XTS APIs

When you write a custom selector, you can use the following APIs to interface with PeopleSoft EnterpriseOne:

- IXTSMTemplateSelector
- IXTSMTemplateSelector.TemplateFetchException

Detailed information for these APIs is provided in the online reference guide.

Example: Creating a Selector

The following code was written by PeopleSoft EnterpriseOne to build the Version 1 XML selector:

```
File: XTSMJDETemplateSelector.java
//
////////////////////////////////////
//
// Copyright (c) 2001 J.D. Edwards World Source Company
//
// This unpublished material is proprietary to J.D. Edwards World
// Source Company. All rights reserved. The methods and techniques
// described herein are considered trade secrets and/or confidential.
// Reproduction or distribution, in whole or in part, is forbidden
// except by express written permission of J.D. Edwards World Source
// Company.
```

```

//
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
package com.jdedwards.xts.xtsm;

import com.jdedwards.xts.xtsr.IXTSRepository;
import com.jdedwards.xts.xtsr.IXTSRKey;
import com.jdedwards.xts.xtsr.XTSRException;
import com.jdedwards.xts.xtsr.XTSRInvalidKeyStringException;
import com.jdedwards.xts.xtsr.XTSRInvalidKeyFieldException;
import com.jdedwards.xts.xtsr.XTSRKeyNotFoundException;

import com.jdedwards.xts.XTSDocument;
import com.jdedwards.xts.XTSFactory;
import com.jdedwards.xts.XTSLog;
import com.jdedwards.xts.XTSConfigurationException;
import com.jdedwards.xts.XTSXMLParseException;

import com.jdedwards.xts.xtse.IXTSEngine;
import com.jdedwards.xts.xtse.IXTSECompiledProcessor;

import java.util.List;
import org.w3c.dom.*;

/**
 * This class is the PeopleSoft Template Selector. It recognizes
 * PeopleSoft EnterpriseOne standard XML documents, and returns the
 * appropriate XSL stylesheets necessary for transformation.
 */
public class XTSMJDETemplateSelector implements IXTSMTemplateSelector
{
    /** Class constructor. */
    public XTSMJDETemplateSelector()
    {
        XTSLog.trace("XTSMJDETemplateSelector()", 3);

        // get repository reference
        XTSFactory factory = XTSFactory.getInstance();
        m_repository = factory.createIXTSRepository();
    }

    /**
     * Fetch the appropriate XSLT document(s) and/or IXTSECompiledProcessors as
     * indicated by the TPT stored in the <code>info</code> parameter.
     * @param info - Selection Info which contains TPI and should be modified
     * by the selector to specify transformation information.
     * @exception IXTSMTemplateSelector.TemplateFetchException - thrown
     * if an error occurs when extracting information from the
     * inclement.
     */
    public void fetchTemplates(IXTSMSelectionInfo info)
        throws IXTSMTemplateSelector.TemplateFetchException

```

```

{
    XTSMLog.trace("XTSMJDETemplateSelector.fetchTemplates(XTSMSelectionResult)",
3);

    NodeList nodes = info.getTPIElement().getElementsByTagName(JDE_TS_XTSR_KEY);
    int numNodes = nodes.getLength();
    for(int i = 0; i < numNodes; i++)
    {
        // extract key info & create a key
        IXTSRKey key = createKeyFromNode((Element)nodes.item(i));

        // fetch the doc and add it to the list
        try
        {
            info.getXSLList().add(m_repository.fetch(key));
        }
        catch (XTSRKeyNotFoundException e)
        {
            throw new IXTSMTemplateSelector.TemplateFetchException(
                "Selected XTSRKey not found in repository: "
                + JDE_TS_XTSR_KEY);
        }
        catch (XTSRException e)
        {
            throw new IXTSMTemplateSelector.TemplateFetchException(
                "Unable to fetch the XSL document specified within '"
                + JDE_TS_XTSR_KEY +
                "' from the XTSRepository");
        }
    }
}

/**
 * Fetch the appropriate XSLT document(s)/compiled processors for
 * the given document.
 * @param inXML - the XTSDocument to try to recognize.
 * @param info - Selection Info object to be modified by selector to
 * indicate transformation information.
 * @return - <code>true</code> if the selector has recognized the
 * document and specified the appropriate selection info via
 * <code>info</code>, <code>false</code> otherwise.
 * @exception TemplateFetchException - thrown when an error occurs
 * when trying to recognize the DOM.
 * @exception XTSXMLParseException - thrown if <inXML> could not be
 * parsed.
 */
public boolean fetchTemplates(XTSDocument inXML,
                             IXTSMSelectionInfo info)
    throws IXTSMTemplateSelector.TemplateFetchException,
           XTSXMLParseException
{
    XTSMLog.trace("XTSMJDETemplateSelector.fetchTemplates(Document, Element)", 3);

    boolean recognized = false;
    Document inDOM = inXML.getDOM();

```

```

// see if a XTSR key is specified within the document:
NodeList nodeList = inDOM.getElementsByTagName(JDE_XTSR_KEY);

if (nodeList.getLength() > 0)
{
    try
    {
        // extract key info & create a key
        IXTSRKey key = createKeyFromNode((Element)nodeList.item(0));

        // add transformation path information to outElement
        createNodeChildFromKey(info.getTPIElement(), key);

        // fetch the doc and add it to the list
        info.getXSLList().add(m_repository.fetch(key));

        info.setResultXML(true);
        info.setPathInfoStored(false);

        recognized = true;
    }
    catch (XTSRException e)
    {
        throw new IXTSMTemplateSelector.TemplateFetchException(
            "Unable to fetch the XSL document specified within '"
            + JDE_XTSR_KEY +
            "' from the XTSRepository");
    }
    catch (XTSRKeyNotFoundException e)
    {
        throw new IXTSMTemplateSelector.TemplateFetchException(
            "Key specified in TPI not found in repository"
            + JDE_XTSR_KEY);
    }
}
else // no XTSR key, so look for JDE information:
{
    nodeList = inDOM.getElementsByTagName(JDE_INT_BPAPI);
    if (nodeList.getLength() != 0)
    {
        // add transformation path information to outElement
        createNodeChildFromKey(info.getTPIElement(), getVersion1toNativeKey());

        // fetch the doc and add it to the list
        info.getXSLList().add(getVersion1toNativeXSL());

        info.setResultXML(true);
        info.setPathInfoStored(true);

        recognized = true;
    }
}

return recognized;
}

```

```

/**
 * Extracts XTSRKey information from the given node, and creates an
 * instance of IXTSRKey based on that information.
 * @return - the new IXTSRKey.
 * @param element - Element that contains the key information.
 * @exception XTSMUnrecognizedElementException - thrown if the
 * Element format is unrecognized.
 */
protected IXTSRKey createKeyFromNode(Element element)
    throws XTSMUnrecognizedElementException
{
    XTSLog.trace("XTSMJDETemplateSelector.createKeyFromNode(Element)", 4);

    IXTSRKey key = null;
    boolean request = false;
    boolean response = false;
    if (element.getNodeName().equals(JDE_XTSR_KEY))
    {
        request = true;
    }
    else if (element.getNodeName().equals(JDE_TS_XTSR_KEY))
    {
        response = true;
    }
    if (request || response)
    {
        key = m_repository.createKey();
        try
        {
            String keyString = element.getAttribute(JDE_XTSR_KEY_ATTRIBUTE);
            key.setFieldsFromString(keyString);
            if (key.getFieldValue(SUBTYPE_FIELD).length() == 0)
            {
                if (request)
                {
                    key.setFieldValue(SUBTYPE_FIELD, SUBTYPE_REQUEST);
                }
                else
                {
                    key.setFieldValue(SUBTYPE_FIELD, SUBTYPE_RESPONSE);
                }
            }
        }
        catch (XTSRInvalidKeyStringException e)
        {
            throw new XTSMUnrecognizedElementException(
                "Specified '" + JDE_XTSR_KEY +
                "' element format is invalid for this XTSRepository");
        }
        catch (XTSRInvalidKeyFieldException e)
        {
            throw new XTSMConfigurationException(
                "Specified '" + SUBTYPE_FIELD +
                "' field name not supported by repository key");
        }
    }
}

```

```

    }
    }
    return key;
}

/**
 * Creates a node that contains the key fields values and appends it
 * to the given parentNode.
 * @param parentNode - Node to which the key information should be
 * appended.
 * @param key - Key information to store in the node.*/
protected void createNodeChildFromKey(Node parentNode, IXTSRKey key)
{
    XTSLLog.trace("XTSMJDETemplateSelector.createKeyFromNode(Node,IXTSRKey)", 4);

    try
    {
        IXTSRKey keyClone = key.getRepository().createKey();
        keyClone.setFieldsFromString(key.getFieldsString());

        // we don't want to store the sub type, so we will clear it here:
        keyClone.setFieldValue(SUBTYPE_FIELD, "");

        // create new node and append it to the provided element:
        Element element =
(Element)parentNode.getOwnerDocument().createElement(JDE_TS_XTSR_KEY);
        element.setAttribute(JDE_XTSR_KEY_ATTRIBUTE, keyClone.getFieldsString());
        parentNode.appendChild(element);
    }
    catch (XTSRInvalidKeyStringException e)
    {
        XTSLLog.log("Unexpected ");
        XTSLLog.log(e);
        throw new RuntimeException("Unexpected Exception: " + e.toString());
    }
}

/**
 * Returns the key of the stylesheet to use in converting
 * EnterpriseOne version 1 documents into EnterpriseOne native
 * documents.
 * @return - The key for the XSL stylesheet.
 */
protected IXTSRKey getVersion1toNativeKey()
{
    XTSLLog.trace("XTSMJDETemplateSelector.getVersion1toNativeKey()", 5);
    if (null == m_version1ToNativeKey)
    {
        try
        {
            // create standard xsl IXTSRKey:
            m_version1ToNativeKey = m_repository.createKey();
            m_version1ToNativeKey.setFieldsFromString(V1_TO_NATIVE_KEY);
        }
    }
}

```

```

        catch (XTSRInvalidKeyStringException e)
        {
            String error = "XTSRKey necessary for JDE template selection is
invalid: "
                + V1_TO_NATIVE_KEY;
            XTSLog.log(error);
            XTSLog.log(e);
            throw new XTSConfigurationException(error);
        }
    }
    return m_version1ToNativeKey;
}

/**
 * Returns the XTSDocument which contains the XSL stylesheet for
 * converting EnterpriseOne version 1 documents into EnterpriseOne
 * native documents.
 * @return - XTSDocument containing the XSL stylesheet.
 */
protected IXTSECompiledProcessor getVersion1toNativeXSL()
{
    XTSLog.trace("XTSMJDETemplateSelector.getVersion1toNativeXSL()", 5);
    if (null == m_version1ToNativeXSL)
    {
        XTSDocument xsl = null;
        Try
        {
            xsl = m_repository.fetch(getVersion1toNativeKey());
            IXTSEngine engine = XTSEngineFactory.getInstance().createXTSEngine();
            m_version1ToNativeXSL = engine.createCompiledProcessor(xsl);
        }
        catch (XTSRException e)
        {
            String error = "Unable to fetch selected template from the repository:
";
            XTSLog.log(error);
            XTSLog.log(e);
            throw new XTSConfigurationException(error + e.toString());
        }
        catch (XTSRKeyNotFoundException knfe)
        {
            String error = "Selected template XTSRKey not found in repository:";
            XTSLog.log(error);
            XTSLog.log(knfe);
            throw new XTSConfigurationException(error + knfe.toString());
        }
        catch (XTSXMLParseException pe)
        {
            String error = "Invalid XSL document in repository";
            XTSLog.log(error);
            XTSLog.log(pe);
            throw new XTSConfigurationException(error + pe.toString());
        }
    }
    return m_version1ToNativeXSL;
}

```

```

}

/** Reference to the XTSRepository */
private IXTSRepository      m_repository      = null;

/** Key for converting version 1 documents to native documents. */
private IXTSRKey            m_version1ToNativeKey = null;

/** Compiled XSL Stylesheet for converting version 1 docs to
 *  native docs. */
private IXTSECompiledProcessor m_version1ToNativeXSL = null;

/** Field Value for the XTSRKey that indicates the document is an XSL doc */
private static final String DOC_TYPE_XSL      = "XSL";

/** Element name that indicates the DOM is a Version 1 document */
private static final String JDE_INT_BPAPI     = "intBPAPI";

/** Element name that indicates the DOM is a request and not a
 *  response or error. */
private static final String JDE_REQUEST       = "jdeRequest";

/** Element name that indicates the DOM is a response */
private static final String RESPONSE          = "jdeResponse";

/** Element name that specifies an XTSRKey to use in transforming
 *  the document. */
private static final String JDE_XTSR_KEY      = "jdeXTSRKey";

/** The attribute of the <code>JDE_XTSR_KEY</code> element that
 *  stores the XTSRKey string value */
private static final String JDE_XTSR_KEY_ATTRIBUTE = "key";

/** XTSRKey field name which specifies the sub-type of the XML
 *  document. Normal values for the sub-type are defined by
 *  <code>SUBTYPE_REQUEST</code> and <code>SUBTYPE_RESPONSE</code> */
private static final String SUBTYPE_FIELD     = "SUB_TYPE";

/** XTSRKey field name which specifies the type of the XML document.
 *  The normal value is defined by <code>DOC_TYPE_XSL</code> */
private static final String FIELD_TYPE        = "TYPE";

/** XTSRKey field name which specifies the format (or owner) of the
 *  XML document. The normal value recognized by this selector is
 *  'JDE' */
private static final String FIELD_FORMAT      = "FORMAT";

/** XTSRKey field name which specifies the particular transformation
 *  that the XSL document will perform. This selector uses
 *  'V1_NATIVE' for transformations between EnterpriseOne Version 1
 *  XML documents and EnterpriseOne native version documents. */
private static final String FIELD_ID          = "ID";

/** The string representation of the XTSRKey for the XSL document to
 *  format EnterpriseOne version 1 request documents into

```

```

    * EnterpriseOne native request documents. */
    private static final String V1_TO_NATIVE_KEY          = "XSL-JDE-V1_NATIVE-
REQUEST";

    /** XTSRKey field <code>SUBTYPE_FIELD</code> value that indicates
    * the XSL document will transform jdeRequest documents. */
    private static final String SUBTYPE_REQUEST          = "REQUEST";

    /** XTSRKey field <code>SUBTYPE_FIELD</code> value that indicates
    * the XSL document will transform jdeResponse documents. */
    private static final String SUBTYPE_RESPONSE        = "RESPONSE";

    /** Element name stored within the Transformation Path Information
    * (TPI)that specifies the XTSRKey used to transform the document. */
    private static final String JDE_TS_XTSR_KEY          = "XTSJDETemplateKey";

    private static class XTSMUnrecognizedElementException
        extends IXTSMTemplateSelector.TemplateFetchException
    {
        public XTSMUnrecognizedElementException(String text)
        {
            super(text);
        }
    }
}

```

Configure the jde.ini File for XTS

The XTS Kernel must be defined in the server jde.ini file. The name of the configuration file is retrieved from the config_file system variable in the JVM, and these property settings are part of a configuration file other than jde.ini. The jde.ini file does not require any special configurations, other than defining the XTS Kernel.

The kernel settings are as follows:

```

[JDENET_KERNEL_DEF23]
krnlName=JDEXTS KERNEL
dispatchDLLName=xtskrnl.dll
dispatchDLLFunction=_JDEK_DispatchXTSMessage@28
maxNumberOfProcesses=1
numberOfAutoStartProcesses=0

```

Refer to the following table for different .dll extensions for other platforms.

	dispatch DLLName	dispatchDLLFunction
iSeries	XTSKRNL	JDEK_DispatchXTS
HP9000B	libxtskrnl.sl	JDEK_DispatchXTS
SUN or RS6000	libxtskrnl.so	JDEK_DispatchXTS

Other jde.ini File settings include the following:

```
[JDE_CG]
CLASSPATH=<install-path>\system\Classes\xalan.jar;<install-path>\system\Classes
\xerces.jar;<install-path>\system\Classes\_Kernel.jar;<install-
path>\system\Classes\XTS.jar; <install-path>\system\Classes\log4j.jar;<install-
path>\system\Classes
```

Note

Update the CLASSPATH to include your custom class file provided in the XTSTemplateSelector2 in the [XTS] section.

Do not use \$SYSTEM in your classpath. The full name of the system directory must be included.

```
[JDENET]
maxKernelRanges=24.
```

Note

For the XTS kernel to run, the maxKernelRanges setting should be at least 23.

```
[XTSRepository]
XSL-JDE-V1_NATIVE-REQUEST=ml.xsl
XSL-JDE-V1_NATIVE-RESPONSE=lm.xsl
```

Note

The first setting is the PeopleSoft EnterpriseOne default XSL to transform the request document from Version 1 to native. The second settings is the EnterpriseOne default XSL to transform the response document from native to version 1.

You can provide your XSL files either at this location or any other location as long as your selector can find and access your XSL. If you want to add your XSL files to this location, use the following naming convention, where Filename is the name of your XSL documents:

```
XSL-JDE-Filename-REQUEST=
XSL-JDE-Filename-RESPONSE=
```

```
[XTS]
XTSTemplateSelector1=com.jdedwards.xts.xtsm.XTSMJDETemplateSelector
XTSTraceLevel=2
```

Note

The XTSTemplateSelector1 setting is the EnterpriseOne default template selector for providing XSL to transform between Version 1 and native format.

You can add your custom template selector to this section. For example, your template selector setting could be defined as follows:

XTSTemplateSelector2=com.customer.CustomTemplateSelector

The XTSTraceLevel=2 setting defines the level of XTS logging.

XML Dispatch

XML Dispatch is XML-based interoperability that runs as a PeopleSoft EnterpriseOne kernel process. The XML Dispatch kernel is the central entry point for all XML documents. For incoming XML documents, XML Dispatch identifies what kind of document is coming into EnterpriseOne and sends the document to the appropriate kernel for processing. If XML Dispatch does not recognize the document, XML Dispatch sends the document to XTS to recognize and transform into native EnterpriseOne format. After XTS transforms the document, the document is sent back to XML Dispatch to be sent to the appropriate kernel for processing. For outgoing documents, XML Dispatch is able to remember whether the request document was transformed into EnterpriseOne native format. If the incoming request was transformed, then the outgoing response document is sent to XTS for transformation from native EnterpriseOne format back into the format of the original request. After XTS transforms the document, the document is sent to XML Dispatch to distribute to the originator.

The XML Dispatch kernel is able to route and load balance the XML documents. For example, if you have many XML CallObject message types coming in at once, XML Dispatch will try to instantiate a new CallObject kernel. You set up the number of instances that a kernel can have in the jde.ini file. For example, if you set the number of instances for the CallObject kernel to 5, if more than one CallObject document comes into EnterpriseOne, XML Dispatch sees that a particular kernel is busy and instantiates another one (up to 5). XML Dispatch is able to recognize new kernel definitions (such as XAPI) if the kernel is defined in the jde.ini file. You are not required to change JDENET code when new kernels are added.

XML Dispatch is available on all platforms that EnterpriseOne supports.

XML Dispatch Process

XML Dispatch receives standard JDENET messages (in the form of an XML document) from a transport driver or other jdenet_n. The communication between a transport and XML Dispatch is local inter-process communication (IPC) using JDENET APIs. The communication between XML Dispatch and XTS and between XML Dispatch and XML kernels can be either IPC or remote network using JDENET APIs.

XML Dispatch parses the XML document and sends the document to the appropriate PeopleSoft EnterpriseOne kernel for processing.

XML Dispatch Recognizers

XML Dispatch uses recognizers to determine how to handle incoming and outgoing XML documents. If XML Dispatch recognizes an incoming XML document as being in PeopleSoft EnterpriseOne native XML format, the XML document is parsed and sent to the appropriate kernel. For outgoing documents, the recognizer determines whether an EnterpriseOne XML document can be left as EnterpriseOne native XML format or whether it must be transformed.

More than one recognizer can be added to XML Dispatch to recognize different XML grammar. XML Dispatch recognizes the following types:

- jdeRequest
- jdeResponse
- jdeWorkflow

The XML Dispatch recognizer raises DocIsRecognized exception on document identification to stop further parsing.

You can write a recognizer that is able to recognize other types of XML documents. The specification for the type is configured in the jde.ini file.

XML Dispatch Transports

As part of XML Dispatch, you can write a transport. Transports communicate with external systems using mechanisms such as MQSeries, MSMQ, HTTP, TCP/IP, and so on. Transport processes must run on the same machine as XML Dispatch. To develop a custom transport to communicate with PeopleSoft EnterpriseOne, use the following APIs:

- jdeTransportInit
- jdeTransportMessagePut
- jdeTransportMessageGet
- jdeTransportDoIExit

The transport APIs assume a polling model, which means calls to put or receive messages are given without a timeout. For more information about the transport APIs, see the online reference documentation.

Configure the jde.ini File for XML Dispatch

The XML Dispatch kernel must be defined in the jde.ini file. The settings for XML Dispatch are as follows:

```
[JDENET_KERNEL_DEF22]
krnlName=XML DISPATCH KERNEL
dispatchDLLName=xmldispatch.dll
dispatchDLLFunction=_XMLDispatch@28
maxNumberOfProcesses=1
numberOfAutoStartProcesses=1
```

The above settings are for Windows 2000 and NT. If you use a different platform, use the following settings for dispatchDLLName and dispatchDLLFunction:

Platform	dispatchDLLName	dispatchDLLFunction
iSeries	XMLDSPATCH	JDEK_XMLDispatch
HP9000	libxmldispatch.sl	JDEK_XMLDispatch
SUN or RS6000	libxmldispatch.so	JDEK_XMLDispatch

XML Dispatch uses the settings in the [XMLLookupInfo] section of the jde.ini file to route XML documents to the corresponding XML kernels. Three keywords (XMLRequestN, XMLKernelMessageRangeN, and XMLKernelHostN) are used to map a pair that consists of an XML request and an XML kernel. A description of the settings in the [XMLLookupInfo] section are explained in the following table:

Setting	Purpose
XMLRequestTypeN=	Identifies the type of message to be processed.
XMLKernelMessageRangeN=	A hard-coded number that identifies the kernel message range.
XMLKernelHostNameN=	The name of the host.
XMLKernelPortN=	Value is 0 or 1. To indicate a local host, enter 0. To indicate a remote host, enter 1.
XMLKernelRplyN=	Value is 0 or 1, with 1 as the default. A value of 0 indicates no reply is required. A value of 1 indicates a reply should be returned to the originator. Note XMLKernelRplyN setting is not required for list, callmethod, and trans. The reply setting is an implied 1 by default. XMLService does not send a response, and the setting for XMLKernelReplyN should be zero (0).

Where N starts with 1, and multiple groups of these keys can be in this section.

The [XMLLookupInfo] section should have 6 groupings, as follows:

```
[XMLLookupInfo]
XMLRequestType1=list
XMLKernelMessageRange=5257
XMLKernelHostName1=local
XMLKernelPort1=0

XMLRequestType2=callmethod
XMLKernelMessageRange2=920
XMLKernelHostName2=local
XMLKernelPort2=0
```

```

XMLRequestType3=trans
XMLKernelMessageRange3=5001
XMLKernelHostName3=local
XMLKernelPort3=0

XMLRequestType4=JDEMSGWFINTEROP
XMLKernelMessageRange4=4003
XMLKernelHostName4=local
XMLKernelPort4=0
XMLKernelReply4=0

XMLRequestType5=xapicalmethod
XMLKernelMessageRange5=14251
XMLKernelHostName5=local
XMLKernelPort5=0
XMLKernelReply5=0

XMLRequestType6=realTimeEvent
XMLKernelMessageRange6=14251
XMLKernelHostName6=local
XMLKernelPort6=0
XMLKernelReply6=0

```

XML Dispatch Error Handling

XML Dispatch handles three types of errors. The following table identifies the errors and the way that XML Dispatch handles the error:

An error occurs while XML dispatch, XTS, and the XL kernel processes are exchanging data. For example, communication is broken.	XML Dispatch generates an error report, which is an XML document that describes the error.
An error occurs while the parser or XTS is processing an XML document. For example, a syntax error, an invalid request, and so on.	XML Dispatch generates an error report that is based on the error message that is generated by either the parser or XTS.
An error occurs while an XML kernel is processing an XML document. For example, the user name is invalid, the transaction is rolled back, and so on.	XML Dispatch uses XTS to transform the XML kernel generated error report when necessary.

XML Dispatch sends generated error reports to the corresponding transport process.

XML CallObject

XML CallObject is XML-based interoperability that runs as a PeopleSoft EnterpriseOne kernel process. You can also use XML CallObject with a messaging adapter. Some features of XML CallObject are as follows:

- Allows you to make business function calls to EnterpriseOne using XML documents.
- Provides business function templates, and also allows you to create your own templates.
- Allows you to call multiple business functions using a single XML document.
- Provides a simpler way of interfacing with EnterpriseOne compared to using COM or Java APIs.

XML CallObject Templates

XML CallObject provides a blank template that you can complete so that you can make CallObject requests for a given business function. You also have the option of creating your own custom XML documents.

To request an XML template for a given business function, you create an XML document that is a callMethod request type. When you make a CallObject template request, the response is the template that has information about all of the function parameters but with no data values filled in. The user, password, and session attribute values are blanked out so that you can cache the response for later use.

A CallObject template request is an exception to the convention that a jdeRequest returns a jdeResponse. Instead of data, you receive the template, which you use to make another CallMethod request. When you request a CallObject template, the request for the template is the only request that can be made in the XML document. The XML document must include the business function.

The following example illustrates a request for a CallObject template:

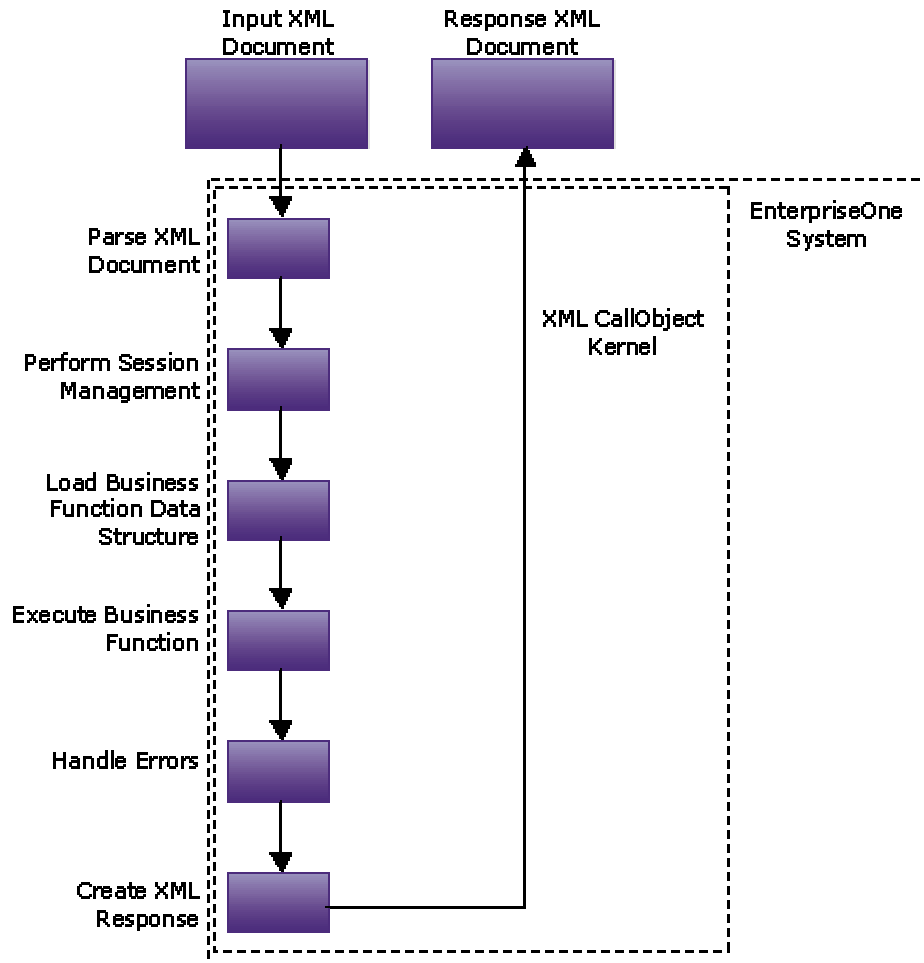
```
<?xml version='1.0' ?>
<jdeRequest type='callmethod' user='steve' pwd='xyz' en
vironment='prod' session=''>
<callMethodTemplate name='myfunc' app='P42101' />
</jdeRequest>
```

The following example illustrates a response to a CallObject template request. This response can then be filled in with the appropriate information and sent back as a request.

```
<?xml version='1.0' ?>
<jdeRequest type='callmethod' user='' pwd='' environ
ment='prod' session=''>
  <callMethod name='myfunc' app='P42101'>
    <params>
      <param name='CostCtr'></param>
      <param name='ExpDate'></param>
      <param name='Quantity'></param>
    </params>
  </callMethod>
</jdeRequest>
```

XML CallObject Process

The XML CallObject process is illustrated below:



In summary:

- The PeopleSoft EnterpriseOne server receives an XML document.
- XML CallObject processes the message by parsing the XML document.
- The session manager validates the user and password.
- Each requested business function is called separately or within requested transaction boundaries until all calls are processed.
- Output data and error messages are merged with the data from the input XML document and a new response document is created and sent to the originator.

XML CallObject Elements

Your XML document must have the following elements at the beginning of the document:

- jdeRequest Type
- Establish Session
- Expire Session

Your XML document must end with the following element:

- Terminate Session

Your XML CallObject document can also have the following optional elements:

- Call Object
- On Error Handling
- Call Object Error Handling
- Error Text
- Multiple Requests per Document
- ID/IDREF Support
- Return NULL Values

Call Object

Tags are used to call business functions on the server.

The following illustration and discussion show how to use callObject:

```
<?xml version='1.0' ?>
<jdeRequest type='callmethod' user='steve' pwd='xyz' environ
ment='prod'>
<callMethod name='myfunc' app='P42101'>
<params>
<param name='CostCtr'>1001</param>
<param name='ExpDate'>1999/10/31</param>
<param name='Quantity'>12</param>
</params>
</callMethod>
</jdeRequest>
```

The callMethod element details which function to call and in what context it is being called. The name attribute specifies which business function to call, and the app attribute allows the business function to know who is calling it.

The params and param elements define the data structure of the business function. Each param element describes one data structure member. The caller is only required to give the name attribute.

If no param element value is given for an input data structure member, then the value will be treated as if it were NULL or zero.

OnError Handling

You can add an onError element to the callMethod request to take a specific action if an error occurs. The onError tag can specify an abort attribute that specifies whether all subsequent requests should be skipped. The allowed values are *yes* or *no*. A global onError tag can be specified as a child of the jdeRequest tag, which will be executed if errors were encountered and no other onError tag with abort='yes' was executed. The global onError tag should be the last request in the document.

```
<?xml version='1.0' ?>
<jdeRequest type='callmethod' user='steve' pwd='xyz' environ
ment='prod' session=''>
  <callMethod name='myfunc' app='P42101' trans='t1' runOnError='yes'>
    <params>
      <param name='CostCtr'          1001</param>
    </params>
    <onError abort='no'>
      <endTransaction trans='t1' action='rollback'>
    </onError>
  </callMethod>
</jdeRequest>
```

Call Object Error Handling

System errors on a call object are reported in the returnCode element. The numeric code is returned in the code attribute, and the corresponding text is returned as a child text node of the returnCode element. The standard jdeCallObject return codes are used for the code attribute.

```
<?xml version='1.0' ?>
<jdeResponse type='callmethod' user='steve' pwd='xyz' environ
ment='prod' session=''>
  <callMethod name='myfunc' app='P42101' trans='t1'>
    <params>
      <param name='CostCtr'          1001</param>
    </params>
    <returnCode code='0'>Success</returnCode>
  </callMethod>
</jdeResponse>
```

Error Text

Business function error messages are returned in the errors element. Within the errors element there can be zero or more error elements containing a code attribute for the error code and a child text node containing the error text. The name attribute describes the param element that is referred to by the error.

```
<?xml version='1.0' ?>
<jdeResponse type='callmethod' user='steve' pwd='xyz' environ
ment='prod' session=''>
  <callMethod name='myfunc' app='P42101' trans='t1'>
    <params>
      <param name='CostCtr'          1001</param>
    </params>
    <returnCode code='2'>Errors</returnCode>
    <errors>
      <error code='192' name='CostCtr'>Cost Center not valid</er
ror>
    </errors>
  </callMethod>
</jdeResponse>
```

Multiple Requests per Document

Multiple requests can be included in the XML document. By default, requests are not run if there have been any errors on previous requests. If a request should be run even if errors have occurred, then the default behavior can be overridden by using the `runOnError` attribute on the request with a value of `yes`.

```
<?xml version='1.0' ?>
<jdeRequest type='callmethod' user='steve' pwd='xyz' environ
ment='prod' session=''>
<callMethod name='myfunc' app='P42101' trans='t1' runOnEr
ror='yes'>
<params>
<param name='CostCtr'>      1001</param>
</params>
</callMethod>
</jdeRequest>
```

ID/IDREF Support

ID type attributes uniquely identify, by a string value, elements in a XML document. IDREF attributes allow other elements to reference the specified element. An IDREF attribute must not be used in a document before the ID it references is defined.

A param element can specify an ID attribute so that its output value from the callMethod request will be saved and referred to later in another param element by an IDREF attribute. If a param element contains an IDREF attribute, the value of the given parameter is used as the input value for the param element. For example, the output value from referenced parameter is used instead of the value in the XML.

```
<?xml version='1.0' ?>
<jdeRequest type='callmethod' user='steve' pwd='xyz' environ
ment='prod' session=''>
  <callMethod name='myfunc' app='P42101' trans='t1' runOnEr
ror='yes'>
    <params>
      <param name='CostCtr'          1001</param>
      <param name='Company1' id='c1'></param>
      <param name='Company2' id='c2'></param>
    </params>
  </callMethod>
  <callMethod name='myfunc2' app='P42101' trans='t1' runOnEr
ror='yes'>
    <params>
      <param name='Company1' idref='c1'></param>
    </params>
    <returnParams><param idref='c2' /></returnParams>
  </callMethod>
</jdeRequest>
```

You can specify a special request tag called returnParams that can contain one or more param elements. If the param elements contain IDREF attributes, then the referenced values are copied into the response.

Return NULL Values

By default, if a parameter was not specified in the request document, it will not be returned in the response document unless its value is non-blank or non-zero. This behavior can be modified by specifying the returnNullData attribute on the callMethod element with a value of yes.

```
<?xml version='1.0' ?>
<jdeRequest type='callmethod' user='' pwd='' environment='prod'
session=''>
<callMethod name='myfunc' app='P42101' returnNullData='yes'>
<params>
<param name='CostCtr'></param>
<param name='ExpDate'></param>
<param name='Quantity'></param>
</params>
</callMethod>
</jdeRequest>
```

Configure the jde.ini File for XML CallObject

The jde.ini settings for the XML callobject kernel are as follows:

```
[JDENET_KERNEL_DEF6]
krnlName=CALL OBJECT KERNEL
dispatchDLLName=XMLCallObj.dll
dispatchDLLFunction=_XMLCallObjectDispatch@28
maxNumberOfProcesses=1
numberOfAutoStartProcesses=1
```

Refer to the following table for different .dll extensions for other platforms:

Platform	dispatchDLLName	dispatchDLLFunction
iSeries	XMLCALLOBJ	XMLCallObjectDispatch
HP9000	libxmlcallobj.sl	XMLCallObjectDispatch
SUN or RS6000	libxmlcallobj.so	XMLCallObjectDispatch

Example: CallObject Request

The following code sample shows a CallObject request:

```
<?xml version="1.0" encoding="utf-8" ?>
<jdeRequest pwd="JDE" type="callmethod" user="JDE" session=""
environment="M7333NIS2" sessionidle="1800">
  <callMethod app="XMLTest" name="AddressBookMasterMBF">
    <params>
      <param name="cActionCode">A</param>
      <param name="cUpdateMasterFile">1</param>
      <param name="mnAddressBookNumber" idref="ABNumber" />
      <param name="szSearchType">C</param>
      <param name="szAlphaName">bobs</param>
      <param name="szMailingName">Bob's Shrimp boats</param>
      <param name="szAddressLine1">One Technology Way</param>
      <param name="szPostalCode">80237</param>
      <param name="szCity">Denver</param>
      <param name="szCounty">Denver</param>
      <param name="szState">CO</param>
      <param name="szCountry">US</param>
      <param name="cPayablesYNM">N</param>
      <param name="cReceivablesYN">Y</param>
      <param name="cEmployeeYN">N</param>
      <param name="cUserCode">N</param>
      <param name="cARAPNettingY">N</param>
      <param name="jdDateEffective">01/23/2001</param>
      <param name="szProgramId">EP01012</param>
      <param name="mnAddNumParentOriginal">0</param>
      <param name="szVersionconsolidated" idref="Version" />
      <param name="szCountryForPayroll">US</param>
    </params>
  </callMethod>
</jdeRequest>
```

Example: CallObject Response

The following code sample shows a CallObject response:

```
<?xml version="1.0" encoding="UTF-8" ?>
<jdeResponse pwd="JDE" role="*ALL" type="callmethod" user="JDE"
session="2360.1049473980.6" environment="PDEVNIS2" sessionidle="1800">
<callMethod app="XMLTest" name="AddressBookMasterMBF">
  <returnCode code="0" />
  <params>
    <param name="cActionCode">A</param>
    <param name="cUpdateMasterFile">1</param>
    <param name="mnAddressBookNumber">57322</param>
    <param name="szSearchType">C</param>
    <param name="szAlphaName">bobs</param>
    <param name="szMailingName">Bob's Shrimp boats</param>
    <param name="szBusinessUnit">1</param>
    <param name="szAddressLine1">One Technology Way</param>
    <param name="szPostalCode">80237</param>
    <param name="szCity">Denver</param>
    <param name="szState">CO</param>
    <param name="szCountry">US</param>
    <param name="cPayablesYNM">N</param>
    <param name="cReceivablesYN">Y</param>
    <param name="cEmployeeYN">N</param>
    <param name="cUserCode">N</param>
    <param name="cARAPNettingY">N</param>
    <param name="cAddressType3YN">N</param>
    <param name="cAddressType4YN">N</param>
    <param name="cAddressType5YN">N</param>
    <param name="jdDateEffective" />
    <param name="szProgramId">EP01012</param>
    <param name="szVersionconsolidated">ZJDE0001</param>
    <param name="cEdiSuccessfullyProcess">0</param>
    <param name="szCountryForPayroll">US</param>
  </params>
</callMethod>
</jdeResponse>
```

Return Codes for CallObject

The following table provides XML CallObject return codes that can be returned from ThinNet APIs.

Code	Description
0	XML request OK.
1	Root XML element is not a jdeRequest or jdeResponse.
2	jdeRequest user identification unknown. Check the user, pwd and environment attributes. or A callmethod request is missing the session attribute.
3	XML parse error at line.
4	Fatal XML parse error at line.
5	Error during parser initialization, server not configured correctly.
6	Unknown parse error.
7	Request session attribute is invalid.
8	Request type attribute is invalid.
9	Request type attribute not given.
10	Request session attribute is invalid, referenced process 'processid' no longer exists.
11	jdeRequest child element is invalid or unknown.
12	Environment 'Env name' could not be initialized for user. Check user, pwd and environment attribute values.
13	jdeXMLRequest parameter invalid.
14	Connection to EnterpriseOne failed.
15	jdeXMLRequest send failed.
16	jdeXMLResponse receive failed.
17	jdeXMLResponse memory allocation failed.
99	Invalid BSFN name.

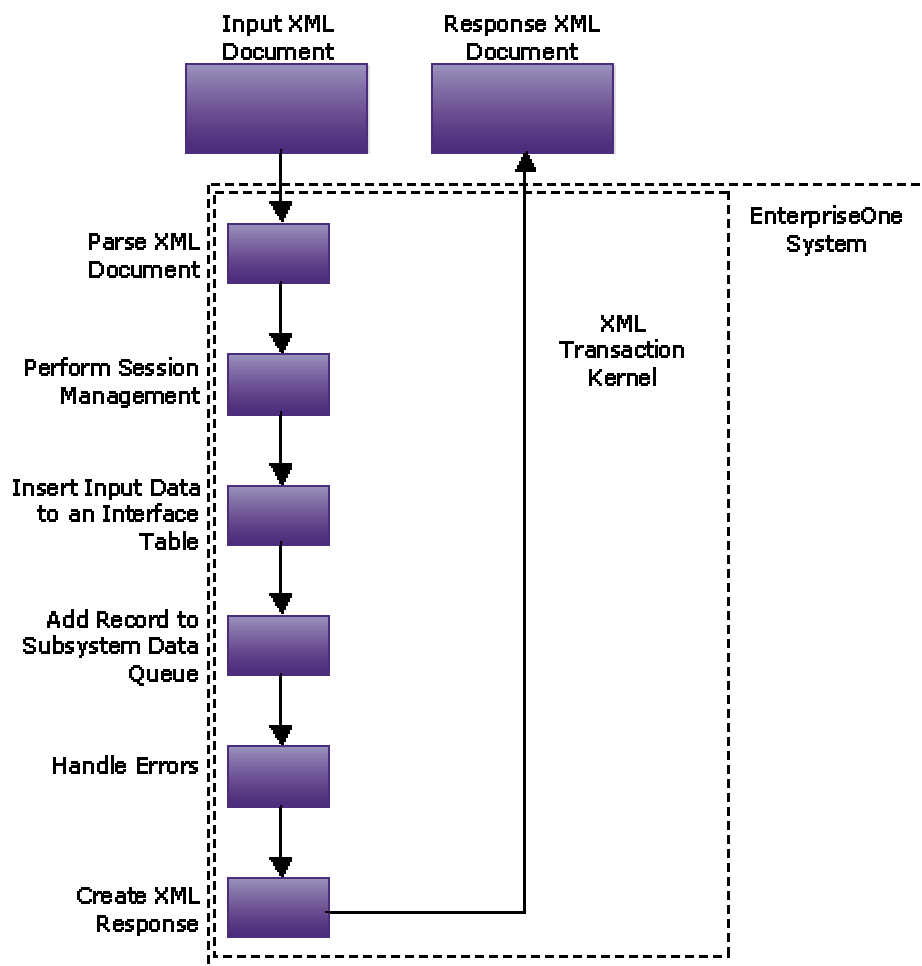
XML Transaction

XML Transaction is XML-based interoperability that runs as a PeopleSoft EnterpriseOne kernel process. You also can use XML Transaction with a messaging adapter. XML Transaction interacts with interface tables (Z tables) to update the EnterpriseOne database or to retrieve EnterpriseOne data. You can create one XML document that includes both updates to EnterpriseOne and retrieval of data from EnterpriseOne.

XML Transaction Update Process

To insert data into PeopleSoft EnterpriseOne, you use a formatted XML document. The XML document includes a predefined transaction type, such as JDEPOIN. The XML document identifies one or more EnterpriseOne interface tables and lists all of the data (data type and actual data values) to be updated.

The following illustration shows the XML Transaction update process.



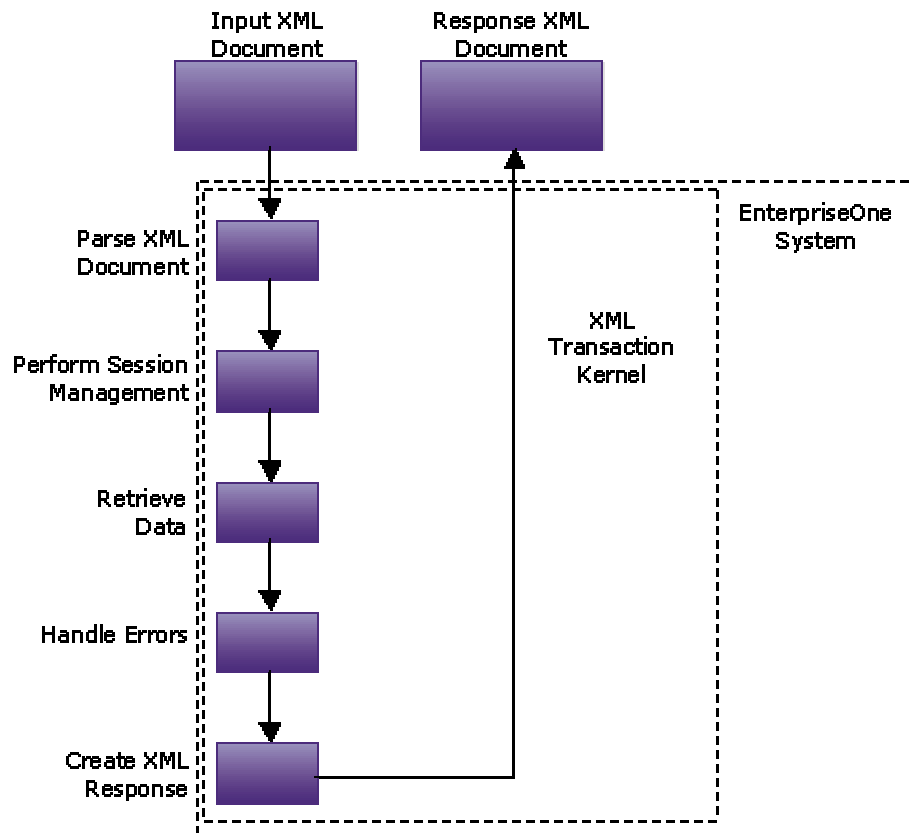
In summary:

- A request in the form of an XML document contains a list of the data for a predefined transaction type.
- XML Transaction parses the XML inbound document and inserts the data into an EnterpriseOne inbound interface table.
- XML Transaction adds a subsystem data queue record to inform the EnterpriseOne subsystem to process the added record.
- A response that indicates whether the insertion into the interface table and the subsystem data queue addition were successful is sent to the requestor.

XML Transaction Data Request Process

To request data from PeopleSoft EnterpriseOne, you use a formatted XML document. The XML document contains a transaction type, such as JDESOUT, and an index that identifies the data to be retrieved from the EnterpriseOne interface tables. You supply a template to retrieve the specific data.

The following illustration shows the XML Transaction data request and response process.



In summary:

- A request in the form of an XML document contains the transaction type and an index of the requested data.
- XML Transaction parses the XML inbound document to get the transaction type and the index.
- XML Transaction retrieves the data from EnterpriseOne and inserts the data into EnterpriseOne interface tables.
- XML Transaction creates a response in the form of an XML document. The response is comprised of the interface table data records that match the transaction type and index. The response also contains any error messages that might have occurred.

Configure the jde.ini File for XML Transaction

The jde.ini settings for the XML transaction kernel are as follows:

```
[JDENET_KERNEL_DEF15]
krnlName=XML TRANSACTION KERNEL
dispatchDLLName=XMLTransactions.dll
dispatchDLLFunction=_XMLTransactionDispatch@28
maxNumberOfProcesses=1
numberOfAutoStartProcesses=1
```

Refer to the following table for different .dll extensions for other platforms:

Platform	dispatchDLLName	dispatchDLLFunction
iSeries	XMLTRANS	XMLTransactionDispatch
HP9000	libxmltransactions.sl	XMLTransactionDispatch
SUN or RS6000	libxmltransactions.so	XMLTransactionDispatch

Example: Outbound Order Status XML Request & Response Format

The XML transaction data request is created by the outbound function and sent to the XML transaction API. The following code samples show a sales order request and response.

The following code sample shows the XML Transaction request:

```
"This format will return all columns for the sales order header (F4201Z1) and
detail lines (F4211Z1). "
```

```
<?xml version='1.0' ?>
<jdeRequest type='trans' user='user' pwd='password' environment='environment'
session='
` sessionidle='300'>
<transaction action='transactionInfo' type='JDES00OUT'>
<key>
<column name='EdiUserId'>value</column>
<column name='EdiBatchNumber'>value</column>
<column name='EdiTransactNumber'>value</column>
</key>
</transaction>
</jdeRequest>
```

The following code sample shows the XML Transaction response:

```
<?xml version='1.0' encoding='utf-8' ?>
<jdeResponse type='trans' user='user' session='session1' environment='env'>
  <transaction type='JDES00OUT' action='transactionInfo'>
    <returnCode code='0'>XML Request OK</returnCode>
    <key>
      <column name='EdiUserId'></column>
      <column name='EdiBatchNumber'></column>
      <column name='EdiTransactNumber'></column>
    </key>
    <table name='F4201Z1' type='header'>
      <column name='EdiUserId'></column>
      <column name='EdiBatchNumber'></column>
    </table>
    <table name='F4211Z1' type='detail'>
      <column name='EdiUserId'></column>
      <column name='EdiBatchNumber'></column>
    </table>
    <table name='F49211Z1' type='additionalHeader'>
      <WARNING>No record found</WARNING>
    </table>
  </transaction>
</jdeResponse>
```

XML List

XML List is XML-based interoperability that runs as a PeopleSoft EnterpriseOne kernel process. XML List provides List/GetNext functionality that allows you to collect a list of records from EnterpriseOne. XML List is built on the EnterpriseOne table conversion (TC) engine. XML List takes an XML document as a request and returns an XML document with the requested data. A list can represent data in a table, a business view, or data from a table conversion. Using data from a table conversion allows you to use multiple tables. By sending an XML document, you can retrieve metadata for a list, create a list, retrieve a chunk of data from a list, or delete a list. You can send the request through JDENet or third-party software to perform any of the following operations:

- Create List
- Get Template
- Get Group
- Delete List

XML List provides both trivial and non-trivial List/GetNext APIs. A trivial List/GetNext API performs simple gets such as selecting data from a single table. A non-trivial API uses additional functionality such as event rules. Each non-trivial List/GetNext BPAPI must have a table conversion designed for it. The data selection and data sequencing can be defined in an XML request at runtime.

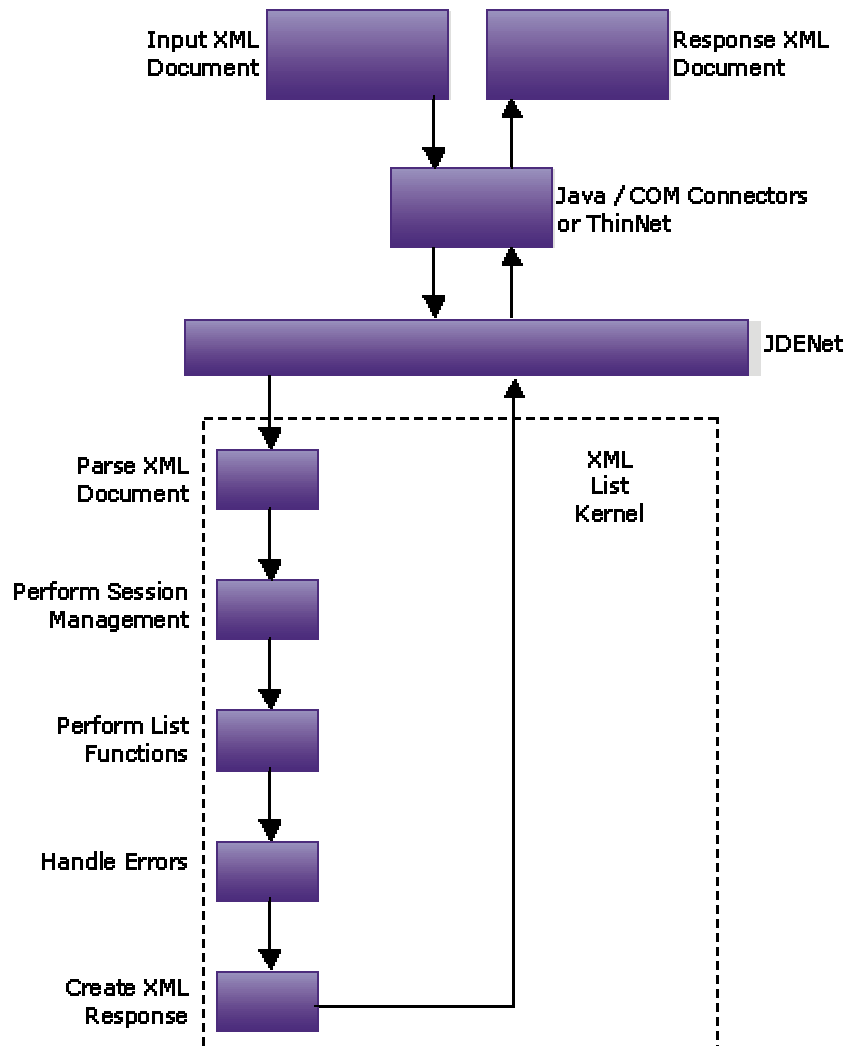
XML List provides a list-retrieval engine that allows you to create an XML data file in the system repository and then retrieve the data in small chunks.

List-Retrieval Engine Table Conversion Wrapper

A list-retrieval engine is an optimized database engine that provides and manages access to XML repository files. Each XML list repository file is a pair of index and data files with *.idb and *.ddb extensions. The .idb file keeps an index generated on a data file, and the .ddb file keeps data generated by the table conversion engine. TCWrapper is a system module that aggregates list-retrieval and list-processing APIs from TCEngine and list-retrieval engine and provides a uniform access to the data for XML List.

XML List Process

The following illustration shows the XML List process for both a trivial and non-trivial XML List request:



In summary:

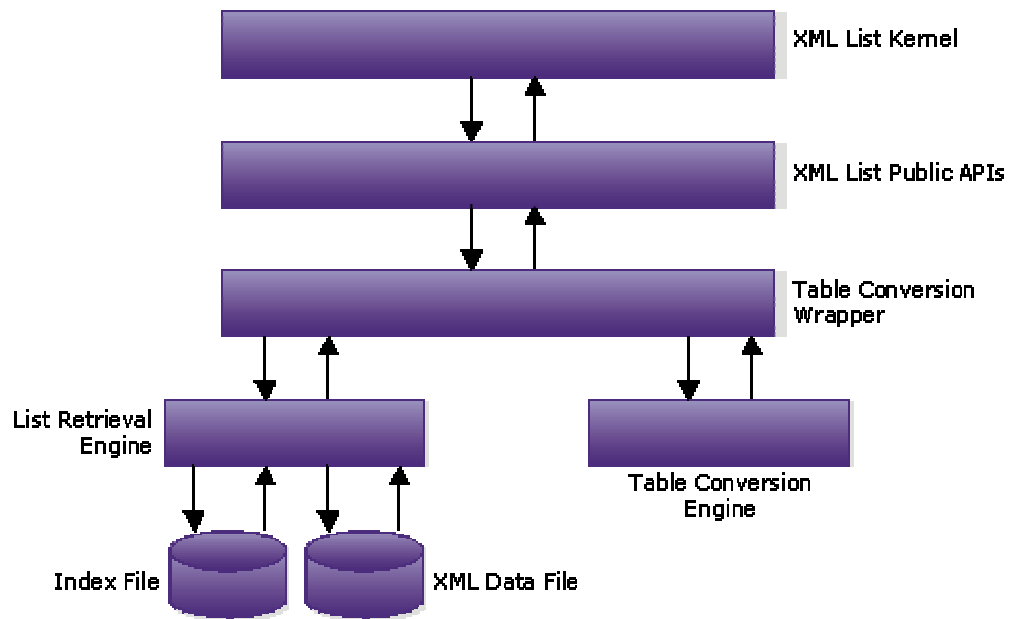
- JDENet receives the XML document.
- JDENet passes the XML Document to the XML List kernel.
 - If the request is for CreateList or GetTemplate, XML List creates a session.
 - If the request is a trivial request, XML List retrieves the data and creates a response message to send to the requestor.
 - If the request is a non-trivial request, XML List kernel passes the request to the appropriate API:
 - GetTemplate
 - CreateList
 - GetGroup
 - DeleteList
- A table conversion wrapper processes data retrieved as a result of a non-trivial request. The table conversion wrapper aggregates list-retrieval and list-processing APIs from the table conversion engine and the list-retrieval engine to provide a uniform access to the data.

XML List Requests

You can make any of the following requests using XML List:

GetTemplate	Send a request to retrieve metadata information for a list so that data selection and data sequencing can be added to the CreateList request.
CreateList	Send a request with TC/Table name along with data selection and sequencing. The response is an XML document that has a handle and size associated with the created list in the repository.
GetGroup	Send a request to retrieve data from the generated list by the previous CreateList request. GetGroup passes the handle value and range of records to be retrieved.
DeleteList	Send a request to delete a list from the repository.

The following illustration shows the various components involved in list operations:



You can use an XML List request to perform the following actions:

- Create a list.
- Retrieve data from a list.
- Delete a list.
- Get column information for a list.

Creating a List

The following example illustrates using CreateList for an XML request with the TC Name/Table Name, and data selection and sequencing. It returns an XML response with a handle associated with the created list.

```
<?xml version="1.0"?>
<jdeRequest type="list" user="JDE" pwd="JDE" environment="PRODHP01" session=""
sessionidle="">
  <ACTION TYPE="CreateList">
    <TC_NAME VALUE="" />
    <TC_VERSION VALUE="" />
    <FORMAT VALUE="UT" />
    <RUNTIME_OPTIONS>
      <DATA_SELECTION>
        <CLAUSE TYPE="WHERE">
          <COLUMN NAME="" TABLE="" INSTANCE="" ALIAS="" />
          <OPERATOR TYPE="EQ" />
          <OPERAND>
            <COLUMN NAME="" TABLE="" INSTANCE="" ALIAS="" />
            <LITERAL VALUE="" />
            <LIST>
              <LITERAL VALUE="" />
            </LIST>
            <RANGE>
              <LITERAL_FROM VALUE="" />
              <LITERAL_TO VALUE="" />
            </RANGE>
          </OPERAND>
        </CLAUSE>
      <CLAUSE TYPE="OR">
        <COLUMN NAME="" TABLE="" INSTANCE="" ALIAS="" />
        <OPERATOR TYPE="EQ" />
        <OPERAND>
          <COLUMN NAME="" TABLE="" INSTANCE="" ALIAS="" />
          <LITERAL VALUE="" />
          <LIST>
            <LITERAL VALUE="" />
          </LIST>
          <RANGE>
            <LITERAL_FROM VALUE="" />
            <LITERAL_TO VALUE="" />
          </RANGE>
        </OPERAND>
      </CLAUSE>
    </DATA_SELECTION>
    <DATA_SEQUENCING>
      <DATA SORT="ASCENDING">
        <COLUMN NAME="Product Code" TABLE="F0004" INSTANCE="" ALIAS="" />
      </DATA>
    </DATA_SEQUENCING>
  </RUNTIME_OPTIONS>
</ACTION>
</jdeRequest>
```

Either TC_NAME and TC_VERSION or TABLE_NAME and TABLE_TYPE must be defined in the request. TABLE_TYPE can be one of the following:

- OWTABLE
- OWVIEW
- FOREIGN_TABLE

The CLAUSE can be WHERE, OR, or AND to simulate a SQL statement.

You can specify the COLUMN NAME with any meaningful name to help recognize the real column name in the table, which should be defined in ALIAS. The values of TABLE, INSTANCE, and ALIAS should be the same as those in the XML response returned by a GetTemplate request. For example, if Column X is in the data selection, it should be <COLUMN NAME="My column" TABLE="F9999" INSTANCE="0" ALIAS="X"/> because the following information will be returned by a GetTemplate request similar to the following example:

```
<COLUMN NAME="X" ALIAS="X" TYPE="String" LENGTH="32" TABLE="F9999"
INSTANCE="0">
```

The OPERATOR uses values of EQ, NE, LT, GT, LE, GE, IN, NI, BW (between) or NB.

The OPERAND node can contain one of the following supported element types:

- Column
- Literal
- List
- Range

The following XML node shows the supported elements in the OPERAND node (in bold type). This is a template fragment and it should be used with only *one* of the supported elements.

```
<CLAUSE TYPE="WHERE" >
  <COLUMN NAME="UserDefinedCodes" TABLE="F0005" INSTANCE="" ALIAS="RT" />
  <OPERATOR TYPE="EQ"/>
  <OPERAND>
    <COLUMN NAME="" TABLE="" INSTANCE="" ALIAS="null"/>
    <LITERAL VALUE="P4"/>
    <RANGE>
    </RANGE>
  </OPERAND>
</CLAUSE>
```

The following sample XML nodes show the operator type (in bold type) and the operand (in bold type) using the different supported elements.

If the operand is a COLUMN, populate the COLUMN element. For example:

```
<CLAUSE TYPE="WHERE">
  <COLUMN NAME="DRSY" TABLE="F0005" INSTANCE="0" ALIAS="SY"/>
  <OPERATOR TYPE="EQ"/>
  <OPERAND>
    <COLUMN NAME="DRRT" TABLE="F0005" INSTANCE="0" ALIAS="RT"/>
  </OPERAND>
</CLAUSE>
```

If the operand is a LITERAL, populate the LITERAL element.

```
<CLAUSE TYPE="WHERE">
  <COLUMN NAME="DRSY" TABLE="F0005" INSTANCE="0" ALIAS="SY"/>
  <OPERATOR TYPE="EQ"/>
  <OPERAND>
    <LITERAL VALUE="08"/>
  </OPERAND>
</CLAUSE>
```

If the operand is a LIST, populate the element LIST. LIST should be used with IN or NI.

```
<CLAUSE TYPE="WHERE">
  <COLUMN NAME="DRSY" TABLE="F0005" INSTANCE="0" ALIAS="SY"/>
  <OPERATOR TYPE="IN"/>
  <OPERAND>
    <LIST>
      <LITERAL VALUE="08"/>
      <LITERAL VALUE="09"/>
    </LIST>
  </OPERAND>
</CLAUSE>
```

If the operand is a RANGE, populate the element RANGE. RANGE should be used with BW or NB.

```
<CLAUSE TYPE="WHERE">
  <COLUMN NAME="DRSY" TABLE="F0005" INSTANCE="0" ALIAS="SY"/>
  <OPERATOR TYPE="BW"/>
  <OPERAND>
    <RANGE>
      <LITERAL_FROM VALUE="08"/>
      <LITERAL_TO VALUE="10"/>
    </RANGE>
  </OPERAND>
</CLAUSE>
```

The XML response for a CreateList request is similar to the following:

```
<?xml version="1.0"?>
<jdeResponse type="list" session="5665.931961929.454">
<returnCode code="0">XMLRequest OK</returnCode>
  <ACTION TYPE="CreateList">
    <TABLE_NAME VALUE="F0005">
      <HANDLE>"1r4670001"</HANDLE>
      <SIZE>773</SIZE>
    </ACTION>
  </jdeResponse>
```

The value of HANDLE can be published and referenced in a GetGroup or DeleteList request.

Retrieving Data from a List

You can retrieve data from a list generated by a previous CreateList request by using a GetGroup request. The HANDLE, FROM VALUE, and TO VALUE can be defined in the request.

```
<?xml version="1.0"?>
<jdeRequest type="list" user="JDE" pwd="JDE" environment="PRODHPO1" >
  <ACTION TYPE="GetGroup">
    <HANDLE VALUE="1r4670001"/>
    <FROM VALUE="10"/>
    <TO VALUE="50"/>
  </ACTION>
</jdeRequest>
```

The XML response lists records falling into the range specified. The response returns the records in the list from #10 to #50 in the following format. The default FROM value is the first record and the default TO value is the last record in the list. For a GetGroup request for the whole list, no FROM and TO values need to be specified.

```
<?xml version="1.0"?>
<jdeResponse type="list">
<returnCode code="0">XMLRequest OK</returnCode>
<ACTION TYPE="GetGroup">
<HANDLE VALUE="1r4670001"/>
  <FROM VALUE="10"/>
  <TO VALUE="50"/>
  <Format name='Standard'><Column name='X'>abc</Column><Column
name='Y'>edf</Column></Format>
  00
</ACTION>
</jdeResponse>
```

Deleting a List

A list can be deleted if all GetGroup requests are done.

```
<?xml version="1.0"?>
<jdeRequest type="list" user="JDE" pwd="JDE" environment="PRODHPO1">
  <ACTION TYPE="DeleteList">
    <HANDLE VALUE="lr4670001"/>
  </ACTION>
</jdeRequest>
```

The list result defined in the HANDLE is deleted from the storage and a response with the status is returned to the caller.

```
<?xml version="1.0"?>
<jdeResponse type="list" >
<returnCode code="0">XMLRequest OK</returnCode>
  <ACTION TYPE="DeleteList">
    <HANDLE VALUE="lr4670001"/>
    <STATUS>OK</STATUS>
  </ACTION>
</jdeResponse>
```

Getting Column Information for a List

You can send a GetTemplate request to get the column information for a list so that data selection and sequencing can be added to the CreateList request. If OUTPUT is defined in the TEMPLATE_TYPE, the response is only for those columns in the XML output generated by a CreateList request based on the table conversion. For a trivial table conversion, both templates should be the same. The default template type is INPUT if no tag is specified.

```
<?xml version="1.0"?>
<jdeRequest type="list" user="JDE" pwd="JDE" environment="PRODHPO1" session=""
sessionidle="">
  <ACTION TYPE="GetTemplate">
    <TABLE_NAME VALUE="F0004"/>
    <TABLE_TYPE VALUE="OWTABLE"/>
    <TEMPLATE_TYPE VALUE="OUTPUT"/>
  </ACTION>
</jdeRequest>
```

The response for the input template lists all of the columns with alias name, type and the length of the data type, even though the length is only meaningful for the string type.

```
<?xml version="1.0"?>
<jdeResponse type="list" session="5665.931961929.454">
<returnCode code="0">XMLRequest OK</returnCode>
  <ACTION TYPE="GetTemplate">
    <TABLE_NAME VALUE="F0004"/>
    <TABLE_TYPE VALUE="OWTABLE"/>
    <TEMPLATE_TYPE VALUE="INPUT"/>
    <COLUMN Name="Address" Alias="X" TYPE="String" LENGTH="32"
TABLE="F9999" INSTANCE="0">
  </ACTION>
</jdeResponse>
```

Configure the jde.ini File for the List-Retrieval Engine

The list-retrieval engine uses a predefined folder as its system directory to keep and manage repository files. This system directory should be configured in jde.ini file as follows:

```
[LREngine]
System=C:\output
Repository_Size=20 (allocates percentage of disk free space for XML list repository)
Disk_Monitor=Yes (monitors free space on the disk)
```

Note

The engine uses the IFS file system on iSeries, so a corresponding system subsection must be set up.

The [SECURITY] section of the jde.ini file should also be configured. The DefaultEnvironment, Password, and User settings should be filled in for the engine to validate the default user and to initialize the default environment.

Configure the jde.ini File for XML List

The jde.ini settings for the XML List kernel are as follows:

```
[JDENET_KERNEL_DEF16]
krnlName=XML LIST
dispatchDLLName=xmllist.dll
dispatchDLLFunction=_XMLListDispatch@28
maxNumberOfProcesses=3
beginningMsgTypeRange=5257
endingMsgTypeRange=5512
newProcessThresholdRequest=0
numberOfAutoStartProcesses=1
```

Refer to the following table for different .dll extensions for other platforms:

Platform	dispatchDLLName	dispatchDLLFunction
iSeries	XMLLIST	XMLListDispatch
HP9000	libxmllist.sl	XMLListDispatch
SUN or RS6000	libxmllist.so	XMLListDispatch

Troubleshooting: XML Kernel

If one or more XML kernels are not working properly, use the following troubleshooting guidelines to ensure that your system is set up correctly.

- Check the kernel definition in the server jde.ini file. Also check that the library name is correct for the platform on which you are running. Check the entry function name.
- Check that the kernel is allowed to start. Check the maxNumberOfProcesses and numberOfAutoStartProcesses values for the kernel in the server jde.ini file. It is not necessary to auto start kernels. To work with a particular kernel, the allowed number of processes should be one or more.
- If you have a large number of simultaneous requests made to a particular kernel type, increase the number of allowed processes for that kernel. This will not only reduce the turnaround time for requests but also eliminate any "Queue Full" errors.
- If you are using XMLList kernel, check that the LREngine section is correctly setup in the server jde.ini file, and that the specified path exists. Also, check that the PeopleSoft EnterpriseOne user has write permission to this location.
- Check that the XML document is a well-formed XML document. To do this, use any XML editor or open the document in Microsoft Internet Explorer and check for errors.
- Check that the root of the input XML document is jdeRequest. All input XML documents should have jdeRequest as their root element.
- Check that valid user ID, password, and environment are provided in the XML document.
- Check that the request type in the XML document is correct. The allowed request types are callmethod, list, and trans for XMLCallObject, XMLList, and XMLTransaction kernels, respectively.

Z Transactions

Z transactions are non-EnterpriseOne information that is properly formatted in EnterpriseOne interface tables (Z tables) for updating to the EnterpriseOne database. Interface tables are working tables that mirror EnterpriseOne applications tables. EnterpriseOne provides predefined interface tables for some application transactions. You also can create your own interface tables as long as they are formatted according to EnterpriseOne standards.

You can process Z transactions into EnterpriseOne one transaction at a time (referred to as a batch of one), or you can place a large number of transactions into the interface table and then process all of the transactions at one time (referred to as a true batch).

You can use Z transactions with the following model types to transfer information into EnterpriseOne:

- Messaging Adapters
 - MQSeries
 - MSMQ
- Batch Interfaces
 - Interface Tables
 - Electronic Data Interchange (EDI)
 - Table Conversions
 - Advanced Planning Agent (APAg)/Integration

See Also

- *Interface Tables* in the *Interoperability Guide* for information about the format for interface tables

Processing Z Transactions

The following tasks are required for using Z transactions to transfer information to PeopleSoft EnterpriseOne:

- Name the transaction.
- Add records to the inbound interface table.
- Run an update process.
- Check for errors.
- Confirm the update (option, user supplied).
- Purge the data from the interface table.

Name the Transaction

Z transaction types are defined in user defined code 00/TT. If you create a new transaction, you must define the transaction in user defined code 00/TT. When you name a new transaction type, the name must start with JDE and can be up to eight characters in length. The following examples illustrate a proper transaction name:

- JDERR for Receipt Routing Transaction
- JDEWO for Work Order Header Transaction

Add Records to the Inbound Interface Table

When you write your transaction to the appropriate interface table, you make the information available to PeopleSoft EnterpriseOne for processing. Z transactions may be written directly to interface tables that are already in EnterpriseOne database format. The following list shows some of the ways that you can add records to the inbound interface tables:

- Create a flat file and then convert the flat file data into records in the interface table.
- Write an Application Programming Interface (API) using EnterpriseOne-published APIs to update the interface table.
- Use Electronic Data Interchange (EDI) to update the interface table.
- Place a message in an MQ Series or MSMQ messaging adapter.
- Use Structured Query Language (SQL) or stored procedures. You must be able to convert your records to the EnterpriseOne interface table format.

Caution

If you are using a flat file to add records to the EnterpriseOne interface tables, verify that a version of the Inbound Flat File Conversion (R47002C) program for the transaction you are trying to create exists.

Run an Update Process

You can process Z transactions in one of the following ways:

- Run an input batch process, which allows you to place a large number of transactions into the interface table and then process all of the transactions at one in batch mode.
- Run a subsystem job, which allows you to send transactions to PeopleSoft EnterpriseOne one at a time without having to wait for completion in order to continue processing using the subsystem.

EnterpriseOne provides input batch and input subsystem processes for some applications.

See Also

- *Interoperability Interface Table Information* in the *Interoperability Guide* for information about the input processor batch processes and input subsystem batch processes

Run an Input Batch Process

The input batch process allows you to place one or more records in an interface table and then run a UBE to process all of the records at one time. You use the Solution Explorer to initiate the input batch process for an application that supports inbound interoperability processing. When you choose the input batch program, the program displays a version list of report features. You can use an existing report version, change an existing report version, or add a report version. You can change the processing options and data selection when you use a report version. The input batch process program generates an audit report that lists the transactions that were processed, totals for the number of processed transactions, and errors that occurred during processing.

► To run the input batch process

Start the Input Batch Process for the inbound transaction type. For example, Purchase Order has an input batch process. The Interoperability Interface Table Information at the end of this section identifies programs that have an input batch process.

1. On Work With Batch Versions, review the processing options to determine how the data is processed by the inbound edit/update program.
2. Choose the version you want to run, and then click Select.
3. On Version Prompting, click any of the following to review the report feature options:
 - Data Selection
 - Data Sequencing
4. Click the Submit button to start the input batch process for the inbound transaction type.

Run a Subsystem Job

Subsystem jobs are continuous jobs that process records from a data queue and run until you terminate the job. Subsystem jobs read records one at a time for a subsystem table, retrieve information from that particular record, and run a UBE or table conversion for each record. This triggers the inbound processor batch process that processes that specific key. If required, a preprocessor runs from the inbound processor batch process to establish key information matching the interface table record to the original application record, for example, the key to a cash receipt or purchase receipt. After processing the last record, instead of ending the job, subsystem jobs wait for a specific period and then tries to retrieve a new record. For each subsystem job, multiple records can exist in the subsystem table.

You can schedule subsystem jobs.

You initiate a subsystem job in one of the following ways:

Use a business function

You can use the generic subsystem business function, Add Inbound Transaction to Subsystem Queue (B0000175), for inbound transactions. This function writes a record to the Subsystem Job Master (F986113) table to specify a batch process that needs to be awakened in the subsystem. The business function also passes keys to the subsystem data queue. The business function then starts processing the transaction.

Use the Solution Explorer

You can use the Solution Explorer to initiate the input subsystem batch process for an application that supports inbound interoperability processing. You start the subsystem job as you would a regular batch job. Unlike other batch jobs, subsystem jobs can only run on a server. Before processing, EnterpriseOne makes sure that limits for the subsystem job on the particular server have not been exceeded. If limits have been exceeded, the subsystem job will not be processed. To process your Z transaction in near real-time mode, start the subsystem when you start your system. You will need to place your request in the data queue before you write your transaction to the interface table.

Caution

Instead of ending the job after the records have been processed, subsystem jobs look for new data in the data queue. Subsystem jobs run until you terminate them.

See Also

The following topics in the *System Administration Guide*:

- ❑ *Enabling Subsystems* for information about creating and starting subsystems
- ❑ *Terminating Subsystems* for more information about stopping and ending a subsystem job
- ❑ *The Scheduler Application* for more information about scheduling subsystem jobs

► To run a subsystem job

Start the inbound processor batch process for the inbound transaction type. For example Purchase Order has an Input Subsystem Process. The Interoperability Interface Table Information at the end of this section identifies programs that have a subsystem batch process.

Review the processing options to determine how the data will be processed by the subsystem job.

1. On Work With Batch Versions - Available Versions, choose the subsystem version you want to run, and then click Select.
2. On Version Prompting, click any of the following to review the report feature options:
 - Data Selection
 - Data Sequencing
3. Click the Submit button.

Check for Errors

The input batch process uses the data in the interface tables to update the appropriate PeopleSoft EnterpriseOne application tables as dictated by the business logic. If the process encounters an error for the transaction, the record is flagged in the processor audit trail report and error messages are sent to the Employee Work Center in the form of action messages. These action messages, when invoked, call a revision application that allows you to make corrections to the interface table.

When you review the errors in the work center, you can link directly to the associated transaction in the interface table to make corrections. You use a revision application to resubmit individual corrected transactions for immediate processing, or you can correct all transaction errors and then resubmit them all at once in a batch process.

Transactions that have been successfully updated to the live files are flagged as successfully processed in the interface tables.

See Also

- ❑ *Revision Application* in the *Interoperability Guide* for information about modifying records in interface tables

Confirm the Update

This step is optional. If you use a business function, you can provide a confirmation function to alert you that a transaction you sent into the PeopleSoft EnterpriseOne system been processed. When processing is complete, EnterpriseOne calls the function specified in the request to notify you of the status of your process. The confirmation functions are written to your specifications, but you must use the EnterpriseOne defined data structure. Interoperability inbound confirmation functions are called from the inbound processor batch program through the Call Vendor-Specific Function - Inbound business function.

The confirmation function is specific to a process and must accept the following parameters:

User ID	11 characters
Batch Number	16 characters
Transaction Number	23 characters
Line Number	Double
Successfully Processed	1 characters

The first four parameters are the keys (EDUS, EDBT, EDTN, EDLN) to the processed transaction. The last full path of the library containing the function must be passed to the subsystem batch process that processes the transaction. This information is passed through the inbound transaction subsystem data structure.

After the subsystem batch process finishes processing the transaction, it calls the inbound confirmation function, passing the keys to the processed transaction and the notification about whether the transaction was successfully processed. You include logic in your function to take appropriate action based on the success or failure of the transaction.

If you create a transaction confirmation function, you can also use the function to perform any of the following tasks:

Update your original transaction	By creating a cross-reference between the original transaction and the transaction written to the interoperability table, you can access the original transaction and update it as completed or at an error status. Using the key returned to this function, you can access the transaction written to the interoperability interface table and retrieve any calculated or default information to update your original transaction.
Run other nonEnterpriseOne business processes	If your transaction is complete, you may want to run a business process that completes the transaction in the non-EnterpriseOne software.
Send messages to users	You may want to inform your users of the status of their original transactions.

Purge the Data from the Interface Table

You should periodically purge records that have been successfully updated to the PeopleSoft EnterpriseOne database from the interface tables.

See Also

The following topics in the *Interoperability Guide*:

- ❑ *Interoperability Interface Table Information* to review the Purge Batch Processes
- ❑ *Purge Interface Table Information* for more information about purging data from the interface tables

Flat Files

Flat files (also known as user-defined formats) are usually text files stored on your workstation or server, and typically use the ASCII character set. Because data in a flat file is stored as one continuous string of information, flat files do not have relationships defined for them as relational database tables do. Flat files can be used to import or export data from applications that have no other means of interaction. For example, you might want to share information between PeopleSoft EnterpriseOne and another system. If the non-EnterpriseOne system does not support the same databases that EnterpriseOne supports, then flat files might be the only way to transfer data between the two systems.

You can use flat files with the following model types:

- XPI

Note

Typically, XPI uses the File I/O Adapter for flat file processing.

- Batch Interfaces
 - Interface Tables
 - Electronic Data Interface (EDI)
 - Table Conversions

When you use flat files to transfer data to EnterpriseOne, the data must be converted to EnterpriseOne format before it can be updated to the live database. You can use EnterpriseOne interface tables along with a conversion program, electronic data interface (EDI), or table conversion to format the flat file data. You can use EDI or table conversion to retrieve EnterpriseOne data for input to a flat file.

XPI and some EnterpriseOne batch interfaces, such as the batch extraction programs, can accept flat files and parse the information to data format.

See Also

- ❑ *Data Interface* in the *Electronic Data Interchange Guide* for information about EDI
- ❑ *Setting Up a Table Conversion* in the *Table Conversion Guide* for more information about table conversions

The following topics in the *Interoperability Guide*:

- ❑ *Interface Tables* for information about creating a custom interface table
- ❑ *Interoperability Interface Table Information* for a list of processes for PeopleSoft EnterpriseOne interface tables

Flat File Formats

When you import data using PeopleSoft EnterpriseOne interface tables, the format for flat files can be user-defined or character-delimited. The following example illustrates a single database character record that has a user-defined format with five columns: Last, First, Addr, City, and Phone.

Doe	John	123 Main	Any town	5551234	← database record
Last	First	Addr	City	Phone	

The above user-defined format example is a fixed-width column format in which all of the data for each column starts in the same relative position in each row of data.

The following is an example of the same data in a character-delimited format:

“Doe”, “John”, “123 Main”, “Anytown”, “5551234”

Setup Requirements for Flat File Conversion

The format of the record in the flat file must follow the format of the interface table. This means that every column in the table must be in the flat file record and the columns must appear in the same order as the interface table. Every field in the interface tables must be written to, even if the field is blank. Each field must be enclosed by a symbol that marks the start and end of the field. Typically, this symbol is a double quotation mark ("). In addition, each field must be separated from the next field with a field delimiter. Typically, this separator value is a comma (.). However, any field delimiter and text qualifier may be used as long as they do not interfere with the interpretation of the fields. You set the processing options on the conversion program to define the text qualifiers and field delimiters. If you are receiving documents with decimal numbers, you must use a placeholder (such as a period) to indicate the position of the decimal. You define the placeholder in the User Preference table.

The first field value in a flat file record indicates the record type. In other words, the first field value indicates which interface table the conversion program should insert the record into. Record type values are defined and stored by the record type user defined code table (00/RD). The hard-coded values are:

- 1 – Header
- 2 – Detail
- 3 – Additional Header
- 4 – Additional Detail
- 5 – SDQ

6 – Address

7 – Header Text

8 – Detail Text

For example, suppose a record in the header table looked like the following (this example ignores table layout standards):

Record Type	Name	Address	City	Zip Code
1	Joe	<Blank>	Denver	80237

The record in the flat file would look like the following.

"1","Joe"," ","Denver","80237"

Note that “1” corresponds to a header record type, and the blank space corresponds to the <Blank> in the Address column.

Dates must be in the format MM/DD/YY. Numeric fields must have a decimal as the place keeper. A comma cannot be used.

Converting Flat Files

PeopleSoft EnterpriseOne supports flat file conversion on the Windows platform only. If you have a Windows platform, you can use the Inbound Flat File Conversion program (R74002C) or the Import Flat File To JDE File (B4700240) business function.

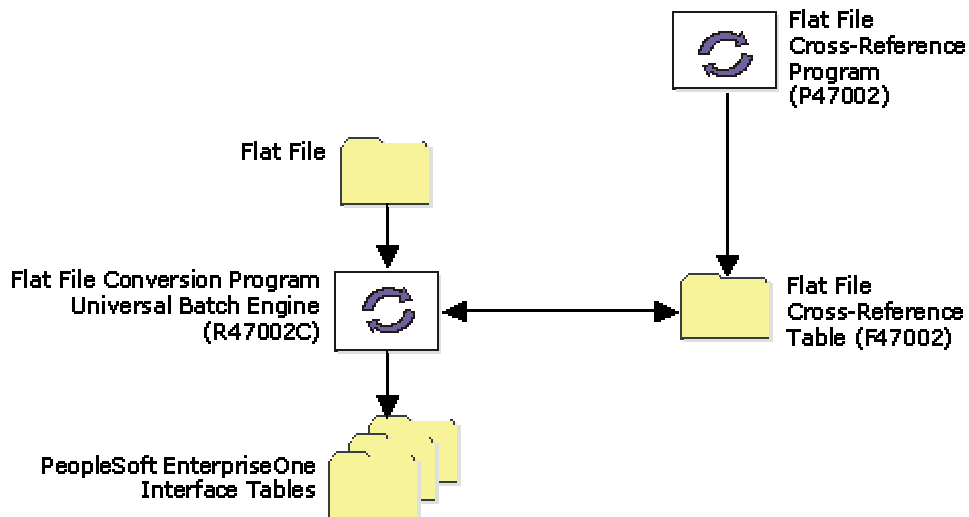
Flat File Conversion Program

If you are on a Windows platform, you can use the Inbound Flat File Conversion program (R47002C) to import flat files into PeopleSoft EnterpriseOne interface tables. You create a separate version of the Inbound Flat File Conversion program for each interface table.

Note

To use the Inbound Flat File Conversion program, you must map a drive on your PC to the location of the flat file.

The following illustration shows the process for updating EnterpriseOne interface tables using flat files.



You use the Flat File Cross-Reference program (P47002) to update the Flat File Cross-Reference table (F47002). The conversion program uses table F47002 to determine which flat file to read from based on the transaction type being received. The following list identifies some of the information that is in table F47002.

Transaction Type	The specific transaction type. The transaction type must be defined in UDC 00/TT.
Direction Indicator	A code that indicates the direction of the transaction. The direction indicator code must be defined in UDC 00/DN.
Flat File Name	Path to the flat file on your Windows PC.
Record Type	An identifier that marks transaction records as header, detail, and so on. The record type indicator must be defined in UDC 00/RD.
File Name	A valid EnterpriseOne interface table.

The conversion program uses the Flat File Cross-Reference table to convert the flat file to the EnterpriseOne interface tables. The conversion program recognizes both the flat file it is reading from and the record type within that flat file. Each flat file contains records of differing lengths based on the corresponding interface table record.

The conversion program reads each record in the flat file and maps the record data into each field of the interface table based on the text qualifiers and field delimiters specified in the flat file. All fields must be correctly formatted in order for the conversion program to correctly interpret each field and move it to the corresponding field in the appropriate inbound interface table.

The conversion program inserts the field data as one complete record in the interface table. If the conversion program encounters an error while converting data, the interface table is not updated. Because the flat file is an external object created by third-party software, the conversion program is not able to determine which flat file data field is formatted incorrectly. You must determine what is wrong with the flat file. When the conversion program successfully converts all data from the flat file to the interface tables, the conversion program automatically deletes the flat file after the conversion. After the data is successfully converted and if you set the processing option to start the next process in the conversion program, the conversion program automatically runs the inbound processor batch process for that interface table.

If the flat file was not successfully processed, you can review the errors in the Employee Work Center, which you can access from the Workflow Management menu (G02). After you correct the error condition, run R47002C again.

► To update flat file cross-reference

From an application that supports flat file conversion, open the Flat File Cross-Reference program (P47002).

1. On Work With Flat File Cross-Reference, complete the following field:
 - Transaction
2. From the Row menu, choose Define.

Record Type	Record Type Description	File Name
1	Header	F47041
2	Detail	F47042
3	Additional Header	F47044
6	Address	F4706
7	Header Text	F4714
8	Detail Text	F4715

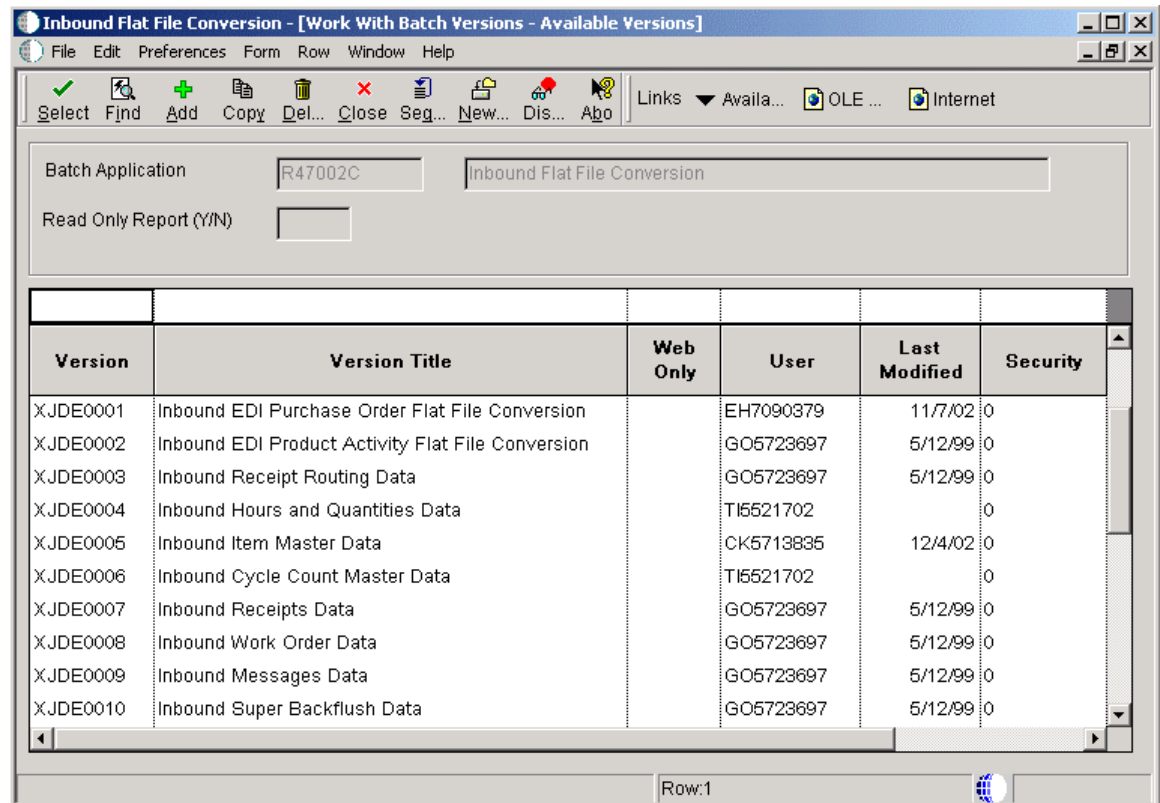
3. On Flat File Cross-Reference, complete the following field in the header area:
 - Flat File Name

4. In the detail area, change the following fields, if necessary:

- Record Type
- Record Type Description
- File Name

► To import from flat files

From an application that supports flat file conversion, open the Flat-File Conversion (R47002C) batch process.



1. On Work With Batch Versions - Available Versions, choose the program version that you want to use.
2. From the Row menu, choose Processing Options.
3. On Processing Options, click the Transaction tab and enter the transaction type that you are importing: for example, 810.
4. Click the Separators tab and enter the field delimiter character and the text qualifier character your system uses to identify fields and text.
5. Click the Process tab and enter the name of the inbound program and the version of the program to run after the Flat File Conversion program has successfully completed.
6. Click OK.
7. On Work With Batch Versions, click Select.

8. On Version Prompting, click either of the following to review the report feature options:
 - Data Selection
 - Data Sequencing
9. Click the Submit button.

Import Flat Files Using a Business Function

If you are on a Windows platform, you can use the business function named Import Flat File To JDE File (B4700240). Because of changes to server operating systems and the various ways that operating systems store files, PeopleSoft EnterpriseOne only supports the business function when run from a Windows platform. If you use the Import Flat File To JDE File (B4700240) business function, note the following constraints:

- Transaction Type and Flat File Name fields must contain data.
- Only one character is allowed in the Record Type field.
- Maximum length per line is 4095 characters.
- Maximum record types are 40.
- Every line must have a record.
- Text qualifier cannot be the same as the column delimiter.

To ensure that flat file data is properly formatted before being inserted into EnterpriseOne interface tables, the business function uses the Primary Index Detail table (F98713) to obtain primary index key information. Normally, table F98713 is located under the Default Business Data table mapping in the Object Configuration Manager. So that the business function can find table F98713, you must do one of the following:

- Map table F98713 in the system data source
- Ensure table F98713 exists in the business data source

Map Table F98713 in the System Data Source

To map the table in the system data source, you add an OCM mapping that points table F98713 to the central objects data source.

Ensure Table F98713 Exists in the Business Data Data Source

If you generate table F98713 in the business data source, you must ensure that file extensions on your PC are hidden. To hide file extensions, do the following:

1. From Start/Settings/Control Panel/Folder Options, click the View tab.
2. Turn on the Hide file extension for known file types option, and then click OK.

You must also ensure that the Flat File Name field in the Flat File Cross-Reference (F47002) table has a file extension: for example, C:\flatfiles\850.txt.

Flat File Conversion Error Messages

The following two errors might occur when you use the business function to convert flat files:

- 4363 Null Pointer
- 4377 Invalid Input Parameter

Both of the above errors are internal problems within the business function.

The following errors might occur as a result of problems with user setup or with the configurable network computing (CNC) implementation:

- 0073 Invalid File Name
- 128J (filename) Insert Failed
- 3003 Open of File Unsuccessful
- 4569 Invalid Format

Flat File APIs

In addition to the existing flat file APIs, PeopleSoft EnterpriseOne provides APIs for non-Unicode flat files. The Unicode APIs are required when flat file data is written to or read by a process outside of EnterpriseOne. The EnterpriseOne APIs, such as `jdeFWrite()` and `jdeFRead()`, do not convert flat file data, which means that the default flat file I/O for character data is in Unicode. If you use EnterpriseOne-generated flat files and the recipient system is not expecting Unicode data, you will not be able to read the flat file correctly. For example, if the recipient system is not Unicode enabled and the system is expecting data in the Japanese Shift_JIS code page (or encoding), you will not be able to read the flat file correctly. To enable the creation of the flat file in the Japanese Shift_JIS page, the application creating the flat file must be configured using the Unicode Flat File Encoding Configuration program (P93081). If the flat file is a work file or debugging file and will be written and read by EnterpriseOne only, the existing flat file APIs should be used. For example, if the business function is doing some sort of caching in a flat file, that flat file data does not need to be converted.

The EnterpriseOne conversion to Unicode uses UCS-2 encoding in memory, or two bytes per character (JCHAR), for representation of all character data. The character data passed to the output flat file APIs needs to be in JCHAR (UCS-2). The input flat file APIs will convert the character data from a configured code page to UCS-2 and return the character in JCHAR (or JCHAR string). The flat file conversion APIs allow you to configure a code page for the flat file at runtime. You use P93081 to set up the flat file code page. Flat file encoding is based on attributes such as application name, application version name, user name, and environment name.

If no code page is specified in the configuration application, the APIs perform flat file I/O passing through the data as it was input to the specific function. For example, `jdeFWriteConvert()` writes Unicode data and no conversion is performed.

► **To set up flat file encoding**

From the System Administration Tools (GH9011) menu, choose Unicode Flat File Encoding Configuration.

1. On Work With Flat File Encoding, click Add.

2. On Flat File Encoding Revisions, complete the following fields:
 - User/Role
 - Environment
 - Program ID
 - Version
 - Encoding Name
3. Click OK to save your entry.
4. Click Cancel to return to Work With Flat File Encoding.
5. On Work With Flat File Encoding, click Find.
6. To activate your new entry, choose the entry in the detail area.
7. From the Row menu, choose Change Status.

Active status is AV and inactive status is NA.
8. Click Close to return to the main menu.

See Also

- *Work with Flat File Encoding* in the *System Administration Guide* for more information about flat file encoding

Events

PeopleSoft event functionality provides an infrastructure that can capture PeopleSoft EnterpriseOne transactions in various ways and provide real-time notification to third-party software, end users, and other PeopleSoft systems, such as XPI and CRM.

EnterpriseOne notifications are called events. The EnterpriseOne event system implements a publish/subscribe model. Events are delivered to subscribers in XML documents that contain detailed information about the event. For example, when a sales order is entered into the system, the sales order information can be automatically sent to a CRM or supply chain management application to be further processed. If your system is IBM, you can use MQSeries messaging to receive events. If your system is MS, you can use MSMQ messaging to receive events. MQSeries and MSMQ provide a point-to-point interface with EnterpriseOne. EnterpriseOne supports three kinds of events:

Z Events	A service that uses interface table functionality to capture EnterpriseOne transactions and provide notification to third-party software, end-users, and other PeopleSoft systems that have requested to be notified when certain transactions occur.
Real-Time Events	A service that uses system calls to capture EnterpriseOne transactions as they occur and provide notification to third-party software, end users, and other PeopleSoft systems that have requested notification when certain transactions occur.
XAPI Events	A service that uses system calls to capture EnterpriseOne transactions as they occur and then calls third-party software, end users, and other PeopleSoft systems that have requested notification when the specified transactions occur to return a response.

The EnterpriseOne event system consists of the following modules:

- Event distributor
- Event generators
- Transport drivers

The event distributor is an EnterpriseOne kernel process called the event notification (EVN) kernel. The EVN kernel manages the subscribers and notifies them when an event occurs. The EVN kernel is shared by Z events, real-time events, and XAPI events.

Event generators are processes or libraries capable of generating events. EnterpriseOne provides three default event generators:

- Z event generator, which generates Z events
- Real-time event generator, which generates real-time events
- XAPI event generator, which generates XAPI events

Z events, real-time events, and XAPI events have slightly different XML documents.

The event distributor uses a transport driver to send events. EnterpriseOne provides a default transport driver that uses JDENET. The event distributor can also send event documents to designated MQSeries or MSMQ outbound queues using MQSeries or MSMQ transport drivers. If you use MQSeries or MSMQ transport drivers to receive events, all events that are defined in the Interoperability Event Definition table (F90701) will be sent to you.

The following model types support event generation:

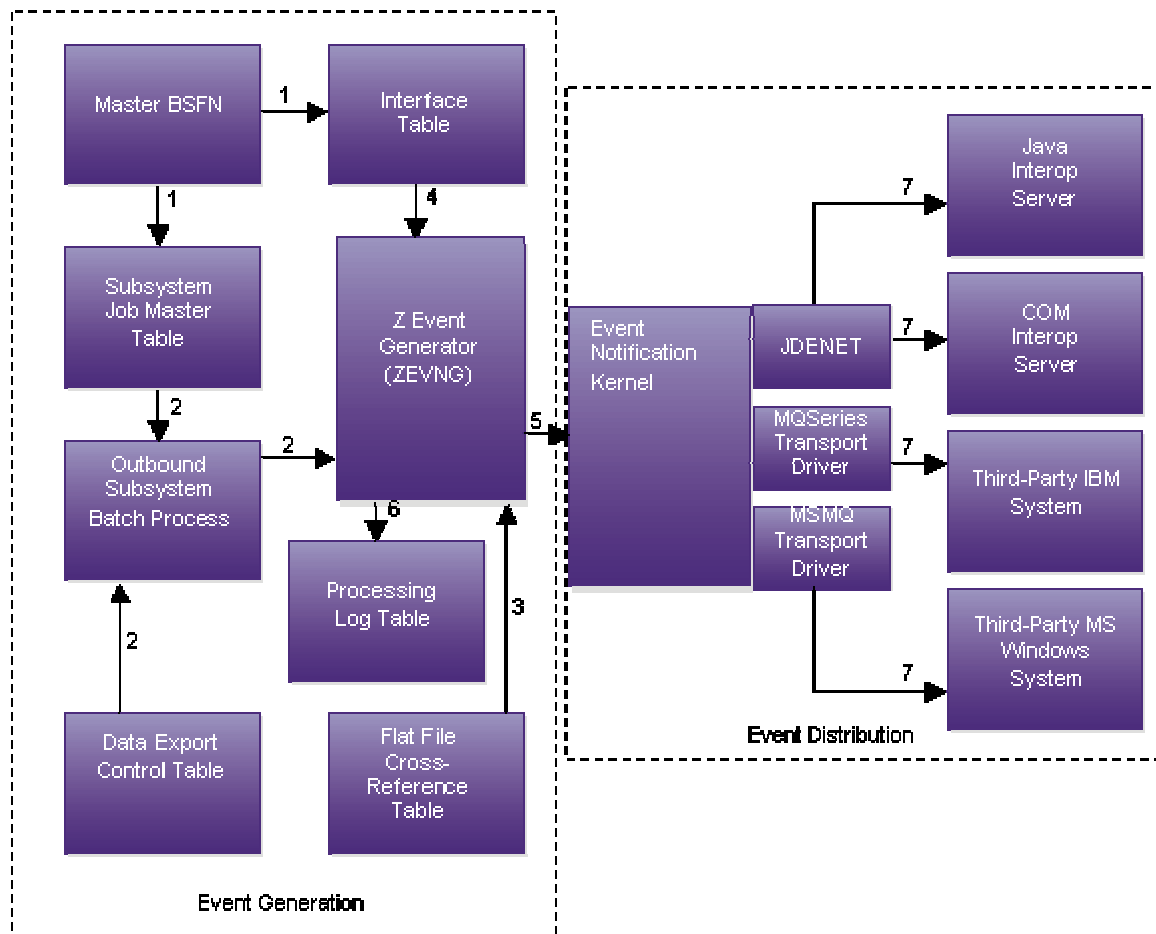
- XPI
- Connectors
 - Dynamic Java Connector
 - Java Connector
 - COM Connector
- Messaging Adapters
 - Adapter for MQSeries
 - Adapter for MSMQ

Prerequisite

- ❑ Security must be enabled for the PeopleSoft EnterpriseOne server.
- ❑ The default user under the [SECURITY] section of the EnterpriseOne server jde.ini file must have a security record (a valid EnterpriseOne user).

Z Events

A Z event is near real-time notification that an interoperability transaction has occurred. To generate Z events, PeopleSoft EnterpriseOne uses the Z event generator and the existing interface table infrastructure. You can use the existing EnterpriseOne interface tables, or you can build customized interface tables as long as the tables are created using EnterpriseOne standards. A logical representation of the processes and data for Z event generation is depicted in the following diagram and discussion:



In summary:

1. When an EnterpriseOne transaction occurs, the master business function writes the transaction information in the appropriate interface table and sends an update record to the Subsystem Job Master (F986113) table.
2. A batch process monitors the Subsystem Job Master table (F986113). When the batch process finds a W status in table F986113, it notifies the Z event generator. The batch process looks in the Data Export Control table (F0047) to determine which Z-event generator to call.
3. The Flat File Cross-Reference table (F47002) provides a cross-reference between the transaction and the interface table where the record is stored. This information is used by the Z-event generator.
4. The Z-event generator retrieves the transaction information from the interface table and converts the transaction information into an XML document using an EnterpriseOne DTD.
5. The Z-event generator sends the event (in the form of an XML document) to the event notification kernel for distribution.
6. After an event is successfully generated, the successfully generated column in the Processing Log table (F0046) is updated. A UBE purges information in the interface table based on information in the Processing Log table.
7. The event notification kernel sends the XML document to all subscribers.

Note

If you use MQSeries or MSMQ transports, system and function errors that the transport driver encounters are written to the JDE error log. The driver writes error messages and adds the error codes if available.

To generate Z events, complete the following tasks:

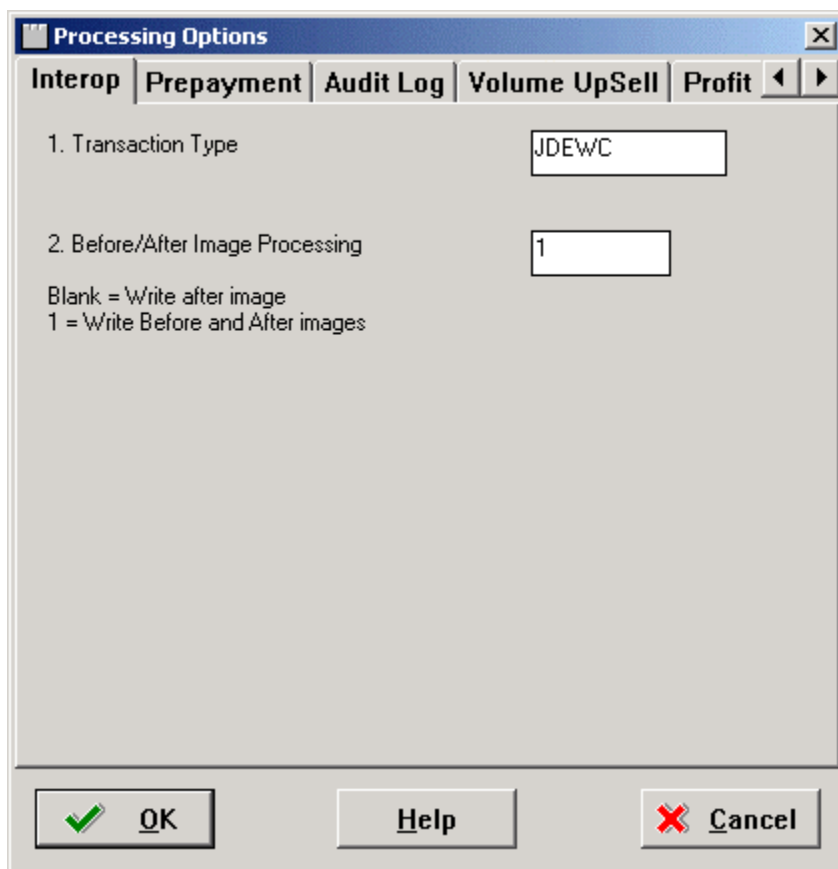
- Enable Z event processing.
- Set up the Flat File Cross-Reference (F47002) table.
- Set up the Data Export Control (F0047) table.
- Set up the Processing Log (F0046) table.
- Verify that the Subsystem Job is running.
- Purge the data from the interface table.
- Define Z events (F90701).
- Configure the jde.ini file.

Enabling Z Event Processing

You can enable or disable master business functions to write transaction information into interface tables and the Subsystem Job Master table (F986113) when a transaction occurs. All outbound master business functions that have the ability to create interoperability transactions have processing options that control how the transaction is written. The first processing option is the transaction type for the interoperability transaction. If this processing option is left blank, outbound interoperability processing will not be performed. The second processing option controls whether the before image is written for a change transaction. If this processing option is set to 1, before and after images of the transaction are written to the interface table. If this processing option is not set, then only an after image is written to the interface table.

► To enable Z event processing

1. Right-click the application that contains the processing options for the transaction's master business function.
2. Choose Prompt for Values from the menu.



The screenshot shows a dialog box titled "Processing Options" with a close button (X) in the top right corner. The dialog has five tabs: "Interop", "Prepayment", "Audit Log", "Volume UpSell", and "Profit". The "Interop" tab is currently selected. Inside the dialog, there are two input fields. The first is labeled "1. Transaction Type" and contains the text "JDEWC". The second is labeled "2. Before/After Image Processing" and contains the number "1". Below these fields, there is explanatory text: "Blank = Write after image" and "1 = Write Before and After images". At the bottom of the dialog, there are three buttons: "OK" (with a green checkmark icon), "Help", and "Cancel" (with a red X icon).

3. On Processing Options, click either the Outbound tab or the Interop tab.
4. Enter the transaction type and set the before image flag if you want before and after images written to the interface table for change transactions, and then click OK.

See Also

- ❑ *Interoperability Interface Table Information* in the *Interoperability Guide* for a list of the applications that have processing options for interoperability

Setting Up Flat File Cross-Reference

When you enable Z events, you also update the Flat File Cross-Reference (F47002) table. The transaction type that you entered in the processing option maps to table F47002 to determine which interface tables to use to retrieve the information. You use the Flat File Cross-Reference program (P47002) to update table F47002.

Setting Up Data Export Controls

The generation of outbound data is controlled through the Data Export Control (F0047) table. You use the Data Export Controls program (P0047) to update table F0047. For each transaction type and order type, you must designate the Z event generator that will process the outbound data. To send a given transaction type to more than one third-party application, you associate the transaction type with each of the individual destinations by making separate entries in table F0047 for each destination. PeopleSoft suggests that you specify the name of a third-party function that is called for each transaction as it occurs. Enough information is provided to notify you of the transaction and give you the key values so that you can retrieve the transaction.

► To set up Data Export Controls

From an application that supports event generation, choose the Data Export Controls application. Alternatively, enter P0047 in the Fast Path command line to access the Work With Data Export Controls form.

1. On Work With Data Export Controls, click Add.
2. On Data Export Control Revisions, complete the following fields:
 - Transaction
 - Order Type
3. For each detail row, enter the following:
 - Function Name
Windows NT: _CallOnUpdate@36

UNIX: CallOnUpdate

iSeries: CallOnUpdate
 - Function Library
Windows NT: EnterpriseOne Bin32 Path\zevg.dll

UNIX(HP): EnterpriseOne Bin32 Path\libzevg.sl

UNIX(AIX, SUN): EnterpriseOne Bin32 Path\libzevg.so

iSeries: EnterpriseOne Bin32 Path\ZEVG

- Enter 1 in the Execute For Add column to generate an event for an add/insert. Complete the same process as appropriate for update, delete, and inquiry.
- Enter 1 in the Launch Immediately column to launch the object from the Outbound Subsystem batch process.

This column does not affect the Outbound Scheduler batch process.

The system automatically increments the Sequence field for each line.

Processing Log Table

The Z event generator uses the Processing Log (F0046) table. Table F0046 contains the keys to the interoperability transaction along with a successfully processed column. The sequence number, transaction type, order type, function name, and function library are obtained from the Data Export Control (F0047) table. A vendor-specific record is sequentially created in table F0046 for every transaction processed by the Interoperability Generic Outbound Subsystem (R00460) UBE or the Interoperability Generic Outbound Scheduler UBE (R00461). For example, if three vendors have subscribed to a transaction using the Data Export Control table, three records are created in the Processing Log table, one record for each transaction. If the vendor-specific object successfully processed the transaction, the Processing Log record is updated with a Y in the successfully processed column. You can use the Processing Log (P0046) program to determine whether a vendor-specific object processed the interoperability transaction correctly.

A purging UBE that purges the interfaces tables runs based on information in the processing log table.

Data in the Processing Log table cannot be changed.

Verifying that the Subsystem Job is Running

When the application master business function adds a record to the Subsystem Job Master table (F986113), a subsystem job is started. Subsystem jobs are continuous jobs that process records from the Subsystem Job Master table. You should verify that the subsystem job is running.

Caution

After the records are processed, instead of ending the job, subsystem jobs look for new data in the data queue. Subsystem jobs run until you terminate them.

You can schedule subsystem jobs.

See Also

See the following topics in the *System Administration Guide*:

- ❑ *Reviewing Job Records for Subsystems* for information about verifying that the subsystem job is running
- ❑ *Terminating Subsystems* for information about stopping and ending subsystem jobs
- ❑ *The Scheduler Application* for information about scheduling jobs

Purging Data from the Interface Table

After you receive the Z event, you should purge the data from the interface table. You can enter a purge UBE in the Processing Log table to purge the interface table.

See Also

See the following topics in the *Interoperability Guide*:

- ❑ *Interoperability Interface Table Information* for a list of applications that have a purge batch process
- ❑ *Purge Interface Table Information* for information about purging data from the interface tables

Z Event Sequencing

When you define your Z events, you indicate whether the event is reliable or volatile. If you define the event as volatile, the system automatically provides event sequencing to guarantee that events are delivered in the correct order. Volatile events will be stamped using features of PeopleSoft EnterpriseOne Next Numbers.

For sequencing of Z events, the Z event generator, ZEVG, retrieves the next number from the Z event sequencing bucket and sends the number to the EVN kernel for sequencing purposes. It is important to note that PeopleSoft only guarantees the sequence for the particular type of event generator. This is due to the inherent delays involved in the Z event processing; an event that occurred earlier can get a later sequence number.

Event sequencing does affect performance. You can turn off events sequencing. You can also define a timeout value to tell the system to stop looking for a missed event when events are out of sequence. The flag and timeout settings are in the [INTEROPERABILITY] section of the jde.ini file.

Configure the jde.ini File for Z Events

To generate Z events, the following sections of the PeopleSoft EnterpriseOne server jde.ini file must be configured:

- [JDENET_KERNEL_DEF19]
- [JDEITDRV]
- [JDENET]
- [INTEROPERABILITY]

The settings for the EVN kernel, [JDEITDRV], and [JDENET] are defined in the jde.ini File Configurations for Events section of this guide. The settings for [INTEROPERABILITY] are as follows:

```
[INTEROPERABILITY]
SequenceTimeOut=XX
XMLElementSkipNullOrZero=X
```

The SequenceTimeOut setting is for sequencing of volatile events. The value is in seconds.

By default, null strings and zeros are trimmed off Z events. You can turn off this feature by entering a value of 0 (zero) for the XMLElementSkipNullOrZero setting.

Z File Events XML Document Format

Z event XML documents use the PeopleSoft EnterpriseOne XML Response format. An example of the Z event XML document can be found in the XML Format Examples (Events). Different events can have different table names and column names.

Vendor-Specific Outbound Functions

The purpose of the vendor-specific outbound function is to pass the key fields for a record in the outbound interface tables to a third party. With these keys, you can process information from the database record into your third-party system. The generic Outbound Subsystem batch process calls the function.

Each vendor-specific function is specific to the transaction being processed. You must decide how the function actually uses the database record information. Although the functions are written to your specifications, and most likely are written outside of PeopleSoft EnterpriseOne, these functions must use the required EnterpriseOne defined data structure shown below.

Data Item	Required	I/O	Description
szUserId	Y	I	User ID - 11 characters
szBatchNumber	Y	I	Batch Number - 16 characters
szTransactionNumber	Y	I	Transaction Number - 23 characters
mnLineNumber	Y	I	Line Number - double
szTransactionType	Y	I	Transaction Type - 9 characters
szDocumentType	Y	I	Document Type - 3 characters
mnSequenceNumber	Y	I	Sequence Number - double

Real-Time Events

A real-time event is notification that a business transaction has occurred in PeopleSoft EnterpriseOne. Real-time events can be generated on the EnterpriseOne server using an EnterpriseOne interface, such as HTML, WIN32, and terminal servers. Real-time events can be used for both synchronous and asynchronous processing.

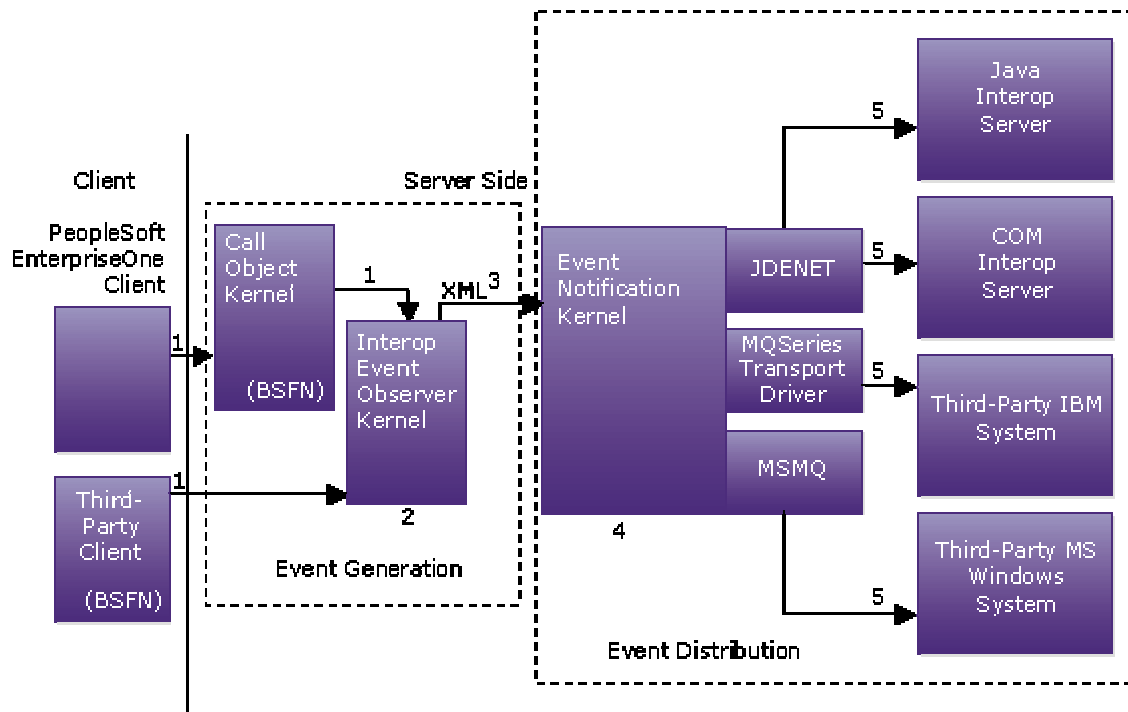
For example, an auction site using EnterpriseOne as a backend can use real-time events to update the database immediately. A user enters a new item for auction, which triggers a transaction into the EnterpriseOne system. The transaction is immediately captured. A notification is sent to an interoperability server, such as a Java connector, that then communicates the information to a web engine to update html pages so that all of the auction users can see the new item. This example illustrates a synchronous process.

Real-time event generation can also be used for asynchronous processing. For example, an online store sends orders to different vendors (business to business), captures the transactions, and the orders are entered into the vendors' systems. A user buys a book. Vendors want to enter a purchase order to the book publisher and a notification to the shipping company to pick up the book and deliver it. The book order itself can be completed as a purchase order transaction with EnterpriseOne, but the shipping request requires that the data is packaged into a commonly agreed upon format for the shipping company to process.

Real-time events use system calls that receive data from business functions that use EnterpriseOne data structures. Events can be one of the following:

Single Event	Contains one partial event. Useful if the receiver requires that events be generated per system call. Can also be used with different event types.
Aggregate Event	Contains multiple partial events. Useful if the receiver requires a document containing multiple events. For example, a supply chain solution might want the complete sales order provided as one event that contains multiple partial events.
Composite Event	Contains only single events. Useful if the customer has multiple receivers, some requiring single events and some requiring a complete event similar to an aggregate event.

Real-time event generation from a client consists of client business functions that call APIs, and then interfaces with the server Interoperability Event Observer (a kernel). Real-time event generation on the server side includes an event observer interface (a set of system APIs) that triggers real-time events, and an interoperability event observer (a kernel). Then the event observer kernel generates XML documents of the triggered real-time events and sends them to an event distribution component. The event distribution component is the same one used to send XML document notification to subscribers. A logical representation of the processes and data for real-time event generation are depicted in the following diagram and discussion:



In summary:

1. Event generation is triggered by business functions. You use the Object Configuration Manager (OCM) to map business functions to run on the EnterpriseOne server or to run locally.
 - When a business function is mapped to run on the EnterpriseOne server, the business function calls the Interoperability Event Interface within the CallObject kernel. The CallObject kernel sends the information to the Interoperability Event Observation (IEO).
 - When a real-time event is generated from a client, the client business function calls the appropriate API. The API does OCM lookup to determine where the IEO kernel is located, validates, filters, and formats the data, and then sends the information to the IEO kernel.
2. The IEO kernel creates the real-time event and produces an XML document when the real-time event is finalized.
3. The IEO kernel packages the XML document and passes the document to the Event Notification (EVN) kernel.

4. The EVN kernel determines which transport driver should handle the event.
5. The transport driver distributes the event.

Note

If you use MQSeries or MSMQ transports, system and function errors that the transport driver encounters are written to the JDE error log. The driver will write error messages and add the error codes if available.

Event Unique ID

Each real-time event has a unique ID that includes the PeopleSoft EnterpriseOne session ID.

Journaling

Real-time events are journaled using the trace feature for the JDEDEBUG log files. Journaling can be turned on or off in the jde.ini file. Journaling occurs at two points:

- A system call logs the parameter received and the APIs called.
- During the interoperability event observer process, the kernel logs additional debugging information. The logging is controlled with the LEVEL key in the [INTEROPERABILITY] section. Following are some possible values for the LEVEL key.

[INTEROPERABILITY]

LEVEL=		Writes specified interoperability event data to the debug log file. You can specify one or more of the allowable logging settings. Separate multiple values with a comma. For example: [INTEROPERABILITY] LEVEL=EVENTS,DATA Note As with any logging operation, enabling any of these settings can impact performance and cause extensive amounts of data to be written.
	EVENTS	Use this value to log the flow of events in the IEO kernel. Receiving event data and sending events are logged, but the values of the event data are not logged. This is the default level. If the LEVEL key is not present, it is identical to LEVEL= EVENTS.
	DATA	Log values of the event data and flow of the events in the IEO kernel. This level also includes all data logged with the EVENTS switch.
	PERF	Log statistics about the number of events received and the time period in which they are processed.

	DOC	Outbound XML documents are written in the temporary file on disk. If the debug log is enabled, the document location is written in the debug log. The location of the document is one of the following: - If the value of the key TempFileDir in the Interoperability section is set, the file is written to that location. For example: <pre>[INTEROPERABILITY] TempFileDir=C:\XML_DOCUMENTS</pre> - If the key TempFileDir is not set, files are written in the same directory where JDE logs and debug logs are written. Caution Setting the LEVEL=DOC key causes all real-time events to be written to the disk, which can cause a significant performance impact on the EnterpriseOne server. PeopleSoft suggests that you not use the LEVEL=DOC setting in a production environment or for stress testing of the QA environment.
	TRACE	This switch will trace execution of the IEO kernel and write data in the debug log. Because of the large amount of data logged, this level should be used only for debugging purposes.

Note

The LEVEL=DOC setting is not affected by whether debug logs are enabled or disabled. All other values under the LEVEL key (for example, TRACE) are affected by the debug log enable or disable setting.

You can also journal EVN documents by setting the SaveEVNDoc key in the [INTEROPERABILITY] section of the jde.ini file. SaveEVNDoc is similar to LEVEL=DOC but applies to the EVN kernel instead of the IEO kernel. The default value for SaveEVNDoc is zero (0), which means that EVN documents are not saved. To save EVN documents, change the value to one (1). EVN documents are saved to the directory where JDE logs and debug logs are written unless you specify a different directory. You can use TempFileDir to specify a directory, as shown in the following example:

```
[INTEROPERABILITY]
SaveEVNDoc=1
TempFileDir=C:\XML_Documents
```

See Also

- ❑ *The jde.ini File* in the *System Administration Guide* for more information about debug settings in the jde.ini file
- ❑ *Debug Tracing* in the *Development Tools Guide* for more information about enabling and disabling debug logs

Real-Time Event APIs

The system APIs are able to determine whether a system call is from a server or client. The following APIs are available for you to generate real-time events:

- jdeIEO_EventInit()
- jdeIEO_EventAdd()
- jdeIEO_EventFinalize()
- jdeIEO_CreateSingleEvent()
- jdeIEO_IsEventTypeEnabled()

Note

For specific information about these APIs, see the online API documentation.

Example: Interoperability Event Interface Calls

The following example illustrates how to create a single event:

1. Design the data structure to decide what values to provide to the real-time event.

```
typedef struct tagDSD55RTTEST
{
    char        szOrderCo[6];
    char        szBusinessUnit[13];
    char        szOrderType[3];
    MATH_NUMERIC mnOrderNo;
    MATH_NUMERIC mnLineNo;
    JDEDATE     jdRequestedDate;
    char        szItemNo[27];
    char        szDescription1[31];
    MATH_NUMERIC mnQtyOrdered;
    MATH_NUMERIC mnUnitPrice;
    MATH_NUMERIC mnUnitCost;
    char        szUserID[11];
} DSD55RTTEST, *LPDSD55RTTEST;
```

2. Define the data structure object in the business function header file.
3. Modify the business function source to call jdeIEO_CreateSingleEvent.

```
JDEBFRTN(ID) JDEBFWINAPI RealTimeEventsTest (LPBHVRCOM lpBhvrCom,
LPVOID lpVoid,
LPDSD55REALTIME lpDS)
{
/* Define Data Structure Object */
DSD55RTTEST zRTTest = {0};
IEO_EVENT_RETURN eEventReturn = eEventCallSuccess;
IEO_EVENT_ID szEventID = {0};

0Populate required members

/* Now call the API */
szEventID = jdeIEO_CreateSingleEvent ( lpBhvrCom,
    ■RealTimeEventsTest",
    ■JDERTOUT",
    ■SalesOrder",
    ■D55RTTEST",
    &zRTTest,
    sizeof(zRTTest),
    0,
    &eEventReturn );

/* Error in jdeFeedCallObjectEvent is not a critical error
   and should only be treated as a warning */
if( eEventReturn != eEventCallSuccess )
{
    /* LOG the Warning and return */
    return ER_WARNING;
}
```

The following example illustrates how to create an aggregate event:

```
IEO_EVENT_RETURN eEventReturn = eEventCallSuccess;
IEO_EVENT_ID szEventID = jdeIEO_EventInit (pBhvrCom,
eEventAggregate,
"MyFunction1",
"JDES00OUT" // EventType for AggregateEvent
"EventScope1",
0,
&eEventReturn);
0

eEventReturn = jdeIEO_EventAdd (pBhvrCom,
szEventID,
"MyFunction2",
NULL,
"D55TEST01",
&zD55TEST01,
sizeof(zD55TEST01));
0

eEventReturn = jdeIEO_EventAdd (pBhvrCom,
szEventID,
"MyFunction3",
NULL,
"D55TEST02",
&zD55TEST02,
sizeof(zD55TEST02));
0

eEventReturn = jdeIEO_EventAdd (pBhvrCom,
EventID,
"MyFunction3",
NULL,
"D55TEST03",
&zD55TEST03,
sizeof(zD55TEST03));
0

eEventReturn = jdeIEO_EventFinalize (pBhvrCom,
EventID,
"MyFunction4");
```

The following example illustrates how to create a composite event:

```
IEO_EVENT_RETURN eEventReturn = eEventCallSuccess;
IEO_EVENT_ID szEventID = jdeIEO_EventInit (pBhvrCom,
eEventComposite,
    □MyFunction1",
    □JDESOUT" // EventType for CompositeEvent
    □EventScope1",
    0,
    &eEventReturn );
0

eEventReturn = jdeIEO_EventAdd ( pBhvrCom,
    szEventID,
    □MyFunction2",
    □SODOCBEGIN", // EventType for SingleEvent
    □D55TEST01",
    &zD55TEST01,
    sizeof(zD55TEST01));
0

eEventReturn = jdeIEO_EventAdd ( pBhvrCom,
    szEventID,
    □MyFunction3",
    □SITEMADD", //EventType for SingleEvent
    □EventScope3",
    □D55TEST02",
    &zD55TEST02,
    sizeof(zD55TEST02));
0

eEventReturn = jdeIEO_EventFinalize (pBhvrCom,
    EventID,
    □MyFunction4");
```

Errors returned by the system calls might not be critical enough to stop the business process. Non-critical errors are flagged as warnings and are logged in the log file. If the business function is on the server, the warning is logged in the callobject kernel log. If the business function is on a client, the warning is logged in the client log file.

The following example illustrates an XML file that shows a composite real-time event consisting of a call to the business function F4211FSEditLine on 12/31/2000, arriving about noon, with the real-time event generated at 12:00:01.000.

```
<?xml version='1.0' encoding='utf-8' ?>
<jdeResponse type='realTimeEvent' user='JDE1214225' session='1234.786321234' environment='XDEVNIS2'>
  <header>
    <eventVersion>1.0</eventVersion>
    <type>JDESOCOUT</type>
    <scope>SalesOrder</scope>
    <user>JDE1214225</user>
    <application>P0101</application>
    <version>XJDE101</version>
    <sessionID>1234.786321234</sessionID>
    <environment>XDEVNIS2</environment>
    <host>HP9000B</host>
    <eventID>
      HP9000B-1234-1231200012000100-JDE1214225-FFA123ECBBA123EC
    </eventID>
    <date>12312000</date>
    <time>120001000</time>
  </header>

  body elementCount='1'
    <PartialEvent name='F4211FSEditLine' type='SOEDTLIN' executionOrder='1' parameterCount='12'>
      <szOrderCo type='String'>JD Edwards</szOrderCo>
      <szBusinessUnit type='String'>Mountain Region</szBusinessUnit>
      <szOrderType type='String'>SO</szOrderType>
      <mnOrderNo type='MATH_NUMERIC'>13209847</mnOrderNo>
      <mnLineNo type='MATH_NUMERIC'>122</mnLineNo>
      <jdRequestedDate type='Date'>12312000</jdRequestedDate>
      <szItemNo type='String'>12243234</szItemNo>
      <szDescription type='String'>Bicycle</szDescription>
      <mnQtyOrdered type='MATH_NUMERIC'>1</mnQtyOrdered>
      <mnUnitPrice type='MATH_NUMERIC'>249.99</mnUnitPrice>
      <mnUnitCost type='MATH_NUMERIC'>213.23</mnUnitCost>
      <szUserID type='String'>JDE1214225</szUserID>
    </PartialEvent>
  </body>
</jdeResponse>
```

Generating Real-Time Events

The following list identifies tasks for generating real-time events.

- Set up the OCM.
- Define the real-time event.
- Configure the jde.ini file.

Note

When you generate events, you might receive the following message:

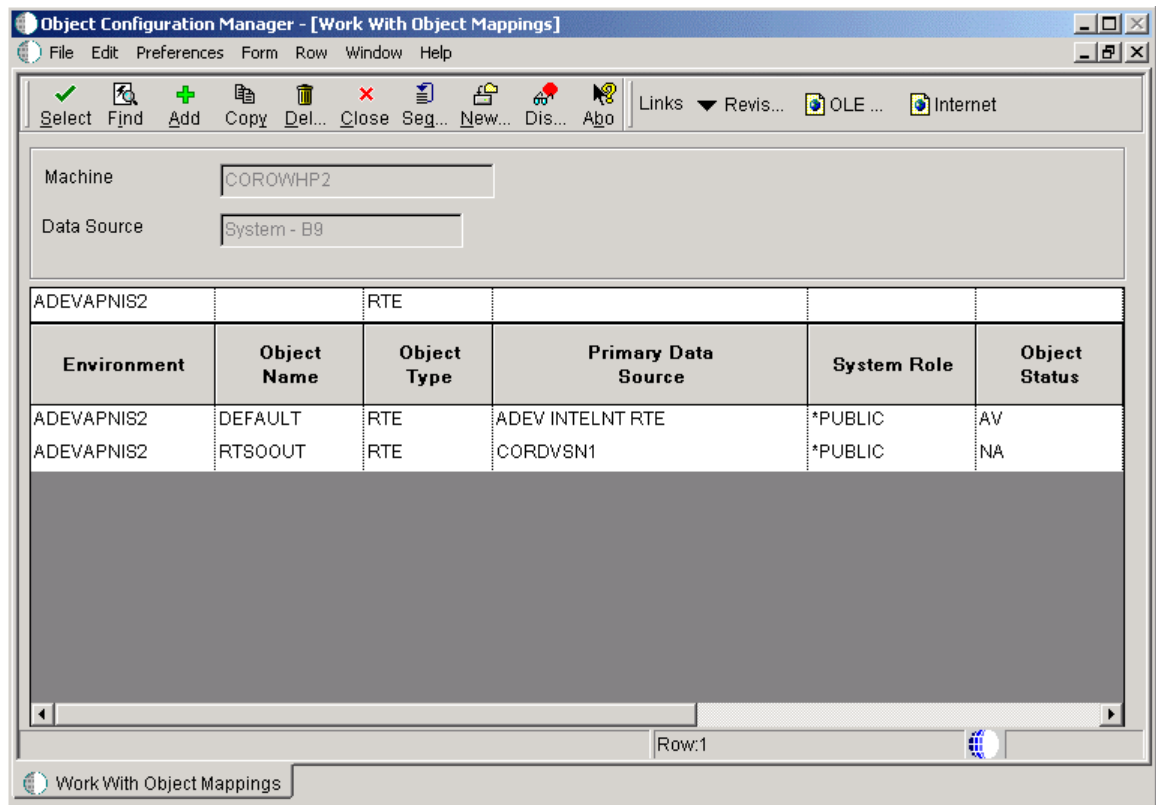
RDEL0000045 – Could not open the tables for reliable event delivery (F90703 and F90704). Reliable event delivery will be disabled.

This is only a warning message and should be ignored unless you are using the Reliable Event Delivery feature.

Setting Up the OCM for Real-Time Events

You configure the Object Configuration Manager (OCM) so that the system call can find the IEO kernel. If the business function is mapped to a client, an error is returned to the client by the system call if the IEO kernel isn't found. If the business function is mapped to the server, the error is logged in the Callobject kernel jde.log.

When you configure the OCM, include a specific environment and ensure that no two duplicate mappings are in active status at the same time. The following illustration shows the environment and the status.



To configure the OCM, access the Object Mapping Revisions form and enter RTE in the Object Type field, as shown below.

If the OCM mapping is not correctly configured on the client, the following message is written in the jde.log, and the event will not be generated:

RT0000011 jdeIEO_EventInit: Unable to find the server

If the OCM mapping is not correctly configured on the server, no error message is generated. The system call defaults to the local server for the location of the IEO kernel.

If the IEO kernel is not found on the machine that is configured in the OCM, the following error might occur:

RT0000004 jdeIEO_EventInit: ReceiveMsg failed. Error = <error test>

See Also

See the following topics in the *Configurable Network Computing Implementation Guide*:

- ❑ *Working with the Object Configuration Manager* for information about the Object Configuration Manager tool
- ❑ *Mapping Objects* for information about how to map a business function for event processing

Defining Real-Time Events

You use the Interoperability Event Definition program (P90701) to define real-time events. After you define your real-time event, be sure to activate the event by changing the status to active. If the event is not defined in the Interoperability Event Definition table (F90701), the system call returns an error message.

See Also

- ❑ *Defining Events* in the *Interoperability Guide* for information about defining events and for a description of errors if the event cannot be found

Event Sequencing

When you define your real-time events, you indicate whether the event is reliable or volatile. If you define the event as volatile, the system automatically provides event sequencing to guarantee that events are delivered in the correct order. Volatile events will be stamped using PeopleSoft EnterpriseOne Next Numbers features.

For sequencing of real-time events, the system call, `jdeIEO_EventFinalize`, retrieves the event number from the real-time events sequencing bucket and sends the number to the IEO kernel. The IEO kernel includes the event number as part of the generated event. The IEO kernel sends the event to the EVN kernel. The EVN kernel remembers the last shipped event and does sequencing based on the event number contained in the received event.

Event sequencing does impact performance. You can turn off events sequencing. You can also define a timeout value to tell the system to stop looking for a missed event when events are out of sequence. The flag and timeout settings are in the [INTEROPERABILITY] section of the `jde.ini` file.

Configure the `jde.ini` for Real-Time Events

To generate real-time events, the following sections of the PeopleSoft EnterpriseOne server `jde.ini` file must be configured:

- [JDENET_KERNEL_DEF19]
- [JDENET_KERNEL_DEF20]
- [JDEITDRV]
- [JDENET]
- [INTEROPERABILITY]

Note

The settings for the kernels [JDEITDRV] and [JDENET] are defined in *jde.ini File Configurations for Events* in the *Interoperability Guide*.

The settings for [INTEROPERABILITY] are as follows:

```
[INTEROPERABILITY]
SequenceTimeOut=XX
XMLElementSkipNullOrZero=X
```

The SequenceTimeOut setting is for sequencing of volatile events. The default value is 10 seconds.

By default, null strings and zeros are trimmed from Z real-time events. You can turn off this feature by entering a value of 0 (zero) for the XMLElementSkipNullOrZero settings.

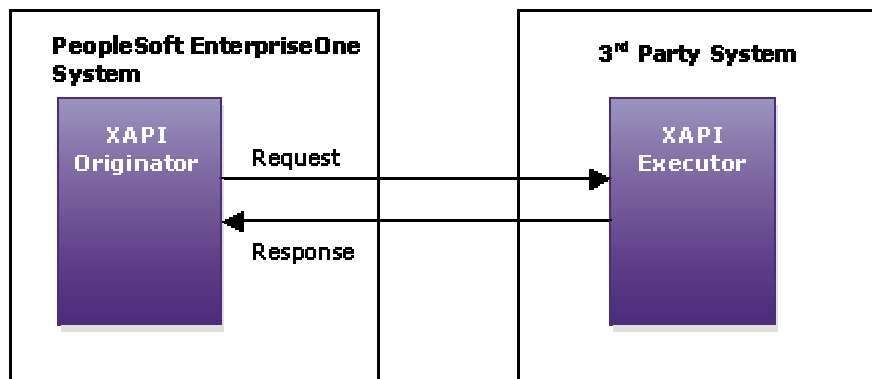
XAPI

XAPI is a PeopleSoft EnterpriseOne service that captures EnterpriseOne transactions as the transaction occurs, and then calls third-party software, end users, and other PeopleSoft systems to obtain a return response. XAPI is very similar to a real-time event, and uses the same infrastructure as real-time events. The difference between a real-time event and a XAPI event is that the subscriber to a XAPI event returns a reply to the originator. The XAPI event contains a set of structured data that includes a unique XAPI event name and a business function to be invoked upon return. Like real-time events, XAPI events can be generated on the EnterpriseOne server using any EnterpriseOne interface, such as HTML, WIN32, and terminal servers.

The XAPI structure sends outbound events and receives a reply from third-party systems. The XAPI structure also provides complete request-reply connectivity between two EnterpriseOne systems.

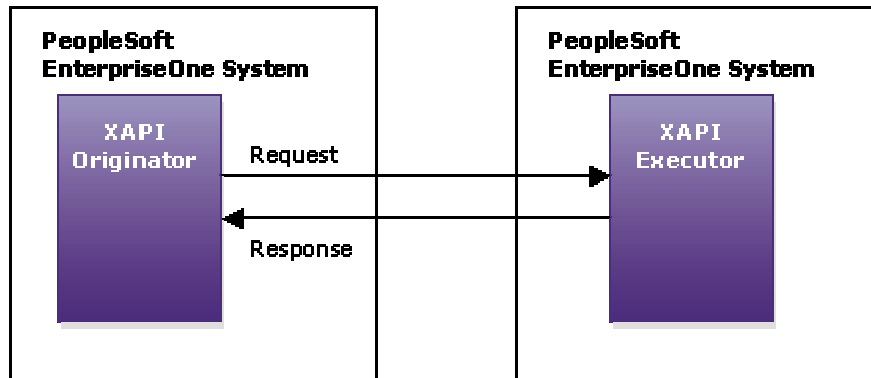
An event is generated in EnterpriseOne and sent to a third-party system for processing. The third-party system sends a response back to EnterpriseOne.

The following diagram shows a logical representation of the XAPI processing from EnterpriseOne to a third-party system:



1. EnterpriseOne, the XAPI originator, sends a request.
2. The request is sent to a third-party system.
3. The third-party system, the XAPI executor, processes the request and sends a response back to the XAPI originator.

XAPI also provides a way for two different EnterpriseOne systems to communicate with each other. The following illustration shows a logical representation of the XAPI processing from one EnterpriseOne system to another EnterpriseOne system:



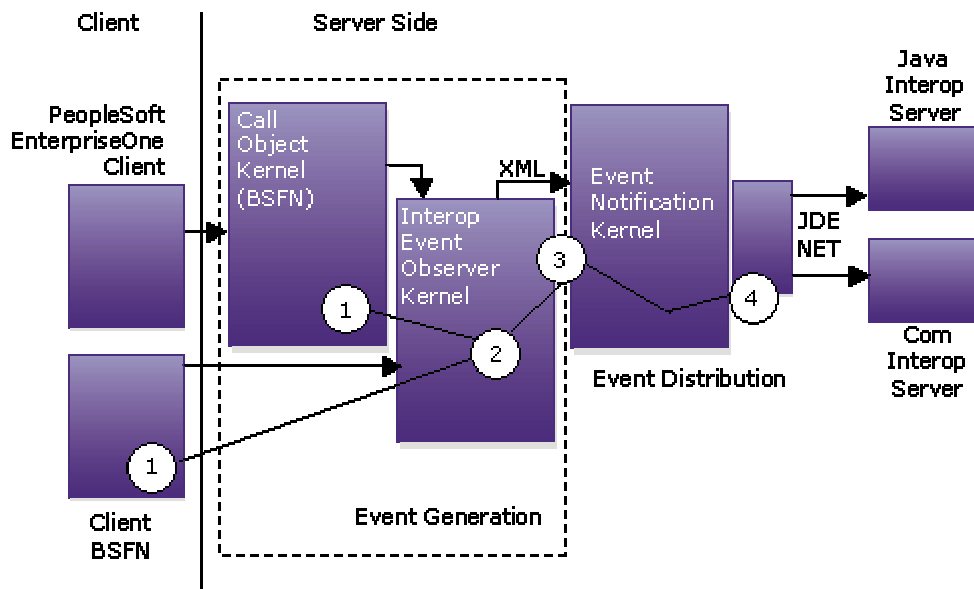
In summary:

1. The first EnterpriseOne system, the XAPI originator, sends a request.
2. The request is sent to a second EnterpriseOne system, which may share the same or different environment as the first EnterpriseOne system.
3. The second EnterpriseOne system (the XAPI executor) processes the request and sends a response back to the first EnterpriseOne system (the XAPI originator).
4. The first EnterpriseOne system (the XAPI originator) processes the response.

XAPI Outbound Events

The XAPI structure supports XAPI outbound event generation. XAPI outbound events are generated exactly the same as real-time events.

The following diagram illustrates the flow for a XAPI outbound event that is sent to a third-party system:



In summary:

1. When a XAPI event is generated from a client, the client business function calls the appropriate API. This API does an OCM lookup to determine where the IEO kernel is located. The API validates, filters, and formats the data. When a XAPI event is generated from an EnterpriseOne server, the business function calls the Interoperability Event Interface within the CallObject kernel. The data is sent as a partial event to the IEO kernel.
2. The IEO kernel creates the XAPI event and produces an XML document when the XAPI event is finalized.
3. The IEO kernel packages the XML document and passes the document to the EVN kernel.
4. The EVN kernel determines the transport driver that should handle the event, and JDENet distributes the information to the subscribers.

Note

XAPI currently does not use MQSeries or MSMQ. All events that are defined in the Interoperability Event Definition (F90701) table are sent to you if you configure your system to receive events using MQSeries and MSMQ transport drivers.

XAPI Event APIs

The following APIs are available for you to generate a XAPI call:

- jdeXAPI_Init
- jdeXAPI_Add
- jdeXAPI_Finalize
- jdeXAPI_Free
- jdeXAPI_SimpleSend
- jdeXAPI_ISCallTypeEnabled
- jdeXAPI_CALLS_ENABLED

Note

For specific information about these events, see the online API documentation.

Example: XAPI Event Creation API Usage

The following code sample illustrates how to create a XAPI event.

```
#ifdef jdeXAPI_CALLS_ENABLED
    XAPI_CALL_ID          ulXAPICallID = 0;
XAPI_CALL_RETURN          eXAPICallReturn = eEventCallSuccess;
#endif

DSD4205010A    dsD4205010A          = {0}; /*Query Header*/
DSD4205010B    dsD4205010B          = {0}; /*Query Detail*/

#ifdef jdeXAPI_CALLS_ENABLED
if(jdeXAPI_IsCallTypeEnabled("XAPIOPOUT") &&
    jdeXAPI_IsCallTypeEnabled("XAPIOPIN") )
{
    bXAPIInUse = TRUE;
}
#endif

/*-----*/
/* Call XAPIInit */

#ifdef jdeXAPI_CALLS_ENABLED
if(bXAPIInUse == TRUE)
{
    ulXAPICallID = jdeXAPI_Init( lpBhvrCom,
                                "SendOrderPromiseRequest",
                                "XAPIOPOUT",
                                NULL,
                                &eXAPICallReturn);
    if (eXAPICallReturn != eEventCallSuccess)
    {
        bExit = TRUE;
    }
}
```

```

    }
}
#endif

/*-----*/
/* Adding Header Information */

#ifdef jdeXAPI_CALLS_ENABLED
if(bXAPIInUse == TRUE)
{
    eXAPICallReturn = jdeXAPI_Add( lpBhvrCom,
                                   ulXAPICallID,
                                   "SendOrderPromiseRequest",
                                   "D4205010A",
                                   &dsD4205010A,
                                   sizeof(DSD4205010A));
    if (eXAPICallReturn != eEventCallSuccess)
    {
        bExit = TRUE;
    }
}
#endif

/*-----*/
/* Loading Detail Information */

#ifdef jdeXAPI_CALLS_ENABLED
if(bXAPIInUse == TRUE)
{
    eXAPICallReturn = jdeXAPI_Add( lpBhvrCom,
                                   ulXAPICallID,
                                   "SendOrderPromiseRequest",
                                   "D4205010B",
                                   &dsD4205010B,
                                   sizeof(DSD4205010B));
    if (eXAPICallReturn != eEventCallSuccess)
    {
        bExit = TRUE;
    }
}
#endif

#ifdef jdeXAPI_CALLS_ENABLED
if(bXAPIInUse == TRUE)
/*-----*/
/* Finalize */
{
    eXAPICallReturn = jdeXAPI_Finalize( lpBhvrCom,
                                         ulXAPICallID,
                                         "SendOrderPromiseRequest",
                                         "OrderPromiseCallback");
    if (eXAPICallReturn != eEventCallSuccess)
    {

```

```

        bExit = TRUE;
    }
}
#endif

#ifdef jdeXAPI_CALLS_ENABLED
if (eXAPICallReturn != eEventCallSuccess)
{
    /*-----*/
    /* CleanUp */

    if(bXAPIInUse == TRUE)
    {
        jdeXAPI_Free( lpBhvrCom,
                     ulXAPICallID,
                     "SendOrderPromiseRequest");
    }
}
#endif

```

Example: XML for XAPI Outbound Event

The following example shows the XML template for a XAPI event.

```

xml version="1.0" encoding="utf-8" ?>
<jdeResponse type="realTimeEvent" user="KL5449350" session="22558100.1004460662"
subtype="XAPICall" environment="DV7333">
<event>
<header>
<eventVersion>1.0</eventVersion>
<type>XAPIOPOUT</type>
<user>KL5449350</user>
<application>APIDRV</application>
<version />
<sessionId>22558100.1004460662</sessionId>
<environment>DV7333</environment>
<host>DEN-PP6954083</host>
<sequenceID>DEN-PP6954083_1540_10302001095648_KL5449350_1</sequenceID>
<date>10302001</date>
<time>095649</time>
<scope />
<codepage>utf-8</codepage>
</header>
<body elementCount="3">
<detail date="10302001" name="APIDRVFunction" time="9:56:48" type=""
DSTMPL="D4205010A" executionOrder="1" parameterCount="23">
    <szRequestId type="String">1234567</szRequestId>
    <szUserId type="String">TestUser</szUserId>
    <szQueryMode type="String">Test</szQueryMode>
    <szCustomerName type="String">John Doe</szCustomerName>
    <mnCustomerId type="Double">12345</mnCustomerId>
    <szCustomerGroup type="String">Group 1</szCustomerGroup>
    <szAddress1 type="String">Line 1</szAddress1>

```

```

<szAddress2 type="String">Suite 1</szAddress2>
<szAddress3 type="String">123 E. Main</szAddress3>
<szPostalCode type="String">50001</szPostalCode>
<szCity type="String">Centennial</szCity>
<szCounty type="String">Arap</szCounty>
<szStateProvince type="String">CO</szStateProvince>
<szCountry type="String">US</szCountry>
<szBusinessObjective type="String" />
<mnTraceDepth type="Double">0</mnTraceDepth>
<mnPenaltyCostAdjustment type="Double">0</mnPenaltyCostAdjustment>
<szOrderNumber type="String">1000</szOrderNumber>
<nAllowBackorders type="Int">49</nAllowBackorders>
<nAllowSubstitution type="Int">48</nAllowSubstitution>
<nAllowPartialLineShip type="Int">49</nAllowPartialLineShip>
<nAllowPartialOrderShip type="Int">49</nAllowPartialOrderShip>
<nAllowMultisource type="Int">49</nAllowMultisource>
</detail>
<detail date="10302001" name="APIDRVFunction" time="9:56:49" type=""
DSTMPL="D4205010B" executionOrder="2" parameterCount="17">
  <mnLineNumber type="Double">1</mnLineNumber>
  <mnCacheLineNumber type="Double">1</mnCacheLineNumber>
  <mnItemNumber type="Double">2222</mnItemNumber>
  <sz2ndItemNumber type="String">1234567</sz2ndItemNumber>
  <sz3rdItemNumber type="String">2234567</sz3rdItemNumber>
  <szOrderUnit type="String">123</szOrderUnit>
  <mnOrderQuantity type="Double">12</mnOrderQuantity>
  <szPlanningUnit type="String">ECL</szPlanningUnit>
  <mnPlanningQuantity type="Double">12</mnPlanningQuantity>
  <mnPlanningMultiple type="Double">1</mnPlanningMultiple>
  <mnPlanningUnitPrice type="Double">1234</mnPlanningUnitPrice>
  <jdRequestDate type="Date">10302001</jdRequestDate>
  <szShippingGroup type="String">Ship Group</szShippingGroup>
  <szMultiSource type="String">MS</szMultiSource>
  <nAllowPartialLineShip type="Int">49</nAllowPartialLineShip>
  <nAllowBackorders type="Int">49</nAllowBackorders>
  <nAllowSubstitution type="Int">48</nAllowSubstitution>
</detail>
<detail date="10302001" name="XAPICall" time="09:56:49" type="" DSTMPL="DXAPIROUTE"
executionOrder="3" parameterCount="4">
  <ClientPort type="Int">6009</ClientPort>
  <ClientIP type="Int">167810863</ClientIP>
  <ClientMagicNumber type="Int">32781408</ClientMagicNumber>
  <XAPIMethodID type="String">GetComputerID</XAPIMethodID>
</detail>
</body>
</event>
</jdeResponse>

```

Routing Information

All XAPI events must include DXAPIROUTE in the XML file, as noted in the shaded area in the example XML file above. DXAPIROUTE contains the routing information that is to be returned to the originating client. The jdeXAPI_Finalize API appends DXAPIROUTE data execution.

Generating XAPI Events

The following list identifies tasks for generating XAPI events to send to a third-party:

- Set up the OCM.
- Define the XAPI" event in table F90701.
- Set up subscription information in table F90702.
- Configure the server jde.ini file.

Note

When you generate events, you might receive the following message:

RDEL0000045 – Could not open the tables for reliable event delivery (F90703 and F90704). Reliable event delivery will be disabled.

This is only a warning message and should be ignored unless you are using the Reliable Event Delivery feature.

Set Up the OCM for XAPI

If your interface to PeopleSoft EnterpriseOne is other than an EnterpriseOne client, you must configure the OCM so that the system call can find the IEO kernel. When you configure the OCM, include a specific environment and ensure that no two duplicate mappings are in active status at the same time.

To configure the OCM, access the Object Mapping Revisions form and enter XAPI in the Object Type field. Configuring the OCM with the XAPI entry enables the system call to find the IEO kernel. If the OCM is not properly configured, the system generates an error message. OCM error messages for XAPI events are the same as the OCM error messages for real-time events.

See Also

- ❑ *Setting Up the OCM for Real-Time Events* in the *Interoperability Guide* for information about setting up the Object Configuration Manager for events processing

See the following topics in the *Configurable Network Computing Implementation Guide*:

- ❑ *Mapping Objects*
- ❑ *Working with the Object Configuration Manager*

Define the XAPI Event

You use the Interoperability Event Definition (P90701) program to define XAPI events. When you define XAPI events, the system automatically updates the Event Category field to Container. All XAPI events use the data structure option. The system automatically adds the DXAPIROUTE data structure, which is required for XAPI events. The DXAPIROUTE data structure contains the routing information that is to be returned to the originating system. The jdeXAPI_Finalize API appends DXAPIROUTE data execution. After you define your XAPI event, be sure to activate the event by changing the status.

See Also

- ❑ *Defining Events* in the *Interoperability Guide* for information about how to add a XAPI event to the Interoperability Event Definition table (F90701)

Set Up XAPI Subscription Information

If you are generating XAPI events, you must define a logical subscriber and set up XAPI event subscriber information. The logical subscriber must exist before you can add XAPI event subscriber information. If subscriber information is missing, the XAPI event is generated but cannot be delivered. You use the Interoperability Event Subscription program (P90702) to define the logical subscriber and to set up XAPI subscriber information. After you set up the XAPI subscriber, be sure to activate the subscriber by changing the status.

See Also

- ❑ *Setting Up Subscriber Information* in the *Interoperability Guide* for information about how to define a logical subscriber and then set up XAPI subscriber information

Configure the jde.ini File for XAPI Events

To generate XAPI events, the following sections of the PeopleSoft EnterpriseOne server jde.ini file must be configured:

- [JDENET_KERNEL_DEF19]
- [JDENET_KERNEL_DEF20]
- [JDEITDRV]

If the jde.ini file is not properly configured for XAPI events, the following error message is written to the jde.log file:

XAPI Event [Event Name] cannot be subscribed. Must have XAPI Definition in the INI file.

Make sure the XAPI event is defined in the Single and Container Event Definition table (F90701) and that XAPI Executor information is defined in the jde.ini file.

You can ignore the following error message because XAPI subscription is persisted and cannot be unsubscribed:

Cannot unsubscribe XAPI event.

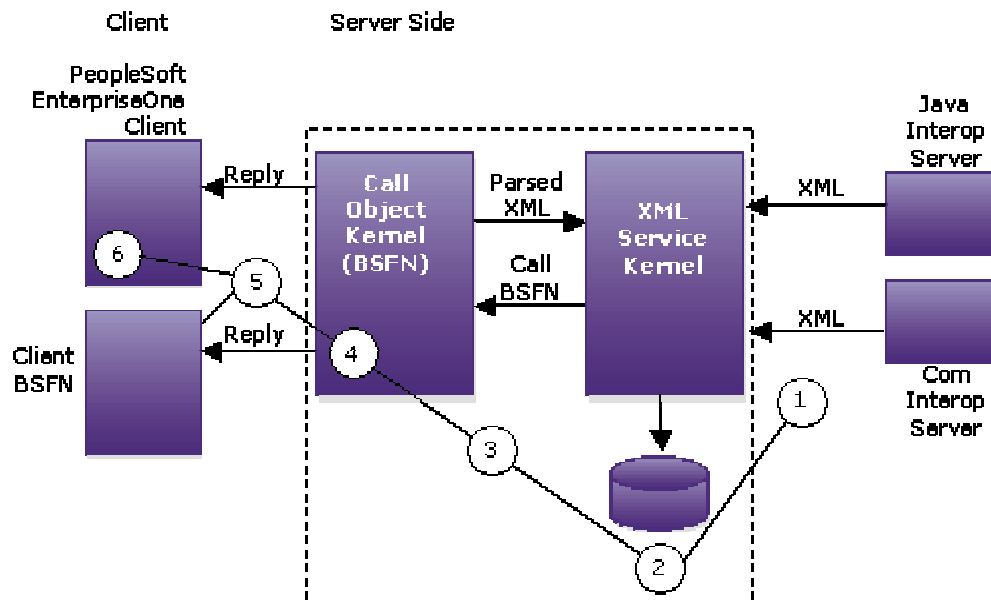
See Also

- ❑ *jde.ini File Configurations for Events* in the *Interoperability Guide* for server kernel and [JDEITDRV] configurations

XAPI Inbound Response

The XAPI structure provides for an inbound response. The XAPI inbound response happens after a XAPI event is generated. The XAPI inbound response is handled by the third-party system. The third-party system, the XAPI executor, processes the request (event) and returns a reply to the XAPI originator.

When the return XML document is received, it is routed to the XML service kernel. The XML service kernel saves the XML document to disk, creates a unique handle, and then calls the callback business function that is provided in the DXAPIROUTE XAPI method ID element in the XML document. The following diagram illustrates the flow of a XAPI response from a third-party system to the PeopleSoft EnterpriseOne originating system. The dotted line indicates the flow.



The inbound portion of XAPI functions as follows:

1. An inbound XML document is passed from a third-party system to the XML service kernel.
2. The XML service kernel creates a unique XML handle and stores the document on disk.
3. The XML service kernel reads the XAPICallMethod attribute from the XML document and passes the XML handle as the parameter to the specified business function.
4. The business function (XAPICallMethod) uses XML service APIs to read and parse the XML data into EnterpriseOne data.
5. The business function (XAPICallMethod) uses XML CallObject to send the reply to the originator.
6. An EnterpriseOne client can poll for the XAPI response from the EnterpriseOne server.

XAPI Response Parsing APIs

The following APIs are available for you to generate an inbound XAPI response:

- jdeXML_GetDSCount
- jdeXML_GetDSName
- jdeXML_ParseDS
- jdeXML_DeleteXML

Note

For specific information about the XML service kernel, see the online API documentation.

Example: XAPI Response Parsing API Usage

The following example illustrates how the business function uses the XML service APIs to read and parse the XML data:

```
int iCurrentRecord;
int      iHeaderCount;

DSD4205030A      dsD4205030A = {0};
DSD4205030B      dsD4205030B = {0};

#ifdef jdeXAPI_CALLS_ENABLED

if(jdeXAPI_IsCallTypeEnabled("XAPIOPOUT") &&
    jdeXAPI_IsCallTypeEnabled("XAPIOPIN") )
{
    iRecordCount = jdeXML_GetDSCount(lpDS->szXMLHandle);

    if (iRecordCount > 0)
    {
        for (iCurrentRecord = 0; iCurrentRecord < iRecordCount;
iCurrentRecord++)
        {
            jdeXML_GetDSName(lpDS->szXMLHandle,
                             iCurrentRecord,
                             nidDSName);
            if (jdestrcmp(nidDSName,(const char*)"D4205030A") == 0)
            {
                jdeXML_ParseDS( lpDS->szXMLHandle,
                                iCurrentRecord,
                                &dsD4205030A,
                                sizeof(DSD4205030A));
            }
            else
            {
                jdeXML_ParseDS( lpDS->szXMLHandle,
                                iCurrentRecord,
                                &dsD4205030B,
                                sizeof(DSD4205030B));
            }
        }

        if (iCurrentRecord == iRecordCount)
        {
            jdeXML_DeleteXML(lpDS->szXMLHandle);
        }
    }
}

#endif
```

Example: Inbound XAPI Response

The following sample shows an inbound XAPI response:

```
<?xml version="1.0" encoding="utf-8" ?>
<jdeRequest pwd="JDE" type="xapicalmethod" user="JDE" session=""
environment="DV7333" sessionidle="">
<header>
<eventVesrion>1.0</eventVesrion>
<type>XAPIOPIN</type>
<user>JDE</user>
<application>XPI</application>
<version />
<sessionID />
<environment>DEVXPINT</environment>
<host>denxpi7</host>
<sequenceID />
<date>09122001</date>
<time>094951</time>
<scope />
<codepage>utf-8</codepage>
</header>
<body elementCount="3">
<params type="D4205030A" executionOrder="1" parameterCount="24">
  <param name="type" />
  <param name="dateStamp" />
  <param name="timeStamp" />
  <param name="szRequestId">1|ZJDE0001</param>
  <param name="szBusinessObjective">Maximize_Service</param>
  <param name="mnResultNumber">0.0</param>
  <param name="mnTotalCost">0.0</param>
  <param name="mnTotalDeliveryCost">0.0</param>
  <param name="mnTotalPrice">0.0</param>
  <param name="mnTotalProfit">0.0</param>
  <param name="mnTotalMargin">0.0</param>
  <param name="mnTotalValue">0.0</param>
  <param name="mnLatestLineDate">0.0</param>
  <param name="mnNumberOfBackorders">0.0</param>
  <param name="mnNumberOfSubstitutions">0.0</param>
  <param name="mnOrderFillRate">0.0</param>
  <param name="szErrorCode" />
  <param name="szErrorDescription" />
  <param name="szOrderNumber">3115|SO|00200</param>
  <param name="nAllowPartialOrderShip">0</param>
  <param name="nAllowMultisource">0</param>
  <param name="nAllowBackorders">0</param>
  <param name="nAllowSubstitution">0</param>
  <param name="nAllowPartialLineShip">0</param>
</params>
<params type="D4205030B" executionOrder="2" parameterCount="28">
  <param name="type" />
  <param name="dateStamp" />
  <param name="timeStamp" />
```

```

<param name="mnLineNumber">1.0</param>
<param name="mnOriginalLineNumber">1.0</param>
<param name="mnCacheLineNumber">1.0</param>
<param name="mnRequestedItem">60011.0</param>
<param name="mnAvailableItem">60011.0</param>
<param name="mnAvailableAmount">25.0</param>
<param name="jdAvaiableDate">09/12/2001 00:00:00</param>
<param name="jdRequestedDate">09/10/2001 00:00:00</param>
<param name="jdPickDate">09/11/2001 00:00:00</param>
<param name="jdShipDate">09/11/2001 00:00:00</param>
<param name="szShipLocation" />
<param name="mnCost">0.0</param>
<param name="mnDeliveryCost">0.0</param>
<param name="mnPrice">0.0</param>
<param name="mnProfit">0.0</param>
<param name="mnMargin">0.0</param>
<param name="mnValue">0.0</param>
<param name="mnSubstitutionRatio">0.0</param>
<param name="szShippingGroup" />
<param name="szMultiSource" />
<param name="szErrorCode" />
<param name="szSuspectedCause" />
<param name="nAllowPartialOrderShip">0</param>
<param name="nAllowBackorders">0</param>
<param name="nAllowSubstitution">0</param>
</params>
<params type="DXAPIROUTE" executionOrder="3" parameterCount="7">
  <param name="type" />
  <param name="dateStamp">09/05/2001 00:00:00</param> <param
name="timeStamp">13:54:04</param>
  <param name="ClientPort">6009</param>
  <param name="ClientIP">168045665</param>
  <param name="ClientMagicNumber">3</param>
  <param name="XAPIMethodID">OrderPromiseCallback</param>
</params>
</body>
</jdeRequest>

```

Configure the jde.ini File for XAPI Response

The following sections of the EnterpriseOne server jde.ini file must be configured for the XAPI response portion of the XAPI structure:

- [JDENET_KERNEL_DEF22]
 - [JDENET_KERNEL_DEF24]
 - [XAPI]
 - [XMLLookupInfo]
- ```

[XAPI]
XMLDirectory=c:\builds\bdev\log\

```

---

**Note**

The XML document directory must be registered in the jde.ini file on the server under the [XAPI] section in the XMLDirectory key. The key contains the directory on the server where XML documents are to be stored.

---

```
[XMLLookupInfo]
XMLRequestType5=realTimeEvent
XMLKernelMessageRange5=14251
XMLKernelHostName5=local
XMLKernelPort5=0
XMLKernelReply5=0
```

**See Also**

- ❑ *jde.ini File Configurations for Events* in the *Interoperability Guide* for kernel configurations

**Configure the Client jde.ini File for XAPI**

If you are using a PeopleSoft EnterpriseOne client to generate XAPI events, you must define the Client Dispatch kernel and [JDENET] sections of the client jde.ini file. If your interface to the EnterpriseOne server is other than an EnterpriseOne client, these two settings are not required. The settings enable the EnterpriseOne client to poll for the XAPI response message from the EnterpriseOne server.

Use the following settings to configure your EnterpriseOne client jde.ini file:

```
[JDENET_KERNEL_DEF27]
krnlName=CLIENT_DISPATCH_KERNEL
dispatchDLLName=jdeuser.dll
dispatchDLLFunction=_JDENET_ClientDispatch
maxNumberOfProcesses=0
numberOfAutoStartProcesses=0

[JDENET]
serviceNameListen=6004
serviceNameConnect=6004
maxKernelRanges=27
netTrace=0
```

---

**Note**

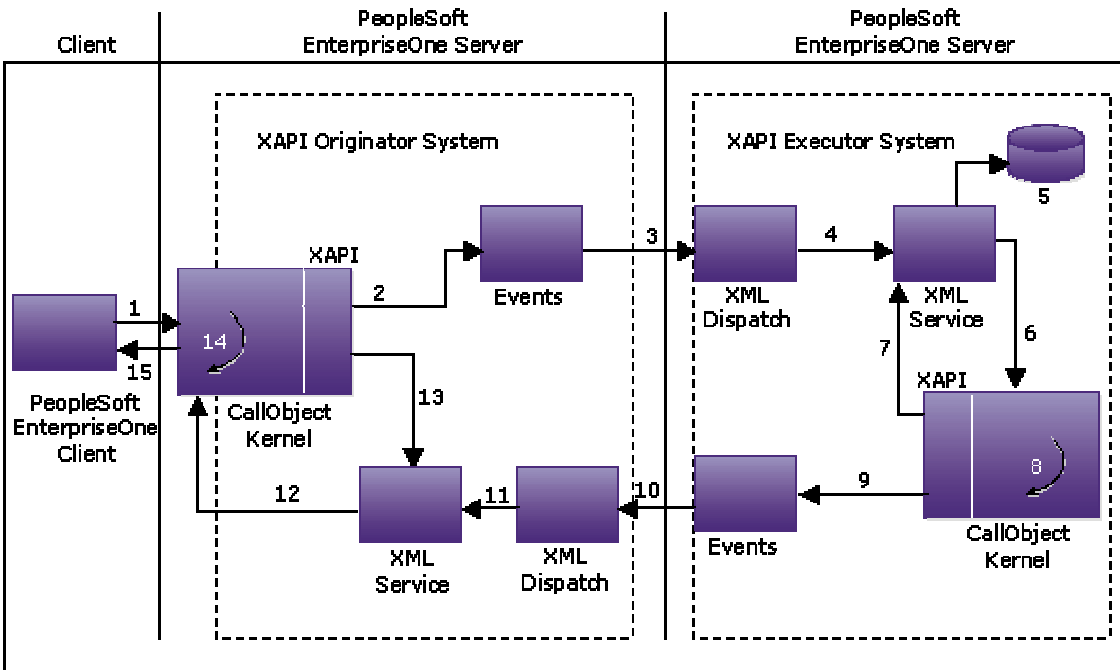
The serviceNameListen and serviceNameConnect settings must be the same as the server's settings. For example, if your server jde.ini file has serviceNameListen=6005 and serviceNameConnect=6005, then your EnterpriseOne client jde.ini file must be serviceNameListen=6005 and serviceNameConnect=6005.

The value for maxKernelRanges setting should be the same value as the server.

---

## XAPI PeopleSoft EnterpriseOne to PeopleSoft EnterpriseOne

The XAPI structure also provides the capability for two different PeopleSoft EnterpriseOne systems to communicate with each other. The first EnterpriseOne system (XAPI originator system) generates a XAPI request (event). Instead of the request being distributed to a third-party system, JDENet sends the request to a second EnterpriseOne system. The second EnterpriseOne system (XAPI executor system) processes the request and sends a response back to the first EnterpriseOne system. The following illustration shows a logical representation of the EnterpriseOne system to EnterpriseOne system XAPI processing:



### XAPI Originator System Flow

1. A business function calls the Interoperability Event Interface within the CallObject kernel to send a request.
2. The business function uses XAPI APIs to create the XAPI request. XAPI adds the callback function and sends the request to the events structure
3. The IEO kernel creates the XAPI event in XML format and sends the XML document to the EVN kernel. The EVN kernel ships the XML document to the XML Dispatch kernel of the second EnterpriseOne system. The XML document is shipped through JDENet using persistent subscription information. A routing token that contains the sender's server and port information is added. The message type for the event must be *RealTimeEvent*.

## XAPI Executor System

4. The XML Dispatch kernel receives the XML package and sends the event request and routing information to the XML Service kernel.
5. The XML Service kernel stores the XAPI request and creates a file handle for the XAPI request. The XML kernel also creates XML based routing information, stores the routing information, and creates a file handle for the routing information. The XML Service kernel uses the Event Request Definition table (F907012) to find the business function that will process the request.
6. The XML Service kernel invokes the business function (in CallObject) with the XML request handle and the routing information handle.
7. The business function uses XAPI APIs to parse and process the request. XAPI APIs load the XML request into memory.
8. The business function processes the XAPI event request. The business function also creates a XAPI response. The message type for the response must be *xapicallmethod*. The XAPI response is in XML format. The business function also passes the routing information handle.
9. The XAPI response originator sends the response and the routing information to the events structure.
10. The IEO kernel formats the XAPI response in XML format and sends the XML document to the EVN kernel. The EVN kernel uses direct routing to send the response and routing information to the XML Dispatch kernel of the first EnterpriseOne system (XAPI originator system). Direct routing means sending the XAPI reply to the same request-originating server.

## XAPI Originator System

11. The XML Dispatch kernel receives the response XML document and sends the response to the XML Service kernel.
12. The XML service kernel stores the response document, creates a file handle, and invokes the callback business function with the file handle.
13. The business function parses the response document using XAPI APIs (XAPI response handler). XAPI APIs use the XML Service kernel to load the document into memory.
14. The business function uses XAPI APIs (CallObject kernel) to process the response.
15. The business function can poll for the XAPI response from the EnterpriseOne server.

---

### Note

You can send a request from one EnterpriseOne system to another EnterpriseOne system for processing with no return reply. If you do not want a response, you would use the above steps through step 8, without processing the request. No response would be generated.

---

You can use the reliable event delivery feature to process XAPI events.

## XAPI PeopleSoft EnterpriseOne to PeopleSoft EnterpriseOne APIs

Following is a list of APIs, categorized by function, that you use to generate XAPI events when you are working with two PeopleSoft EnterpriseOne systems. For specific information about these APIs, see the online API documentation.

### XAPI Outbound Request Generation APIs

You use the following APIs to generate a XAPI event (request from the originator system). The APIs listed below are the same as those APIs identified in the XAPI Outbound Events section.

- jdeXAPI\_SimpleSend
- jdeXAPI\_Init
- jdeXAPI\_Add
- jdeXAPI\_Finalize
- jdeXAPI\_Free

### See Also

The following examples in the *Interoperability Guide*:

- ❑ *Example: XAPI Event Creation API Usage*
- ❑ *Example: XML for XAPI Outbound Event*

### XAPI Outbound Request Handling APIs

The following APIs are used by the mapped business function (executor system) to retrieve XML data from the outbound XAPI request document:

- jdeXMLRequest\_GetDSCount
- jdeXMLRequest\_GetDSName
- jdeXMLRequest\_ParseDS
- jdeXMLRequest\_DeleteXML
- jdeXMLRequest\_ParseNextDSByName
- jdeXMLRequest\_PrepareDSLlistForIterationByName

## Example: XAPI Outbound Request Parsing API Usage

The following code example shows the API Usage for generating a outbound request from the PeopleSoft EnterpriseOne originator to the EnterpriseOne executor:

```
API_System FunctionsSampleXAPIRequestParsingAPIUsage
Last Modified: | October 21, 2002

Example
int iXMLRecordCount = 0;
int iCurrentRecord = 0;
NID nidDSName;
ID idReturnValue = ER_SUCCESS;
ID idSORecordCount = ER_ERROR; /*Return Code*/
MATH_NUMERIC mnBatchNumber = {0};
unsigned long lBatchNumber = {0};

DSD4206030A dsD4206030A = {0};
/* CacheProcessInboundDemandRequest B4206030.c */
DSD4206000I dsD4206000I = {0};
/* Demand scheduling inbound DSTR */

iXMLRecordCount = jdeXMLRequest_GetDSCount(lpDS->szXMLHandle);

if(iXMLRecordCount > 0)
{
 for (iCurrentRecord = 0; iCurrentRecord < iXMLRecordCount; iCurrentRecord++)
 {
 memset((void *)&dsD4206000I, (int)(_J('\0')), sizeof(DSD4206000I));
 memset((void *)&nidDSName, (int)(_J('\0')), sizeof(NID));

 if(jdeXMLRequest_GetDSName(lpDS->szXMLHandle,
 iCurrentRecord,
 nidDSName))
 {
 /* Retrieving data*/
 if (jdeStricmp(nidDSName, (const JCHAR *)_J("D40R0180B")) == 0)
 {
 if (jdeXMLRequest_ParseDS(lpDS->szXMLHandle,
 iCurrentRecord,
 &dsD4206000I,
 sizeof(DSD4206000I)))
 {
 /* Get next number for the batch number of the
 inbound INVRPT record*/if
 (dsD4206000I.cInventoryAdvisement == _J('1'))
 {
 lBatchNumber = JDB_GetInternalNextNumber();
 LongToMathNumeric(lBatchNumber, &mnBatchNumber);
 FormatMathNumeric(dsD4206000I.szBatch,
 &mnBatchNumber);
 }

 /* Setup cancel flag for pending delete record */
 }
 }
 }
 }
}
```

```

 if (dsD4206000I.cPendingDelete == _J('1'))
 { /* Flag set as 1 for any cancel demand record */
 dsD4206000I.cCancelFlag = _J('1');
 }
 else
 { /* Flag set as 9 for any non cancel demand record */
 dsD4206000I.cCancelFlag = _J('9');
 }

 /* Load parms for cache */
 memset((void *)(&dsD4206030A), (int)(_J('\0')),
sizeof(DSD4206030A));

 I4206000_LoadParmsToCache(&dsD4206000I, &dsD4206030A);
 MathCopy(&dsD4206030A.mnJobnumberA, lpmnJobNumber);

 /* Add the DSTR to cache */
 idReturnValue = jdeCallObject(_J("CacheProcessInboundDemandRequest")
,
 (LPFNBHVR)NULL ,
 lpBhvrCom ,
 lpVoid ,
 (LPVOID)&dsD4206030A,
 (CALLMAP *) NULL,
 (int) 0,
 (JCHAR*)NULL ,
 (JCHAR*)NULL ,
 (int) 0);

 /* Write XML DSTR to cache fail */
 if (idReturnValue == ER_ERROR)
 {
 jdeErrorSet(lpBhvrCom, lpVoid, (ID) 0, _J("032E"), (LPVOID) NULL);
 }

 }
 else
 { /* warning XML parse fail */
 jdeErrorSet(lpBhvrCom, lpVoid, (ID) 0, _J("40R46"), (LPVOID)
NULL);
 }
} /* end if */
} /* end if DS name */
} /* end for - looping all matching XML DSTR */

/* Ensure there is at least one record */
idSORecordCount = ER_SUCCESS;

} /*if(iXMLRecordCount > 0) */

return idSORecordCount;

```

The following illustrates the XAPI request document from the EnterpriseOne originator system to the EnterpriseOne executor system:

130

```

 <params type="DXAPIROUTE" executionOrder="1" parameterCount="4">
 <param name="ClientPort">6024</param>
 <param name="ClientIP">168007331</param>
 <param name="ClientMagicNumber">1</param>
 <param name="XAPIMethodID">XAPITestResponse</param>
 </params>
 </body>
</jdeRequest>

```

## XAPI Inbound Response Generation APIs

The following APIs are used to generate a response (executor system):

- jdeXAPIResponse\_SimpleSend
- jdeXAPIResponse\_Init
- jdeXAPIResponse\_Add
- jdeXAPIResponse\_Finalize
- jdeXAPIResponse\_Free

## Example: XAPI Inbound Response Parsing API Usage

The following code example shows the API usage for generating an inbound reply from the PeopleSoft EnterpriseOne executor system to the EnterpriseOne originator system:

```

JDEBFRTN (ID) JDEBFWINAPI SendOrderPromiseRequest (LPBHVRCOM lpBhvrCom, LPVOID
lpVoid, LPDSD4205010 lpDS)
{
 /******
 * Variable declarations
 *****/

 char cPromisableLine = ' ';
 int nHeaderBackOrderAllowed = ' ';
 HUSER hUser ;
 ID JDEDBResult = JDEDB_PASSED;
 BOOL bExit = FALSE;
 BOOL bB4001040Called = FALSE;
 BOOL bXAPIInUse = FALSE;
 BOOL bAtLeastOneDetail = FALSE;

 #ifdef jdeXAPI_CALLS_ENABLED
 XAPI_CALL_ID ulXAPICallID = 0;
 XAPI_CALL_RETURN eXAPICallReturn = eEventCallSuccess;
 #endif

 /******
 * Declare structures
 *****/

 DSD4001040 dsD4001040 = {0};
 DSD4205020 dsD4205020 = {0};
 DSD4205040 dsD4205040 = {0}; /* Header Info */
 DSD4205050 dsD4205050 = {0}; /* Detail Info */
 DSD4205010A dsD4205010A = {0}; /* Query Header */

```

```

 DSD4205010B dsD4205010B = {0} /* Query Detail */
 DSD0100042 dsD0100042 = {0};
 LPDSD4205040H lpDSD4205040H = (LPDSD4205040H) NULL;
 LPDSD4205050D lpDSD4205050D = (LPDSD4205050D) NULL;

 /*****
* Declare pointers
*****/

/*****
* Check for NULL pointers
*****/
if ((lpBhvrCom == (LPBHVRCOM) NULL) ||
 (lpVoid == (LPVOID) NULL) ||
 (lpDS == (LPDSD4205010) NULL))
{
 jdeErrorSet (lpBhvrCom, lpVoid, (ID) 0, "4363", (LPVOID) NULL);
 return ER_ERROR;
}

/* Retrieving hUser */

JDEDBResult = JDB_InitBhvr (lpBhvrCom, &hUser, (char *)NULL, JDEDB_COMMIT_AUTO) ;

if (JDEDBResult == JDEDB_FAILED)
{
 jdeSetGBRError (lpBhvrCom, lpVoid, (ID) 0, "4363") ;
 return ER_ERROR ;
}

/*****
* Set pointers
*****/

/*****
* Main Processing
*****/
 /*-----*/
 /* Setting Up ErrorCode
 */
 lpDS->cErrorCode = '0';

 /*-----*/
 /* Determining if XAPI is ready to be used
 */

 bXAPIInUse = FALSE;

#ifdef jdeXAPI_CALLS_ENABLED
 if(jdeXAPI_IsCallTypeEnabled("XAPIOPOUT") &&
 jdeXAPI_IsCallTypeEnabled("XAPIOPIN"))
 {
 bXAPIInUse = TRUE;
 }
#endif
 /*-----*/
 /* Data validation and defaults.
 */

```

```

/* When Display Before Accept Mode is on, validate Key */
/* Information. Otherwise retrieve it from Header Record*/

if((lpDS->cDisplayBeforeAcceptMode == '1') &&
 ((MathZeroTest(&lpDS->mnOrderNumber) == 0) ||
 (IsStringBlank(lpDS->szOrderType)) ||
 (IsStringBlank(lpDS->szOrderCompany))))
{
 bExit = TRUE;
}
else
{
 MathCopy(&dsD4205040.mnOrderNumber, &lpDS->mnOrderNumber);
 strncpy(dsD4205040.szOrderType,
 lpDS->szOrderType,
 sizeof(dsD4205040.szOrderType));
 strncpy(dsD4205040.szComputerID,
 lpDS->szOrderCompany,
 sizeof(dsD4205040.szOrderCompany));
 dsD4205040.cUseCacheOrWF = lpDS->cUseCacheOrWF;
 strncpy(dsD4205040.szComputerID,
 lpDS->szComputerID,
 sizeof(dsD4205040.szComputerID));
 MathCopy(&dsD4205040.mnJobNumber, &lpDS->mnJobNumber);
 jdeCallObject("GetSalesOrderHeaderRecord",
 NULL,
 lpBhvrCom, lpVoid,
 (LPVOID)&dsD4205040,
 (CALLMAP *) NULL,
 (int) 0,
 (char *) NULL,
 (char *) NULL,
 (int) 0);

 lpDSD4205040H = (LPDSD4205040H)jdeRemoveDataPtr(hUser,
(ulong)dsD4205040.idHeaderRecord);

 if (lpDSD4205040H == NULL)
 {
 bExit = TRUE;
 }
}

/*-----*/
/* Set error if we're exiting at this point */

if (bExit == TRUE)
{
 lpDS->cErrorCode = '1';
 /* Sales Order Header Not Found */
 strncpy(lpDS->szErrorMessageID,
 "072T",
 sizeof(lpDS->szErrorMessageID));
 if (lpDS->cSuppressError != '1')
 {

```

```

 jdeErrorSet (lpBhvrCom, lpVoid, (ID) 0, "072T", (LPVOID) NULL);
 }
}

/*-----*/
/* Default Promising Flag is always 1 */
lpDS->cDefaultPromisingFlags = 1;

if (bExit == FALSE)
{
 /*-----*/
 /* Call XAPIInit */

 #ifdef jdeXAPI_CALLS_ENABLED
 if(bXAPIInUse == TRUE)
 {
 ulXAPICallID = jdeXAPI_Init(lpBhvrCom,
 "SendOrderPromiseRequest",
 "XAPIOPOUT",
 NULL,
 &eXAPICallReturn);
 if (eXAPICallReturn != eEventCallSuccess)
 {
 bExit = TRUE;
 }
 }
 #endif
 if (bExit == FALSE)
 {
 /*-----*/
 /* Loading Header Information */

 I4205010_PopulateQueryHeader(lpDS, &dsD4205010A,
 lpDSD4205040H, &dsD0100042, hUser, lpVoid, lpBhvrCom);

 nHeaderBackOrderAllowed = dsD4205010A.nAllowBackorders;

 /*-----*/
 /* Adding Header Information */

 #ifdef jdeXAPI_CALLS_ENABLED
 if(bXAPIInUse == TRUE)
 {
 eXAPICallReturn = jdeXAPI_Add(lpBhvrCom,
 ulXAPICallID,
 "SendOrderPromiseRequest",
 "D4205010A",
 &dsD4205010A,
 sizeof(DSD4205010A));
 if (eXAPICallReturn != eEventCallSuccess)
 {
 bExit = TRUE;
 }
 }
 #endif
 }
}

```

```

 }
}

if (bExit == FALSE)
{
 /*-----*/
 /* Loading Detail Information */

 MathCopy(&dsD4205050.mnOrderNumber,&lpDS->mnOrderNumber);
 strncpy(dsD4205050.szOrderType,lpDS->szOrderType,
 sizeof(dsD4205050.szOrderType));
 strncpy(dsD4205050.szOrderCompany,lpDS->szOrderCompany,
 sizeof(dsD4205050.szOrderCompany));
 dsD4205050.cUseCacheOrWF = lpDS->cUseCacheOrWF;
 strncpy(dsD4205050.szComputerID,lpDS->szComputerID,
 sizeof(dsD4205050.szComputerID));
 MathCopy(&dsD4205050.mnJobNumber,&lpDS->mnJobNumber);
 if (lpDSD4205040H->cActionCode != 'A')
 {
 dsD4205050.cCheckTableAfterCache = '1';
 }
 else
 {
 dsD4205050.cCheckTableAfterCache = '0';
 }
 jdeCallObject("GetSalesOrderDetailRecordOP",
 NULL,
 lpBhvrCom, lpVoid,
 (LPVOID)&dsD4205050,
 (CALLMAP *) NULL,
 (int) 0, (char *) NULL,
 (char *) NULL, (int) 0);

 if (dsD4205050.cRecordFound != '1')
 {
 bExit = TRUE;
 lpDS->cErrorCode = '1';
 /* Sales Order Detail Not Found */
 strncpy(lpDS->szErrorMessageID,"4162",
 sizeof(lpDS->szErrorMessageID));
 if (lpDS->cSuppressError != '1')
 {
 jdeErrorSet (lpBhvrCom, lpVoid, (ID) 0, "4162", (LPVOID) NULL);
 }
 }

 while ((dsD4205050.cRecordFound == '1') && (bExit == FALSE))
 {
 lpDSD4205050D = (LPDSD4205050D)jdeRemoveDataPtr(hUser,
 (ulong)dsD4205050.idDetailRecord);
 /* Reset flags */
 cPromisableLine = '0';
 bB4001040Called = FALSE;

 /*-----*/

```

```

/* Evaluate the Record from F4211 (cDataSource = 2)*/
/* to find out if we should promise the line */
/* else find out from Order Promising Detail.*/

if(dsD4205050.cDataSource == '1')
{
 if (lpDSD4205050D->cOPPromiseLineYN == 'Y')
 {
 cPromisableLine = '1';
 }
}
else if(dsD4205050.cDataSource == '2')
{
 MathCopy (&dsD4001040.mnShortItemNumber,
 &lpDSD4205050D->mnShortItemNumber);
 strncpy (dsD4001040.szBranchPlant,
 lpDSD4205050D->szBusinessUnit,
 sizeof(dsD4001040.szBranchPlant));

 jdeCallObject ("GetItemMasterDescUOM",
 NULL,
 lpBhvrCom, lpVoid,
 (LPVOID)&dsD4001040,
 (CALLMAP *) NULL,
 (int) 0, (char *) NULL,
 (char *) NULL, (int) 0);

 bB4001040Called = TRUE;

 cPromisableLine = I4205010_IsLinePromisable(lpBhvrCom,lpVoid,
 hUser,lpDS,lpDSD4205050D,
dsD4001040.cStockingType);
}
if (cPromisableLine == '1')
{
 /* Set this flag if at least one promisable */
 /* detail record exists. */
 bAtLeastOneDetail = TRUE;

 if (bB4001040Called == FALSE)
 {
 MathCopy (&dsD4001040.mnShortItemNumber,
 &lpDSD4205050D->mnShortItemNumber);
 strncpy (dsD4001040.szBranchPlant,
 lpDSD4205050D->szBusinessUnit,
 sizeof(dsD4001040.szBranchPlant));

 jdeCallObject ("GetItemMasterDescUOM",
 NULL,
 lpBhvrCom, lpVoid,
 (LPVOID)&dsD4001040,
 (CALLMAP *) NULL,
 (int) 0, (char *) NULL,
 (char *) NULL, (int) 0);
 }
}

```

```

I4205010_PopulateQueryDetail(lpDS,&dsD4205010B,
 lpDSD4205050D,
 &dsD4001040,
 &dsD4205010A,
 &dsD0100042,
 cPromisableLine,
 hUser,
 lpVoid,
 lpBhvrCom);

#ifdef jdeXAPI_CALLS_ENABLED
if(bXAPIInUse == TRUE)
{
 eXAPICallReturn = jdeXAPI_Add(lpBhvrCom,
 ulXAPICallID,
 "SendOrderPromiseRequest",
 "D4205010B",
 &dsD4205010B,
 sizeof(DSD4205010B));
 if (eXAPICallReturn != eEventCallSuccess)
 {
 bExit = TRUE;
 }
}
#endif
}

/*-----*/
/* Fetching the next Detail Record */

MathCopy(&dsD4205050.mnOrderNumber,&lpDS->mnOrderNumber);
strncpy(dsD4205050.szOrderType,lpDS->szOrderType,
 sizeof(dsD4205050.szOrderType));
strncpy(dsD4205050.szOrderCompany,lpDS->szOrderCompany,
 sizeof(dsD4205050.szOrderCompany));
dsD4205050.cUseCacheOrWF = lpDS->cUseCacheOrWF;
strncpy(dsD4205050.szComputerID,lpDS->szComputerID,
 sizeof(dsD4205050.szComputerID));
MathCopy(&dsD4205050.mnJobNumber,&lpDS->mnJobNumber);
if (lpDSD4205040H->cActionCode != 'A')
{
 dsD4205050.cCheckTableAfterCache = '1';
}
else
{
 dsD4205050.cCheckTableAfterCache = '0';
}
jdeCallObject("GetSalesOrderDetailRecordOP",
 NULL,
 lpBhvrCom, lpVoid,
 (LPVOID)&dsD4205050,
 (CALLMAP *) NULL,
 (int) 0, (char *) NULL,
 (char *) NULL, (int) 0);

```

```

 }

 if (!bAtLeastOneDetail)
 {
 bExit = TRUE;
 lpDS->cErrorCode = '1';
 /* Sales Order Detail Not Found */
 strncpy(lpDS->szErrorMessageID, "4162",
 sizeof(lpDS->szErrorMessageID));
 if (lpDS->cSuppressError != '1')
 {
 jdeErrorSet (lpBhvrCom, lpVoid, (ID) 0, "4162", (LPVOID)
NULL);
 }
 }
 if (bExit == FALSE)
 {
 #ifdef jdeXAPI_CALLS_ENABLED
 if (bXAPIInUse == TRUE)
 {
 eXAPICallReturn = jdeXAPI_Finalize(lpBhvrCom,
 ulXAPICallID,
 "SendOrderPromiseRequest",
 "OrderPromiseCallback");
 if (eXAPICallReturn != eEventCallSuccess)
 {
 bExit = TRUE;
 }
 }
 #endif
 }

 /*-----*/
 /* Call B4205020 in Add Mode */

 if ((bExit == FALSE) &&
 (lpDS->cDisplayBeforeAcceptMode != '1') &&
 (lpDS->cUseCacheOrWF == '2'))
 {
 MathCopy (&dsD4205020.mnOrderNumber, &lpDS->mnOrderNumber);
 strncpy (dsD4205020.szOrderType, lpDS->szOrderType,
 sizeof (dsD4205020.szOrderType));
 strncpy (dsD4205020.szOrderCompany, lpDS->szOrderCompany,
 sizeof (dsD4205020.szOrderCompany));
 strncpy (dsD4205020.szComputerID, lpDS->szComputerID,
 sizeof (dsD4205020.szComputerID));
 MathCopy (&dsD4205020.mnJobNumber, &lpDS->mnJobNumber);

 jdeCallObject ("MaintainOPWorkFile",
 NULL,
 lpBhvrCom, lpVoid,
 (LPVOID) &dsD4205020,
 (CALLMAP *) NULL,
 (int) 0, (char *) NULL,
 (char *) NULL, (int) 0);
 }

```

```

 }
}

/*****
* Function Clean Up
*****/
#ifdef jdeXAPI_CALLS_ENABLED
if (eXAPICallReturn != eEventCallSuccess)
{
 /*-----*/
 /* CleanUp */

 if(bXAPIInUse == TRUE)
 {
 jdeXAPI_Free(lpBhvrCom,
 ulXAPICallID,
 "SendOrderPromiseRequest");
 }

 lpDS->cErrorCode = '1';
 /* System Error - no reasonable error messages exist. */
 strncpy(lpDS->szErrorMessageID, "018Y",
 sizeof(lpDS->szErrorMessageID));
 if (lpDS->cSuppressError != '1')
 {
 jdeErrorSet (lpBhvrCom, lpVoid, (ID) 0, "018Y", (LPVOID) NULL);
 }
}
#endif

if(lpDSD4205040H != (LPDSD4205040H) NULL)
{
 jdeFree((void *)lpDSD4205040H);
}

if(lpDSD4205050D != (LPDSD4205050D) NULL)
{
 jdeFree((void *)lpDSD4205050D);
}

return (ER_SUCCESS);
}

```

### Example: XAPI Response from Originator System

The following illustrates the XAPI response document from the PeopleSoft EnterpriseOne executor system to the EnterpriseOne originator system:

```

<?xml version="1.0" encoding="UTF-16" ?>
<jdeResponse pwd="4f3e65076f446c5d20666f4172536518435c" role="*ALL"
type="realTimeEvent" user="PP6954083" session="35087181.1050101193"
environment="DV9NIS2" responseCreator="XAPI">
 <event>
 <header>
 <eventVersion>1.0</eventVersion>

```

```

<type>XAPIDEMO</type>
<user>PP6954083</user>
<role>*ALL</role>
<application>P90701XT</application>
<version />
<sessionID>35087181.1050101193</sessionID>
<environment>DV9NIS2</environment>
<host>DEN-PP6954083B</host>
<sequenceID>DEN-PP6954083B_2864_041120031636402</sequenceID>
<date>04112003</date>
<time>164646</time>
<scope />
<codepage>utf-8</codepage>
<instanceInfo>
 <host>DEN-PP6954083B</host>
 <port>6025</port>
 <type>JDENET</type>
</instanceInfo>
</header>
<body elementCount="2">
 <detail date="04112003" name="XAPITestFunctionInitiateRequest"
time="16:39:54" type="" DSTMPL="D907001A" executionOrder="0" parameterCount="14">
 <szXMLHandle type="String" />
 <szNameAlpha type="String">Pradip Pandey</szNameAlpha>
 <szNameMailing type="String">Pradip K Pandey</szNameMailing>
 <szAddressLine1 type="String" />
 <szAddressLine2 type="String" />
 <szZipCodePostal type="String">80237</szZipCodePostal>
 <szCity type="String">Denver</szCity>
 <szState type="String">CO</szState>
 <szCountry type="String" />
 <mnAmountGross type="Double">100.00</mnAmountGross>
 <mnUnits type="Double">100.00</mnUnits>
 <jdDtForGLAndVouch1 type="Date">2001/01/01</jdDtForGLAndVouch1>
 <cDefaultAddressLine1 type="Character" />
 </detail>
 <detail date="04112003" name="XAPITestFunctionInitiateRequest" time="16:39:54"
type="" DSTMPL="DXAPIROUTE" executionOrder="1" parameterCount="4">
 <ClientPort type="Int">6024</ClientPort>
 <ClientIP type="Int">168007331</ClientIP>
 <ClientMagicNumber type="Int">1</ClientMagicNumber>
 <XAPIMethodID type="String">XAPITestResponse</XAPIMethodID>
 </detail>
</body>
</event>
</jdeResponse>

```

## XAPI Inbound Response Handling APIs

The following APIs are used to retrieve XML data from the inbound XAPI document and generate an inbound XAPI response (originator system). The APIs that are listed below are the same as those APIs identified in the XAPI Inbound Response section.

- jdeXML\_GetDSCount
- jdeXML\_GetDSName
- jdeXML\_ParseDS
- jdeXML\_DeleteXML
- jdeXML\_ParseNextDSByName
- jdeXML\_PrepareDSLListForIterationByName

### See Also

The following examples in the *Interoperability Guide*:

- ❑ *Example: XAPI Response Parsing API Usage*
- ❑ *Example: Inbound XAPI Response*

## XAPI Error Handling APIs

The following APIs are for error handling in the executor system:

- jdeXML\_CheckSystemError  
The check system error API is for system errors. It tells the PeopleSoft EnterpriseOne originator system that a system error happened in the EnterpriseOne executor system.
- jdeXML\_GetErrorCount
- jdeXML\_SetErrors  
The get error count and set errors APIs are for business errors. These two APIs, when used together, find the number of business errors and then send the errors to the BHVRCOM structure for you to resolve.

## Generating XAPI Request and Response

To generate XAPI requests from the PeopleSoft EnterpriseOne originator system and replies from the EnterpriseOne executor system, you perform the following tasks:

- Set up the OCM.
- Define the XAPI event (F90701).
- Set up subscription information (F90702).
- Configure the server jde.ini file.

---

**Note**

Setting up the OCM, defining the XAPI event, and setting up subscription information tasks are the same as for the tasks discussed in *XAPI Outbound Events* in the *Interoperability Guide*.

---

The jde.ini file configuration is discussed below:

In addition to the above setup tasks, you must do the following:

- Map the business function (F907012).
- Modify the element name for XML documents.

### **Configure the jde.ini File for XAPI Originator and Executor**

To generate XAPI events, the following sections of the PeopleSoft EnterpriseOne server jde.ini file must be configured:

- [JDENET\_KERNEL\_DEF19]
- [JDENET\_KERNEL\_DEF20]
- [JDENET\_KERNEL\_DEF22]
- [JDENET\_KERNEL\_DEF24]
- [JDEITDRV]
- [XAPI] – XMLDirectory setting
- [XMLLookupInfo]
- [INTEROPERABILITY] – LEVEL setting

```
[XAPI]
XMLDirectory=c:\builds\bdev\log\

[XMLLookupInfo]
XMLRequestType5=XAPICallMethod
XMLKernelMessageRange5=14251
XMLKernelHostName5=local
XMLKernelPort5=0
XMLKernelReply5=0

XMLRequestType6=realTimeEvent
XMLKernelMessageRange6=14251
XMLKernelHostName6=local
XMLKernelPort6=0
XMLKernelReply6=0

[INTEROPERABILITY]
LEVEL=DOC
```

---

**Note**

The LEVEL setting is for logging the Event XML document in the EnterpriseOne server for debugging purposes.

Setting the LEVEL=DOC key causes all real-time events to be written to the disk, which can cause a significant performance impact on the host server. PeopleSoft suggests that you not use the LEVEL=DOC setting in a production environment or for stress testing of the QA environment.

---

If you are using an EnterpriseOne client to generate XAPI events, you must define the Client Dispatch kernel and [JDENET} sections of the client jde.ini file.

**See Also**

See the following topics in the *Interoperability Guide*:

- ❑ *jde.ini File Configurations for Events* for information about setting kernel configurations for the host server
- ❑ *Configure the Client jde.ini File for XAPI* for information about setting kernel and JDENET configurations for the EnterpriseOne client

**Map the Business Function for XAPI Request and Response**

When the PeopleSoft EnterpriseOne executor system receives an event from the EnterpriseOne originator, it needs to know what business function or system API to invoke to process the XAPI request. You must map the business function or system API to the XAPI event name. You map business functions and system APIs in the Event Request Definition table (F907012).

If you are mapping business functions, you enter the name of the business function. If you map APIs, you must enter the name of the API and the library where it is defined. In addition, the signature of the API must be made common, similar to the business function.

Mapping business functions allows you to point a XAPI event to a business function or system API that you wrote. You do not need to modify source code of a business function that PeopleSoft delivered to you.

You use the Event Request Definition program (P907012) to map business functions and APIs.

---

**► To set up handler information**

---

*Enter P907012 on the fast path command line to access the Work with Definition form.*

1. On Work with Definition, click Add.

P907012 - [Request Definition]

File Edit Preferences Window Help

OK Cancel Dismiss Abort Links Display OLE ... Internet

Event Name XAPISOOU

☐ API Definition ☒ BSFN Definition

Function Name DmdSchXAPISalesOrderProcess

2. On Request Definition, complete the following fields:
  - Event Name
  - Function Name
3. Choose one of the following options:
  - API Definition
  - BSFN Definition

---

**Note**

If you choose API definition, the DLL Name field appears on the form. Complete the DLL Name field.

---

4. Click OK to save your entries.
5. Click Cancel to return to Work with Definition.
6. Click Close to return to the main menu.

## Modify Element Name for XML Documents

Before XAPI PeopleSoft EnterpriseOne processing, any document sent out from EnterpriseOne was considered to be a response document, and any document coming in to EnterpriseOne was considered to be a request document. However, with XAPI, request documents are generated by the EnterpriseOne originating system and can be sent to an EnterpriseOne executor system. Response documents are generated and sent out by the EnterpriseOne executor system and received by the EnterpriseOne originating system. To support XAPI and to enable the XML dispatch kernel to be able to distinguish between a response and reply, the following new “Type” attributes were created to be used with the `jdeResponse` element:

|                                         |                                                                                                                                                     |
|-----------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>jdeResponse=RealTimeEvent</code>  | Use this element and attribute to identify a XAPI request from the EnterpriseOne originating system and sent to the EnterpriseOne executor system.  |
| <code>jdeResponse=xapicallmethod</code> | Use this element and attribute to identify a XAPI response from the EnterpriseOne executor system and sent to the EnterpriseOne originating system. |

When the XMLDispatch kernel receives a document with the `jdeResponse` element and one of the above “Type” attributes, XMLDispatch sends the document to the XMLService kernel. XMLService can distinguish a response or a reply based on the “Type” attribute associated with the `jdeResponse` element and then processes the document appropriately.

## Security for XAPI Originator and Executor

Access to the PeopleSoft EnterpriseOne originator and EnterpriseOne executor systems is based on the following:

- User ID
- Password
- Environment
- Role

The EnterpriseOne originating system verifies that the above security information is valid and creates an `Huser` object with an encrypted password to send to the EnterpriseOne executor system. Encryption APIs (`jdeEnchyper/jdeDecypher`) are used to encrypt and decode the password. The security information is sent in the XAPI request XML document.

---

### Note

The user ID, password, environment, and role must be the same on both EnterpriseOne systems (originator and executor).

---

## Error Processing for XAPI Originator and Executor

The following two kinds of errors might be encountered during XAPI error processing between two PeopleSoft EnterpriseOne systems:

- Business-related errors  
The business function or the business function specs cannot be found.
- System errors  
These errors occur in other parts of the system—for example, message delivery failure.

XAPI error processing for business-related errors are handled in the following ways:

- The XAPI system logs business-related errors in the EnterpriseOne server log. The errors are delivered as part of the XAPI reply.
- XAPI APIs parse business errors from the response document.
- XAPI logs all information available about the error in the EnterpriseOne server log.

---

## jde.ini File Configurations for Events

The PeopleSoft EnterpriseOne server jde.ini file must be properly configured to support Z, real-time, and XAPI event generation. You use a text editor to manually edit and verify specific settings in the EnterpriseOne server jde.ini file.

---

### Note

If your enterprise contains more than one EnterpriseOne server, you must ensure that each server has the same settings for all logic, batch, and interoperability sections.

---

Use the following kernel and [JDEITDRV] settings to configure the jde.ini file on your EnterpriseOne server. Configure the kernels that are appropriate for the type of event (Z, real-time, or XAPI) that you want to generate.

---

### Note

To determine which kernels you need to set and for other jde.ini settings for each specific type of event, refer to the *Configure the jde.ini File* topics in the Events section of the *Interoperability Guide*.

---

```
[JDENET_KERNEL_DEF19]
krnlName=EVN KERNEL
dispatchDLLName=jdeie.dll
dispatchDLLFunction=_JDEK_DispatchITMessage@28
maxNumberOfProcesses=1
numberOfAutoStartProcesses=0
```

```
[JDENET_KERNEL_DEF20]
krnlName=IEO KERNEL
dispatchDLLName=jdeieo.dll
dispatchDLLFunction=_JDEK_DispatchIEOMessage@28
maxNumberOfProcesses=1
numberOfAutoStartProcesses=0
```

```
[JDENET_KERNEL_DEF22]
krnlName=XML Dispatch KERNEL
dispatchDLLName=xmldispatch.dll
dispatchDLLFunction=_XMLDispatch@28
maxNumberOfProcesses=1
numberOfAutoStartProcesses=0
```

```
[JDENET_KERNEL_DEF24]
krnlName=XML Service KERNEL
dispatchDLLName=xmlservice.dll
dispatchDLLFunction=_XMLServiceDispatch@28
maxNumberOfProcesses=1
numberOfAutoStartProcesses=0
```

```
[JDEITDRV]
DrvCount=5
Drv1=Z:zdrv.dll
Drv2=RT:rtdrv.dll
Drv3=JDENET:jdetdrv.dll
Drv4=MSMQ:msmqtrdrv.dll
Drv5=MQS:mqsrtdrv.dll
```

---

**Note**

You set event generation and transport drivers in the [JDEITDRV] section of the jde.ini file. You are not required to set all of these drives. For example, if you do not use messaging transports, you would not use the MSMQ and MQS settings. Be sure that you define DrvCount with the number of settings that you are using.

---

```
[JDENET]
MaxKernelRanges=27
```

---

**Note**

The MaxKernelRanges setting specifies the maximum number of JDENET kernels. You must set this value to encompass the total number of kernels that you defined.

---

The above settings are for Windows 2000 and NT. The following table shows the settings for other platforms:

|                                                  | <b>iSeries</b>          | <b>HP9000B</b>          | <b>Sun or RS6000</b>    |
|--------------------------------------------------|-------------------------|-------------------------|-------------------------|
| <b>EVN (19)<br/>dispatchDLLName</b>              | JDEIE                   | libjdeie.sl             | libjdeie.so             |
| <b>EVN (19)<br/>dispatchDLLFunction</b>          | JDEK_DispatchITMessage  | JDEK_DispatchITMessage  | JDEK_DispatchITMessage  |
| <b>IEO (20)<br/>dispatchDLLName</b>              | JDEIEO                  | libjdeieo.sl            | libjdeieo.so            |
| <b>IEO (20)<br/>dispatchDLLFunction</b>          | JDEK_DispatchIEOMessage | JDEK_DispatchIEOMessage | JDEK_DispatchIEOMessage |
| <b>XML Dispatch (22)<br/>dispatchDLLName</b>     | XMLDSPATCH              | libxmldispatch.sl       | libxmldispatch.so       |
| <b>XML Dispatch (22)<br/>dispatchDLLFunction</b> | JDEK_XMLDispatch        | JDEK_XMLDispatch        | JDEK_XMLDispatch        |
| <b>XML Service (24)<br/>dispatchDLLName</b>      | XMLSERVICE              | libxmlservice.sl        | libxmlservice.so        |
| <b>XML Service (24)<br/>dispatchDLLFunction</b>  | JDEK_XMLServiceDispatch | JDEK_XMLServiceDispatch | JDEK_XMLServiceDispatch |
| <b>Drv1</b>                                      | RTDRV                   | librtdrv.sl             | librtdrv.so             |
| <b>Drv2</b>                                      | ZDRV                    | libzdrv.sl              | libzdrv.so              |
| <b>Drv3</b>                                      | JDETRDRV                | libjdetdrv.sl           | libjdstdrv.so           |
| <b>Drv4</b>                                      | MSMQRTDRV               | libmsmqrtdrv.sl         | libmsmqrtdrv.so         |
| <b>Drv5</b>                                      | MQSRTDRV                | libmqsrtdrv.sl          | libmqsrtdrv.so          |

---

## Defining Events

You use the Event Request Definition program (P90701) to add new single and container events and to review your existing events. You add single events by event name. When you add a single event, you must include a data structure. A container event contains either single events or aggregate events or both. When you add a container event, you define events, single events to be used individually, or data structures, single events to be aggregated. You can change the information for single and container events. You can delete single and container events. You can change the status of an event to active or non-active. If your system has multiple environments, the event status is the same in all environments. You can use menu options to access the subscriber information.

When an event is generated, the IEO kernel reads the Interoperability Event Definition table (F90701) for that event. If the specified event category is different from the event category configured in the database, the system writes an error to the log file, as follows:

- If the database definition of the event is not found, the following error messages are written to the EVN log:  
Error in getNextNumberF0002: failed to open F0002 table.  
Warning: table F90702 doesn't exist. Some features will be turned off.
- If the database tables are missing, the following message is written to the EVN log:  
CheckTableExists failed: invalid hEnv or hUser.
- And the following message is written to the IEO log.  
Warning: table %s doesn't exist. Some features will be turned off.

---

### Reminder

When you update or modify the F90701 table, you must restart the server for the changes to become effective.

---

### ► To add a single event

---

*From the Interoperability menu (GH9070), choose Interoperability Event Definition or enter P90701 in the Fast Path to access the Event Definition Workbench form.*

1. On Event Definition Workbench, click Add.

The screenshot shows a software window titled "Interoperability Event Definition - [Event Entry]". The window has a menu bar with "File", "Edit", "Preferences", "Form", "Window", and "Help". Below the menu bar is a toolbar with icons for "OK", "Can...", "Dis...", and "Add". The main area of the window contains a form with the following fields and values:

| Field             | Value              |
|-------------------|--------------------|
| Event Name        | RTSODTL            |
| Event Description | Sales Order Detail |
| Event Type        | RTE                |
| Event Category    | SINGLE             |
| Product Code      | H42                |
| Reliable Delivery | 1                  |
| Timeout Threshold | 0                  |
| Data Structure    | D4202310B          |

At the bottom right of the form, there is a label "Reliable" and another label "Sales Order Detail". The window has a status bar at the bottom with a small globe icon.

2. On Event Entry, complete the following fields:

- Event Name
- Event Description
- Event Type
- Event Category
- Product Code
- Reliable Delivery
- Timeout Threshold
- Data Structure

---

**Note**

If you are using the Reliable Event Delivery feature, you must set the Reliable Delivery field to reliable (1 or Y) and the Timeout Threshold field must be set. The Timeout Threshold field is in seconds and applies only to the reliable events for which an initial delivery attempt fails. This field determines the maximum amount of time that has to pass from the event creation to the time when the event will be discarded if not delivered successfully. Events with a threshold of zero never expire.

---

3. Click OK to save your updates.

---

**Note**

If you are adding XAPI events, the system automatically completes the Event Category field with Container and, after you click OK, the Event Definition Detail form appears. Complete the Data Structure and Data Description fields, and then click OK.

---

4. Click Cancel to return to Event Definition Workbench.
5. On Event Definition Workbench, click Find to view your events.
6. Click Close to return to the main menu.

---

**► To add a container event**

*From the Interoperability menu (GH9070), choose Interoperability Event Definition; or enter P90701 in the Fast Path to access the Event Definition Workbench form.*

1. On Event Definition Workbench, click Add.
2. On Event Entry, complete the following fields:
  - Event Name
  - Event Description
  - Event Type
  - Event Category

- Product Code
- Reliable Delivery
- Timeout Threshold

---

**Note**

If you are using the Reliable Event Delivery feature, you must set the Reliable Delivery field to reliable (1 or Y) and the Timeout Threshold field must be set. The Timeout Threshold field is in seconds and applies only to the reliable events for which an initial delivery attempt fails. This field determines the maximum amount of time that has to pass from the event creation to the time when the event will be discarded if not delivered successfully. Events with a threshold of zero never expire.

---

3. Click OK to access Event Definition Detail.
4. On Event Definition Detail, click one of the following options:
  - Event Data
  - Data Structure Data

---

**Note**

Choose Event Data to define single individual (composite) events for the container event. Choose Data Structure Data to define aggregate events for the container event.

---

5. Do one of the following:
  - If you chose Event Data:
    - Click the visual assist in the first empty line under the Single Event field in the detail area.
    - On Event Search & Select, choose an event, and then click Select.
  - If you chose Data Structure Data:
    - Click the visual assist in the first empty line under the Data Structure field in the detail area.
    - On Individual Object Search and Select, choose a data structure, and then click Select.

Repeat this step as many times as necessary to link the appropriate single events to the container event.

6. To save your changes and return to Event Entry, click OK.
7. On Event Entry, click Cancel to return to Event Definition Workbench.
8. On Event Definition Workbench, click Find to view your events.
9. Click Close to return to the main menu.

► **To view or change the definition of a single event**

---

*From the Interoperability menu (GH9070), choose Interoperability Event Definition; or enter P90701 in the Fast Path to access the Event Definition Workbench form.*

1. On Event Definition Workbench, complete the following fields:
  - Event Name
  - Description
  - Event Type
  - Product Code
2. Click one of the following options, and then click Find to display existing events.
  - Active Statuses
  - Non Active Statuses
  - All Statuses
3. Choose a single event, and then click Select.
4. On Event Entry, view or change the following fields:
  - Event Description
  - Event Type
  - Product Code
  - Reliable Delivery
  - Data Structure
5. Click OK to save your changes and return to Event Definition Workbench.
6. On Event Definition Workbench, click Close to return to the main menu.

► **To change a single event to a container event**

---

*From the Interoperability menu (GH9070), choose Interoperability Event Definition; or enter P90701 in the Fast Path to access the Event Definition Workbench form.*

1. On Event Definition Workbench, complete the following fields:
  - Event Name
  - Description
  - Event Type
  - Product Code
2. Click one of the following options, and then click Find to display existing events.
  - Active Statuses
  - Non Active Statuses
  - All Statuses

3. Choose the single event that you want to change to a container event, and then click Select.
4. On Event Entry, change the following field:
  - Event Category
5. Click OK to access Event Definition Detail.
6. On Event Definition Detail, click the visual assist in the first empty line under the Single Event field in the detail area.
7. On Event Search & Select, choose an event, and then click Select.  
Repeat steps 6 and 7 as many times as necessary to link the appropriate single events to the container event.
8. Click OK to save your changes and return to the Event Definition Workbench.
9. On Event Definition Workbench, click Find to view your events.
10. Click Close to return to the main menu.

► **To view or change the definition for a container event**

---

*From the Interoperability menu (GH9070), choose Interoperability Event Definition; or enter P90701 in the Fast Path to access the Event Definition Workbench form.*

1. On Event Definition Workbench, complete the following fields:
  - Event Name
  - Description
  - Event Type
  - Product Code
2. Click one of the following options, and then click Find to display existing events.
  - Active Statuses
  - Non Active Statuses
  - All Statuses
3. Choose the detail line that contains the event to be viewed or changed, and then click Select.
4. On Event Entry, view or change the following fields:
  - Event Description
  - Event Type
  - Product Code
  - Reliable Delivery
5. Click OK to save any changes and access Event Definition Detail.

6. On Event Definition Detail, do one of the following:
  - View the single events associated with the container event.
  - Add a single event to the container event.
    - In the detail area, click the visual assist in the empty row under the Single Event field.
    - When you click the visual assist, the Event Search & Select form appears. Choose an event type and then click Select. Repeat this process for each single event type that you want to link with the container event.
  - Change an existing single event that is associated with the container event.
    - In the detail area, choose the detail line that contains the event to be changed, and then click the visual assist that appears in the Single Event field.
    - When you click the visual assist, the Event Search & Select form appears. Choose an event type, and then click Select to return to the Event Definition Detail form. The row you chose will be updated.
  - Delete a single event from the container event.
    - In the detail area, choose the detail line that contains the event to be deleted, and then click delete.
    - Confirm the deletion.
7. Click OK to save changes and return to Event Definition Workbench.
8. On Event Definition Workbench, click Cancel to return to the main menu.

► **To change a container event to a single event**

---

*From the Interoperability menu (GH9070), choose Interoperability Event Definition; or enter P90701 in the Fast Path to access the Event Definition Workbench form.*

1. On Event Definition Workbench, complete the following fields:
  - Event Name
  - Description
  - Event Type
  - Product Code
2. Click one of the following options, and then click Find to display existing events.
  - Active Statuses
  - Non Active Statuses
  - All Statuses
3. Choose the detail line that contains the container event that you want to change to a Single event, and then click Select.
4. On Event Entry, change the following field:
  - Event Category

5. Click OK.  
A warning message indicating the change will result in loss of detail appears.
6. On the warning message, click OK.
7. To complete the Data Structure field, click the visual assist.
8. On Individual Object Search and Select, choose a data structure, and then click Select.
9. On Event Definition Detail, click OK to save your changes and return to Event Definition Workbench.
10. On Event Definition Workbench, click Cancel to return to the main menu.

---

► **To change the status of an event**

---

*From the Interoperability menu (GH9070), choose Interoperability Event Definition. Alternatively, enter P90701 on the fast path command line to access the Event Definition Workbench form.*

1. On Event Definition Workbench, complete the following fields:
  - Event Name
  - Description
  - Event Type
  - Product Code
2. Click the following option, and then click Find to display existing events.
  - All Statuses
3. In the detail area, choose the event for which you want to change the status.
4. From the Row menu, choose Change Status.
5. To view the status change, click Find.
6. Click Close to return to the main menu.

---

**Note**

The status of the event is the same for all environments. If the event is active, that event is active for all environments. If the event is non-active, that event is non-active for all environments.

---

---

► **To copy an event**

---

*From the Interoperability menu (GH9070), choose Interoperability Event Definition; or enter P90701 in the Fast Path to access the Event Definition Workbench form.*

1. On Event Definition Workbench, complete the following fields:
  - Event Name
  - Description
  - Event Type
  - Product Code

2. Turn on one of the following options, and then click Find.
  - Active Statuses
  - Non Active Statuses
  - All Statuses
3. Choose the detail line that contains the event to be copied, and then click Copy.
4. On the Event Entry form, complete the following field:
  - Event Name
5. Do one of the following:
  - If you copied a single event, modify the definition for the new event.

---

**Note**

See Steps 4 through 6 of the *To view or change the definition of a single event task* in the *Interoperability Guide*.

---

- If you copied a container event, modify the definition for the new event.

---

**Note**

See Steps 4 through 8 of the *To view or change the definition for a container event task* in the *Interoperability Guide*.

---

---

**► To delete an event**

---

*From the Interoperability menu (GH9070), choose Interoperability Event Definition; or enter P90701 in the Fast Path to access the Event Definition Workbench form.*

1. On Event Definition Workbench, complete the following fields:
  - Event Name
  - Description
  - Event Type
  - Product Code
2. Turn on the following option, and then click Find.
  - Active Statuses
3. Choose the detail line that contains the event to be deleted, and then click Delete.

If you are deleting a container event, you must confirm the deletion.
4. To verify the event is deleted, click Find.
5. Click Close to return to the main menu.

---

## Setting Up Subscriber Information

For XAPI events, you must update the Interoperability Subscriber Enrollment table (F90702) so that you can receive a response to your XAPI event. Each XAPI event must have a logical subscriber, which you might have to set up. For Z and real-time events, the system dynamically updates this table when the event is created. You can use this table to view the persistent subscriptions for your Z and real-time events. You use the Interoperability Event Subscription program (P90702) to add new subscription information and to review and change existing subscription information. You can also add a new subscription by copying and then modifying an existing subscription, and you can delete subscriptions.

The F90702 table contains subscriber information, such as the machine name and port number, and is read by EVN. If subscriber information is missing for the XAPI event, the event is generated but cannot be delivered. You set up a logical subscriber by choosing Logical Subscriber from the Form menu.

You can access and view your real-time and XAPI event definitions by choosing Event Definition from the Form menu. You can also access and view Z events by clicking the Z File Events icon on the Subscriber Workbench form or by choosing the Z File Events option on the Form menu.

---

### Reminder

When you add to or modify the F90702 table, you must restart the server for the changes to become effective.

---

---

### ► To add a logical subscriber

---

*From the Interoperability menu (GH9070), choose Interoperability Subscriber Enrollment; or enter P90702 in the Fast Path to access the Subscriber Workbench form.*

1. On Subscriber Workbench, choose Logical Subscriber from the Form menu.
2. On Work With Logical Subscriber Names, click Add.

Interoperability Subscriber Enrollment - [Logical Subscriber Entry]

File Edit Preferences Window Help

OK Cancel Dismiss Abort Links ▼ Displ... OLE ... Internet

Logical Subscriber Name INTNT2

Event Transport Driver JDENET JDENET

Host Name INTNT2

Port Number 6005

Subscriber Group CommonGroup

3. On Logical Subscriber Entry, complete the following fields:
  - Logical Subscriber Name  
Do not use spaces in the Logical Subscriber Name.
  - Event Transport Driver
  - Host Name
  - Port Number
4. Enter CommonGroup in the following field:
  - Subscriber Group
5. Click OK to save your updates.
6. Click Cancel to return to Work With Logical Subscriber Names.
7. Click Close to return to Subscriber Workbench.

► **To add a subscription**

---

*From the Interoperability menu (GH9070), choose Interoperability Subscriber Enrollment or enter P90702 in the Fast Path to access the Subscriber Workbench form.*

1. On Subscriber Workbench, click Add.

The screenshot shows a software window titled "Interoperability Subscriber Enrollment - [Subscriber Entry]". The window has a menu bar with "File", "Edit", "Preferences", "Form", "Window", and "Help". Below the menu bar is a toolbar with icons for "OK", "Can...", "Dis...", "Add", "Links", "Logic...", "OLE ...", and "Internet". The main area of the window contains several labeled text input fields:

- Event Subscriber: CA6859852
- Event Environment: ADEVNIS2 (with "APPL-NT, INTEL, SQL C=ON" displayed to the right)
- Purpose: Adjust Demand Outbound
- Logical Subscriber Name: INTNT2
- Event Type: XAPI
- Event Name: XAPIADOUT
- Event Description: XAPI Adjust Demand Outbound
- Event Filter Name: FILTER0

At the bottom right of the window, there is a small globe icon.

2. On Subscriber Entry, complete the following fields:
  - Event Subscriber
  - Event Environment
  - Purpose
  - Logical Subscriber Name
  - Event Type
3. The system automatically enters Filter0 for the following field:
  - Event Name
4. Click OK to save your updates.
5. Click Cancel to return to Subscriber Workbench.
6. On Subscriber Workbench, click Find to view your subscription.
7. Click Close to return to the main menu.

► **To view or change a subscription**

---

*From the Interoperability menu (GH9070), choose Interoperability Subscriber Enrollment; or enter P90702 in the Fast Path to access the Subscriber Workbench form.*

1. On Subscriber Workbench, complete the following fields:
  - Subscriber Name
  - Purpose
2. Click one of the following options, and then click Find to display existing subscriptions.
  - Active Status
  - Non-Active Status
  - All Statuses
3. Choose the subscription you want to view or change, and then click Select.
4. On Subscriber Entry, view or change either of the following fields:
  - Purpose
5. Click OK to save your updates and return to Subscriber Workbench.
6. On Subscriber Workbench, click Find to view the subscriptions.
7. Click Close to return to the main menu.

► **To change the status of a subscription**

---

*From the Interoperability menu (GH9070), choose Interoperability Subscriber Enrollment; or enter P90702 in the Fast Path to access the Subscriber Workbench form.*

1. On Subscriber Workbench, complete the following fields:
  - Subscriber Name
  - Purpose
2. Turn on the following option, and then click Find to display existing subscriptions.
  - All Statuses
3. In the detail area, choose the event for which you want to change the status.
4. From the Row menu, choose Change Status.
5. To view the status change, click Find.
6. Click Close to return to the main menu.

### ► To copy a subscription

---

*From the Interoperability menu (GH9070), choose Interoperability Subscriber Enrollment; or enter P90702 in the Fast Path to access the Subscriber Workbench form.*

1. On Subscriber Workbench, complete the following fields:
  - Subscriber Name
  - Purpose
2. Turn on one of the following options, and then click Find to display existing subscriptions.
  - Active Status
  - Non-Active Status
  - All Statuses
3. Choose the detail line that contains the event to be copied, and then click Copy.
4. On Subscriber Entry, update the following fields:
  - Event Subscriber
  - Event Environment
  - Purpose
  - Logical Subscriber Name
  - Event Type
  - Event Name
5. Click OK to return to Subscriber Workbench.
6. On Subscriber Workbench, click Find to view your subscriptions.
7. Click Close to return to the main menu.

### ► To delete a subscription

---

*From the Interoperability menu (GH9070), choose Interoperability Subscriber Enrollment; or enter P90702 in the Fast Path to access the Subscriber Workbench form.*

1. On Subscriber Workbench, complete the following fields:
  - Subscriber Name
  - Purpose
2. Turn on the following option, and then click Find.
  - All Statuses
3. Choose the detail line that contains the subscription to be deleted, and then click Delete.
4. To verify the event is deleted, click Find.
5. Click Close to return to the main menu.

---

## Reliable Event Delivery

Reliable Event Delivery supports Z events, real-time events, and XAPI events. To use the Reliable Event Delivery feature, you must define your events in database tables. You cannot define your events in the `jde.ini` file.

The JDENET transport delivers Z events, real-time events, and XAPI events. Reliable event delivery ensures recovery and delivery of an event when transport problems arise, including some network problems. The following scenarios identify circumstances for which events might be lost, but can be recovered and delivered:

- JDENET process is down.
- JDENET fails to deliver because the network link between sender and receiver is permanently down.
- JDENET fails to deliver because the IPC buffer of the receiving kernel is full (sender and receiver are on different boxes).

---

### Caution

Reliable delivery covers failures related to the transport of the events only. Reliable delivery does not provide a “once and only once”-type guarantee. Events might be lost and not recovered (or duplicates redelivered) in the presence of process failures (client and server).

---

Real-time event delivery is reliable for transportation failures between the real-time API and the Java connector, which includes IEO and EVN kernels. XAPI outbound event delivery is reliable for transportation failures between the XAPI API and the Java connector, including the IEO and EVN kernels. Z event delivery is reliable for transportation failures between the Z event generator and the Java connector.

The level of reliability is configurable based on whether the event is reliable or volatile. Volatile events are events that might be lost if the network or process fails and delivery is not reliable. Reliable events could be lost in the case of process failures only. You can configure the level of reliability for every event type. The level of reliability depends on whether the event is a business critical event. For example, you might configure an inquiry as volatile, because an inquiry is not a critical business event and you don't want the system to continually look for the event should the event fail. You might configure a purchase order as reliable, because this is a critical business event and you do want the system to continually look for the event and make the transaction update. Volatile events offer better performance than reliable events, but delivery is not reliable if the event is lost during transportation.

Real-time and XAPI events can be single, aggregate, or composite events. A composite event consists of single events. The composite event and the single events that make up the composite event can have different levels of reliability. For example, you register composite events as RTSOOUT with a level of reliability as reliable, and you register single events as RTSOLINE with a level of reliability as volatile. The level of reliability configured for RTSOOUT will not override the level of reliability configured for RTSOLINE. The rationale for this is that the reliability of events is based on the event type. If you decide that single event types are not important enough to configure as reliable delivery, then the single events that are created during composite event creation should have the same level of reliability as other single events.

The APIs you use to create real-time and XAPI events are not affected by the level of reliability.

## Configure Your System for Reliable Event Delivery

To use the reliable event delivery feature, you must define your events in the Single and Container Event Definition (F90701) table. Use the Interoperability Event Definition (P90701) program to accomplish this task. On the Event Entry form, the Reliable Delivery field must be set to reliable (either Y or 1), and the Threshold Timeout field must be set. The Timeout Threshold field is in seconds and applies only to the reliable events for which an initial delivery attempt fails. This field determines the maximum amount of time that has to pass from the event creation to the time when the event is going to be discarded if not delivered successfully. Events with a threshold of zero never expire.

Two database tables, Event Protocol (F90704) table and Event Link (F90703) table, enable communication between the sender and receiver. Event Protocol stores information that is related to the protocol that delivers an event. Event Link stores information that is related to the reliable event for which initial delivery failed. These tables are updated by the system when an event is created.

---

### Caution

Both the sender and receiver must access the same instances (the data sources are the same) of the interoperability database tables.

---

---

### Note

If the reliable event is not found, the following message might be generated in the client, Callobject, IEO, and EVN logs:

RDEL0000045 – Could not open tables for reliable event delivery (F90703 and F90704).  
Reliable event delivery will be disabled.

If you receive this error message, verify your events are defined in the Single and Container Event Definition (F90701) table, that the Reliable Delivery and Threshold Time fields are set as discussed above, and that the Event Protocol and Event Link tables exist.

---

---

## Events Self-Diagnostic Utility Tool

The Events Self-Diagnostic Utility Tool supports Z events and real-time events. Normally, your System Administrator runs the Self-Diagnostic Utility Tool to verify that your events infrastructure features are functional. The Self-Diagnostic Utility Tool can be used on the following platforms:

- Windows 2000 and NT
- iSeries
- HP
- Sun
- AIX

The Events Self-Diagnostic Utility Tool analyzes the infrastructure of an event and reports configuration, kernel, and network problems that are detected as the event is processed through your system. You can use the tool to perform a comprehensive analysis, or you can configure the tool to perform an analysis that is specific for your needs. The Events Self-Diagnostic Tool uses XML comparator to compare XML documents to detect the presence of any data corruption in event information. The tool also suggests actions that you can take to resolve problems. You can run this tool on either a server or a client or both.

## Events Self-Diagnostic Utility Tool Process Overview

After an event is generated at the call object API on the server or the application API on the client, problems that cause the event to fail can occur. Problems that might occur include the following:

- The jde.ini file has a configuration error.
- The ZEVG library is unavailable or the IEO or EVN kernel process is down.
- Subscribers and supported events have not loaded successfully.
- One or more of the kernels involved in the event delivery is corrupting the event information.
- The network link between any or all of the components involved in this infrastructure is permanently down.

When the Events Self-Diagnostic Tool detects a problem, the tool sends messages to you explaining the problem and suggesting resolutions and also logs the error in the appropriate log files. The message that is sent to you indicates which log files you should review. The following list provide some examples of how the Events Self-Diagnostic Tool detects problems:

- Performs an in-depth interoperability-oriented analysis of the jde.ini file.
- Reads the Interoperability Event Definition table (F90701) to determine whether the event is defined.
- Reads the Interoperability Subscriber Enrollment table (F90702) to determine whether the persistent subscription/un-subscription request, which is sent to the EVN kernel by the tool, is successful.
- Reads the Object Configuration Manager to find the location of the IEO kernel. In this process, the tool ensures there is only one active entry for the RTE object.
- Checks interconnectivity within events infrastructure by sending self-diagnostic connectivity message calls.
- Generates self-diagnostic events to test different services offered by the infrastructure and to verify event information against possible data corruption.

---

### Note

The above list is general and not all-inclusive.

---

# Events Self-Diagnostic Utility Tool Components

The Events Self-Diagnostic Utility Tool consists of three components:

- Event generator
- Event receiver
- XML comparator

## Event Generator

The Events Self-Diagnostic Utility Tool starts with an event generator process. During startup, the event generator performs basic background analysis of the events infrastructure, which includes the following:

- Verification of interoperability specific sections of the jde.ini file
- Verification of real-time events definition
- Inter-component connectivity check within the events infrastructure

If startup is successful, the event generator tests different features offered by the events infrastructure. These features include generating and testing different types of events, listing the valid events, checking the event template, and testing subscription information. You can run one or more of these tests by using one of the following methods:

- Running the test against an existing configuration file that you previously set up
- Running the test against a new configuration file, which you will set up
- Choosing options and executing the test from the command line menu of the tool

After successful generation of a self-diagnostic event, the event is passed through the event infrastructure system. To test the accuracy of the event information being conveyed through the system, the event generator attaches an additional packet, in the form of XML stream, to the event. The diagnostic XML packet contains information about the event. At each stage of communication, each kernel (or component) verifies the event information by comparing standard message packets with the self-diagnostic XML packet. The kernel (or component) logs the result of this comparison at each point of comparison in respective log files. The accuracy of the information in template requests is tested the same way.

## Event Receiver

The event receiver acts as a NULL transport driver that subscribes itself for self-diagnostic events during EVN kernel startup. The event receiver compares and verifies the XML documents contained in the received self-diagnostic events. The event receiver logs the result of this comparison in the EVN kernel log file.

## XML Comparator

The event generator uses the XML comparator tool to test the accuracy of event information or of an event template request that is being passed through the system. The XML comparator compares any two given XML documents for equivalency, similarity, or both. To perform the comparison, the XML comparator requires three XML documents. Two of the documents are the actual XML documents to be compared. The third document is an exclusion XML document that contains nodes that are to be ignored during the comparison of the two given XML documents.

## Setting Up Database Tables for Self-Diagnostic Events Generation

If you use database tables to generate events, the Interoperability Event Definition (F90701) table must contain the self-diagnostic events. You use the Interoperability Event Definition (P90701) program to add both container and single events.

## Executing the Event Self-Diagnostic Tool

To use the Event Self-Diagnostic Tool, you must have a valid PeopleSoft EnterpriseOne user ID, password, environment, and role. If you are using the tool from an EnterpriseOne server and you do not supply this information as parameters, the username, password, environment, and role information is read from the security section of the server jde.ini file. If you are using a client, you must enter a valid EnterpriseOne username, password, environment, and role. If you do not enter this information, the tool will stop. If you are generating events from a client, you must also have a valid OCM mapping for RTE or Z events to a valid server.

### Prerequisite

- ❑ Ensure PORTTEST runs successfully on your system.
- ❑ Ensure that one instance of the IEO and EVN kernel is running.

### Start the Tool

To start the Events Self-Diagnostic Tool on the PeopleSoft EnterpriseOne Server, double-click the executable file at the following location:

`$system\bin32\sdtool.exe`

Or you can pass parameters, as follows:

`$system\bin32\sdtool.exe username password environment role`

To start the tool from the client side, you must include parameters as follows:

`$system\bin32\sdtool.exe username password environment role`

---

### Note

`$system` refers to the path where the application is installed on your system.

---

The Events Self-Diagnostic Tool accesses the Security section of the jde.ini file for a valid username, password, environment, and role. Upon startup, the tool analyzes the jde.ini file, verifies that events are defined, and checks the inter-component connectivity within the events infrastructure. As the tool analyzes each of these areas, it provides you with feedback about what is being analyzed and whether the analysis was successful.

If the tool detects a problem in any one of the startup areas, the tool terminates the diagnosis and sends you a message explaining the problem encountered and suggesting actions for resolving the problem.

After successful startup, you have a choice of creating and using a customized configuration file or using the command line of the tool to run the diagnosis. The Customize(d) Tool option allows you to build and save a diagnostic test to a file so that you can run that test as often as needed without having to re-enter information into the tool. When you use the Command Line Execution option, you must enter the test information when the tool prompts you. When you run tests from the command line, the Interface menu always follows the results statements so that you can run another test or exit the tool.

Whether you choose Option 1, Customize(d) Tool, or Option 2, Command Line Execution, the tests that the tool performs are the same.

You can choose one or more tests by typing the number associated with the test at the prompt and then pressing Enter or Return. For multiple tests, separate the number of the test with a comma (,). Some of the tests provide further options. At the prompt, you enter one or more options, using a comma to separate multiple options. The tool performs the test and provides feedback to you indicating success or failure. If the test failed, the tool provides feedback that tells you that the test failed and identifies the logs you should review for more information.

## **Generate/Test Real-Time Event**

When you choose Action 1, Generate/Test Real Time Event(s), the tool displays the real-time event types from which you choose one or more real-time event types to test.

The tool generates the real-time event you requested and attaches a self-diagnostic XML document to the event. The event contents are verified for any data corruption against the attached XML document at each kernel in the events infrastructure and event receiver transport driver. You receive a message indicating whether the event was successfully generated. You also receive the following feedback message: Please see log files corresponding to IEO and EVN for any present event data corruption. This message tells you to look at the log files to determine whether an XML document mismatch occurred. The tool also provides a final message that indicates that the tool has completed the analysis for that action, and then it returns you to the tool interface menu.

## **Generate/Test Z Event**

When you choose Action 2, Generate/Test Z Event, the tool displays another set of options. You can test a simulated Z event and you can test a Z event that uses the actual interface tables (Z tables). If you choose to test a simulated Z event, the tool generates a simulated Z event and attaches a self-diagnostic XML document to the event. The event contents are verified for any data corruption against the attached XML document at the EVN kernel and event receiver transport driver. You receive a message indicating whether the event was successfully generated. You also receive the following feedback message: Please see log files corresponding to EVN for any present data corruption. This message tells you to look at the EVN log file to determine whether an XML document mismatch occurred. The tool also provides a final message that indicates that the self-diagnostic tool has completed the analysis for the action, and then it returns you to the tool interface menu.

If you generate an actual Z event, you must have a valid UBE and you must set up the appropriate interface tables. The tool asks you for your user name, batch number, transaction number, line number, transaction type, document type, and sequence number. The tool uses the live interface tables (Z tables) for this test. When you request an actual Z event, the tool generates the Z event but does not include a self-diagnostic XML document. The EVN kernel returns a message indicating whether the event was successful. Because no self-diagnostic XML document exists, the tool cannot diagnose data corruption.

## **Test All Types of Events**

When you choose Action 3, Test all types of events, from the Execution menu, the tool tests all of the real-time events (single, aggregate, and composite) and both Z events (simulated and actual). Action 3 is the same as Action 1 (all three options) and Action 2 (both options) combined. For the real-time events and the simulated Z event, the tool generates the event and attaches a self-diagnostic XML document to the event so that any data corruption can be detected. If you choose this action, you must have a valid UBE and you must set up appropriate interface tables. If you run this test but do not have actual Z event data, you can abort that portion of the test by entering Exit when the tool asks for the Z event information.

The tool sends the event information to the IEO and EVN kernels, and the kernels return messages indicating whether each event was successful.

## **Get Event List**

When you choose Action 4, Get Event List (List of events supported), from the Execution menu, the tool immediately runs the test.

The tool sends a `getEventList` request to the EVN kernel. The EVN kernel responds with the list of event names that you have defined. To validate that the list is complete, the tool checks the list for the presence of self-diagnostic event names. The tool provides a list of the events to you along with a message indicating whether the test was successful.

## **Get Event Template**

When you choose Action 5, Get Event Template, the tool displays the real-time event types that have a template associated with them.

The tool generates the template request and attaches a self-diagnostic XML document to the request. The template request is verified for any data corruption against the attached XML document at each kernel in the events infrastructure and event generator. The tool provides feedback that the template request was successful and that the template data was validated against the XML packet. If the test fails, the tool provides a message that indicates the reason for the failure.

## **Subscription Services**

When you choose Action 6, Subscription Services, the tool displays a set of options for the type of subscription service to be tested.

When you choose the Persistent Subscribe option, the tool sends a persistent subscription request for a registered self-diagnostic event to the EVN kernel. The tool verifies the list of subscribers that are maintained in a file or from the database table (depending on how your system is configured), and then sends you a message indicating whether the test was successful.

When you chose the Persistent Unsubscribe option, the tool sends a persistent un-subscription request for a registered self-diagnostic event to the EVN kernel. The tool verifies the subscription is no longer in the file or database table (depending on how your system is configured), and then sends you a message indicating whether the test was successful.

When you choose the Non-Persistent Subscribe option, the tool sends a non-persistent subscription request for a registered self-diagnostic event to the EVN kernel. The tool verifies the list of subscribers that is kept by the EVN kernel, and then sends you a message indicating whether the test was successful.

When you choose the Non-Persistent Unsubscribe option, the tool sends non-persistent un-subscription request for a registered self-diagnostic event to the EVN kernel. The tool verifies the subscription is no longer in EVN, and then sends you a message indicating whether the test was successful.

## **Comprehensive System Analysis**

When you choose Action 7, Comprehensive System Analysis, the tool performs all of the tests and provides messages to you indicating whether each test was successful.

## **Customized Tool**

When you choose the Customize(d) Tool option from the Interface menu, the tool prompts you to provide a new file name or to use an existing configuration file (one that you previously created using this tool). The tests (actions) and options for each test are the same tests that are discussed previously.

To use an existing configuration file (an .xml file that your previously created), type the filename at the prompt and press Enter or Return. The diagnosis against your previously built configuration file starts.

To create a new configuration file, type a filename using .xml as the extension, and then press Enter or Return. The tool offers the same tests that are on the Command Line Execution Menu. You can choose one or more tests by using a comma to separate the test numbers.

# Batch Interfaces

You usually use a batch interface to collect transactions over a period of time and then process all of the transactions at once. PeopleSoft EnterpriseOne supports the following batch interface model types:

- Interface Tables
- Electronic Data Interface (EDI)
- Table Conversions
- Output Stream Access (OSA) UBEs
- Advanced Planning Agent (APAg)/Integration

---

## Interface Tables

An interface table (also called a Z table) is a working table where non-EnterpriseOne information can be stored and then processed into EnterpriseOne. You can also use interface tables to retrieve EnterpriseOne data. EnterpriseOne interface tables mirror EnterpriseOne application tables. The following capabilities can be used with interface tables:

- Business Functions
- Z Transactions
- Flat Files

EnterpriseOne provides predefined interface tables for transactions for some applications. You can also create your own interface tables as long as your interface table is formatted in accordance with EnterpriseOne standards.

### See Also

- ❑ *Interface Tables* in the *Interoperability Guide* for a list of predefined interface tables and processes

## Interface Table Structure

Each PeopleSoft EnterpriseOne transaction uses a set of interface tables. Some files share a common set of interface tables. The interface table name is based on the EnterpriseOne application table name and has Z1 as a suffix. For example, if the application table were F4211, the interface table would be F4211Z1.

Use the following guidelines to determine the based-on table:

- Inbound is based on the application table that is updated with data from the interface table.
- Outbound is based on the application table that has data extracted from it and placed in the interface table.

Both the inbound and outbound directions of an internal transaction within a system use a single set of interface tables. For example, in the Sales Order system, for a sales order, the inbound Customer Order (850) and the outbound Order Acknowledgment (855) share a set of interface tables.

If the interface table is used for both inbound and outbound transactions, the based-on table should be the same application table. In the above Sales Order example with an inbound Customer Order and an outbound Order Acknowledgment, the detail interface table would be based on table F4211.

If the interface table exceeds 250 columns or has a record length greater than 1968, an additional interface table is needed for the remaining columns. Columns in the additional interface table should contain infrequently used data. The additional interface table is named after the primary interface table with a letter, starting with A, after the Z1 suffix. For example, if the primary interface table were F4211Z1, the additional table would be F4211Z1A.

The beginning of the table has the following columns, which act as control fields:

- User ID (EDUS) - key field
- Batch Number (EDBT) - key field
- Transaction Number (EDTN) - key field
- Line Number (EDLN) - key field
- Document Type (EDCT)
- Transaction Type (TYTN)
- Translation Format (EDFT)
- Transmission Date (EDDT)
- Direction Indicator (DRIN)
- Number of Detail Lines (EDDL)
- Processed (EDSP)
- Trading Partner ID (PNID)
- Action Code (TNAC)

You must use the above key structure.

The end of the table has the following columns, which are reserved for user and audit fields:

- User Reserved Code (URCD)
- User Reserved Date (URDT)
- User Reserved Amount (URAT)
- User Reserved Number (URAB)
- User Reserved Reference (URRF)
- Transaction Originator (TORG)

- User ID (USER)
- Program ID (PID)
- Work Station ID (JOBID)
- Date Updated (UPMJ)
- Time of Day (TDAY)

The middle of the table has all of the columns from the based-on application table, excluding user reserved and audit field columns. An exception to this is when the interface table is near the 250-column limit or the 1968-record length limit. In this case, columns from the application table that most likely will not be needed should be excluded.

Prefixes for the table columns are SY for the header and SZ for the detail.

Change or match interface tables, such as a cash receipt or purchase receipt, might require additional columns that correspond to user input capable controls on an interactive form.

A header table is not required for every transaction.

---

#### Note

If you create custom interface tables, use the structure and format discussed above.

---

## Inbound Interface Table Processing

You use interface tables to import non-EnterpriseOne transactions into the live EnterpriseOne database. Inbound interface tables are based on the EnterpriseOne application table where the transaction will be stored. EnterpriseOne can process the records from the inbound interface tables in the following ways:

- Run an inbound batch process.
- Use a business function.
- Use a subsystem to trigger the transaction.

---

#### Note

The non-EnterpriseOne transactions are referred to as *Z* transactions, and specific information about processing information in to EnterpriseOne is provided in *Z Transactions* in the *Interoperability Guide*.

---

#### See Also

- ❑ *Z Transactions* in the *Interoperability Guide* for information about transferring non-EnterpriseOne transactions into EnterpriseOne

## Outbound Interface Table Processing

You can use interface tables to retrieve information from PeopleSoft EnterpriseOne. Outbound interface tables are based on the EnterpriseOne application table from where the data is extracted. You can retrieve records from EnterpriseOne in any of the following ways:

- Run an extraction batch process.
- Use a vendor-specific outbound batch process.
- Use the subsystem business function.

### See Also

- ❑ *Z Events* in the *Interoperability Guide* for information about retrieving information from PeopleSoft EnterpriseOne by setting outbound master business function processing options to process one transaction at a time

### Run an Extraction Batch Process

Batch transactions from PeopleSoft EnterpriseOne allow you to place a large number of transactions into an interface table and then process all of the transactions at once. You copy the records from the EnterpriseOne application tables to the EnterpriseOne outbound interface tables using the extraction batch process that is specifically set up for the type of document you are sending.

You use the Solution Explorer to initiate the extraction batch process for applications that support extraction batch processing. The extraction batch process displays a version list of report features. You can run an existing version, change an existing version, or add a version. You can also change the processing options and data selection options for that version to fit your needs.

When you run the extraction batch process, the program retrieves data from the EnterpriseOne application tables for the transaction and copies the data into the interface tables. The system also generates an audit report that lists which documents were processed. Errors are placed on the audit report and also sent to the Employee Work Center. You can use the Revisions Application to add, change, or delete records in the interface table.

#### ► To run the Extraction Batch Process

---

*Start the Extraction Batch Process for the transaction type you want to export. For example, Work Order Routing has an extraction batch process. The Interoperability Interface Table Information at the end of this section identifies programs that have an extraction batch process.*

1. On Version Prompting, click any of the following to review the report feature options:
  - Data Selection
  - Data Sequencing
2. Click the Submit button.

The Extraction Batch Process retrieves data from the PeopleSoft EnterpriseOne application tables and copies the data into the outbound interface tables.

The Extraction Batch Process program generates an audit report that lists the documents that completed successfully.

## Subsystem Business Function

You can use the generic outbound subsystem business function, Add Transaction To Subsystem Queue (B0000176), to write a record to the subsystem data queue to specify a batch process that needs to be awakened in the subsystem. This business function starts processing of a batch of one (single transaction). The business function also passes keys to the subsystem data queue.

The data structure for the outbound transaction is as follows:

|                  |      |
|------------------|------|
| Line number      | EDLN |
| Transaction type | TYTN |
| Document type    | DCTO |
| Action code      | TNAC |

## Interface Table Maintenance

If you receive an error message when the interface table is processed, you can use a revision application to make corrections to the data and then reprocess the data in batch or transaction mode. After you have successfully processed the data in the interface table, you should run a purge application to remove all records from the interface table and to any remove secondary interface tables from the system.

### Revision Application

A Revision Application is used to add, delete, edit, and review transactions in the interface tables. Use the revision application that is appropriate for the type of transaction you are processing. You can use a revision application to correct the record in error. You then run the process again, continuing to make corrections and rerunning the transaction process until the program completes without errors. When deleting records, a revisions application calls the appropriate purge named event rule to delete records. The name is based on the detail interface table. For example, if the tables for Sales Order Entry were F4201Z1 and F4211Z1, the application would be called P4211Z1.

### See Also

- ❑ *Interoperability Interface Table Information* in the *Interoperability Guide* for a list of Revision Applications that are available for you to use with the EnterpriseOne predefined interface tables

### Purge Interface Table Information

You should run a purge batch process periodically after you have successfully processed the data in the interface tables. The purge batch process should have one or two sections; the number of sections depends on the interface tables. The purge batch process calls the purge named event rule (NER). The name of the purge batch process is based on the revisions application with a P suffix. For example, if the revisions application is P4211Z1, the purge batch process is R4211Z1P.

Purge NERs have two modes:

- Header mode, which deletes the header record and all associated records in independent tables
- Detail mode, which deletes the detail record and all associated records in dependent tables

The purge NER is named after the purge batch process. Only eight characters are allowed for the NER name. If the name has nine characters using these standards, remove the P suffix. For example, if the purge batch process is R4211Z1P, the purge NER will be N4211Z1P.

When a "before image" for net change is deleted, the corresponding "after image" is also deleted. When an "after image" is deleted, the corresponding "before image" is also deleted.

### See Also

- ❑ *Interoperability Interface Table Information* in the *Interoperability Guide* for a list of the Purge Batch Processes

---

## Interoperability Interface Table Information

| Application                   | Interface Table (Z table)   | Input Subsystem Batch Process | Input Processor Batch Process | Extraction Batch Process | Revisions Application | Purge Batch Process | Application with Processing Options |
|-------------------------------|-----------------------------|-------------------------------|-------------------------------|--------------------------|-----------------------|---------------------|-------------------------------------|
| <b>Financials</b>             |                             |                               |                               |                          |                       |                     |                                     |
| Address Book                  | F0101Z2                     | R01010Z - ZJDE0002            | R01010Z - ZJDE0001            |                          | P0101Z1               | R0101Z1P            | P0100041                            |
| Customer Master               | F03012Z1                    | R03010Z - ZJDE0002            | R03010Z - ZJDE0001            |                          | P0301Z1               | R0101Z1P            | P0100042                            |
| Supplier Master               | F0401Z1                     | R04010Z - ZJDE0002            | R04010Z - ZJDE0001            |                          | P0401Z1               | R0101Z1P            | P0100043                            |
| A/R Invoice                   | F03B11Z1, F0911Z1, F0911Z1T | R03B11Z1I                     | R03B11Z1I - ZJDE0001          | N/A                      | P03B11Z1              | R03B11Z1P           | N/A                                 |
| A/P Invoice                   | F0411Z1, F0911Z1            | R04110Z - ZJDE0002            | R04110Z - ZJDE0001            | N/A                      | P0411Z1               | R0411Z1P            | N/A                                 |
| Payment Order with Remittance | F0413Z1, F0414Z1            | N/A                           | N/A                           |                          | P0413Z1               | R0413Z1             | P0413M                              |
| Journal Entry                 | F0911Z1, F0911Z1T           | R09110Z - ZJDE0005            | R09110Z - ZJDE0002            |                          | P0911Z1               | R0911Z1P            | N/A                                 |
| Fixed Asset Master            | F1201Z1, F1217Z1            | R1201Z1I - XJDE0002           | R1201Z1I - XJDE0001           | R1201Z1X                 | P1201Z1               | R1201Z1P            | P1201                               |

| Application                           | Interface Table (Z table)  | Input Subsystem Batch Process | Input Processor Batch Process | Extraction Batch Process | Revisions Application | Purge Batch Process | Application with Processing Options |
|---------------------------------------|----------------------------|-------------------------------|-------------------------------|--------------------------|-----------------------|---------------------|-------------------------------------|
| Account Balance                       | F0902Z1                    | N/A                           |                               | N/A                      | P0902Z1               | R0902ZP             | N/A                                 |
| Batch Cash Receipts                   | F03B13Z1                   | N/A                           | R03B13Z1I - ZJDE0001          | N/A                      |                       |                     | N/A                                 |
|                                       |                            |                               |                               |                          |                       |                     |                                     |
| <b>HRM</b>                            |                            |                               |                               |                          |                       |                     |                                     |
| Payroll Time Entry                    | F06116Z1                   | R05116Z1I                     | R05116Z1I - ZJDE0001          | N/A                      | P05116Z1              | R05116Z1P           | N/A                                 |
|                                       |                            |                               |                               |                          |                       |                     |                                     |
| <b>Distribution</b>                   |                            |                               |                               |                          |                       |                     |                                     |
| Purchase Order                        | F4301Z1, F4311Z1           | R4311Z1I - XJDE0002           | R4311Z1I - XJDE0001           |                          | P4311Z1               | R4301Z1P            | P4310                               |
| Outbound Purchase Receipts            | F43121Z1                   | N/A                           | N/A                           |                          | P43121Z1              | R43121Z1P           | P4312                               |
| Receipt Routing                       | F43092Z1                   | R43092Z1I - XJDE0002          | R43092Z1I - ZJDE0001          |                          | P43092Z1              | R43092Z1P           | P43250                              |
| Outbound Sales Order                  | F4201Z1, F4211Z1, F49211Z1 | N/A                           | N/A                           |                          | P4211Z1               | R4211Z1P            | P4210                               |
| Outbound Shipment Confirmation        | F4201Z1, F4211Z1, F49211Z1 | N/A                           | N/A                           |                          | P4211Z1               | R4211Z1P            | P4205                               |
|                                       |                            |                               |                               |                          |                       |                     |                                     |
| <b>Logistics</b>                      |                            |                               |                               |                          |                       |                     |                                     |
| Cycle Counts                          | F4141Z1                    | R4141Z1I                      | R4141Z1I - ZJDE0001           | N/A                      | P4141Z1               | R4141Z1P            | N/A                                 |
| Item Master                           | F4101Z1, F4101Z1A          | R4101Z1I                      | R4101Z1I - ZJDE0001           |                          | P4101Z1               |                     | P4101                               |
| Item Cost                             | F4105Z1                    | N/A                           | R4105Z1I - XJDE0001           |                          | P4105Z1               | R4105Z1P            | P4105                               |
| Warehouse Confirmations (Suggestions) | F4611Z1                    | R4611Z1I                      | R4611Z1I - ZJDE0001           |                          | P4611Z1               | R4611Z1P            | N/A                                 |

| Application                    | Interface Table (Z table) | Input Subsystem Batch Process | Input Processor Batch Process | Extraction Batch Process | Revisions Application | Purge Batch Process | Application with Processing Options           |
|--------------------------------|---------------------------|-------------------------------|-------------------------------|--------------------------|-----------------------|---------------------|-----------------------------------------------|
|                                |                           |                               |                               |                          |                       |                     |                                               |
| <b>Manufacturing</b>           |                           |                               |                               |                          |                       |                     |                                               |
| Work Order Header              | F4801Z1                   | Use Work Order Completions    | Use Work Order Completions    | R4101Z1O                 | P4801Z1               | R4801Z1P            | P48013                                        |
| Work Order Parts List          | F3111Z1                   | Use Planning Messages         | Use Planning Messages         |                          | P4801Z1               | R3111Z1P            | P3111                                         |
| Work Order Routing             | F3112Z1                   | Use Planning Messages         | Use Planning Messages         | R4801Z2X                 | P4801Z1               | R3112Z1P            | P3112                                         |
| Work Order Employee Time Entry | F31122Z1                  | R31122Z1I - XJDE0002          | R31122Z1I - XJDE0001          |                          | P31122Z1              | R31122Z1            | P311221                                       |
| Work Order Inventory Issues    | F3111Z1                   | R31113Z1I - ZJDE0002          | R31113Z1I - ZJDE0001          |                          | P3111Z1               | R3111Z1P            | N/A                                           |
| Work Order Completions         | F4801Z1                   | R31114Z1I - XJDE0002          | R31114Z1I - XJDE0001          |                          | P4801Z1               | R4801Z1P            | N/A                                           |
| Super Backflush                | F3112Z1                   | R31123Z1I                     | R31123Z1I - ZJDE0001          |                          | P3112Z1               | R3112Z1P            | N/A                                           |
| Bill of Material               | F3002Z1                   | R3002Z1I - ZJDE0002           | R3002Z1I - ZJDE0001           |                          | P3002Z1               | R3002Z1P            | P3002                                         |
| Routing Master                 | F3003Z1                   | R3003Z1I - ZJDE0002           | R3003Z1I - ZJDE0001           |                          | P3003Z1               | R3003Z1P            | P3003                                         |
| Work Center Master             | F30006Z1                  | R30006Z1I - ZJDE0002          | R30006Z1I - ZJDE0001          |                          | P30006Z1              | R30006Z1P           | P3006                                         |
| Work Day Calendar              | F0007Z1                   | R0007Z1I - XJDE0002           | R0007Z1I - XJDE0001           |                          | P0007Z1               | R0007Z1P            | P00071                                        |
| Planning Messages              | F3411Z1                   | R3411Z1I - ZJDE0002           | R3411Z1I - ZJDE0001           |                          | P3411Z1               | R3411Z1P            | N/A                                           |
| Detail Forecast                | F3460Z1                   | R3460Z1I - XJDE0002           | R3460Z1I - XJDE0001           |                          | P3460Z1               | R3460Z1P            | P3460, R3465, R34650 (Each done individually) |

| Application         | Interface Table (Z table) | Input Subsystem Batch Process | Input Processor Batch Process | Extraction Batch Process | Revisions Application | Purge Batch Process | Application with Processing Options |
|---------------------|---------------------------|-------------------------------|-------------------------------|--------------------------|-----------------------|---------------------|-------------------------------------|
| Kanban Transactions | F30161Z1                  | R30161Z1I - XJDE0002          | R30161Z1I - XJDE0001          | N/A                      | P30161Z1              | R30161Z1P           | N/A                                 |

---

## Electronic Data Interface (EDI)

The PeopleSoft EnterpriseOne Data Interface for Electronic Data Interchange system acts as an interface between the EnterpriseOne system data and the translator software. In addition to exchanging EDI data, this data interface can also be used for general interoperability and electronic commerce needs where a file-based interface meets the business requirements.

### See Also

- ❑ *Overview for Data Interface for Electronic Data Interchange System* and other topics in the *Data Interface for Electronic Data Guide* for information about using electronic data interchange for interoperability

---

## Table Conversion

Table conversion is a special form of Universal Batch Engine (UBE) that allows you to do high-speed manipulation of data in tables. PeopleSoft EnterpriseOne has a table conversion utility that you can use to gather, format, export, and import enterprise data. The table conversion tool allows you to transfer and copy data and to delete records from tables.

### See Also

- ❑ *Table Conversions Overview* and other topics in the *Table Conversion Guide* for more information about table conversions

---

## Output Stream Access (OSA) UBEs

If you have set up an OSA interface, you can pass PeopleSoft EnterpriseOne data to another software program for processing and formatting. OSA can use its own set of commands or it can use an XML library.

### See Also

- ❑ *Output Stream Access* in the *EnterpriseOne Report Writing Guide* for information about using OSA for processing PeopleSoft EnterpriseOne data in another software program

---

## Advanced Planning Agent (APAg)/Integration

The PeopleSoft EnterpriseOne Advanced Planning Agent (APAg) is a tool for batch extracting, transforming, and loading of data. APAg supports access to data sources in the form of relational databases, flat file format, and other data or message encoding, such as XML. APAg also moves data from one place to another and initiates tasks related to the movement of the data.

### See Also

- *APAg Integrates Your System* and other topics in the *Advanced Planning Agent 3\_3\_3 User Guide* for information about the APAg tool

# Open Data Access (ODA)

The PeopleSoft EnterpriseOne Open Data Access (ODA) ODBC driver is a read-only driver that is compliant with version 2.5 or higher. Front-end Windows Query and Reporting tools can use ODA to access the EnterpriseOne database. ODA supports the following front-end tools:

- Microsoft Query
- Microsoft Access
- Microsoft Excel
- ODBCTEST
- Crystal Report
- Microsoft Analysis Service (not certified)

ODA sits between the front-end Query/Reporting tools and the EnterpriseOne-configured ODBC drivers.

The EnterpriseOne database contains object and column names, specific data types and security rules that must be converted or applied so that the data is presented correctly. The specific data types and rules include decimal shifting, Julian date, currency, media object, security, and user defined codes. In some instances, ODA modifies the SQL SELECT statement, as well as the data, so that it appears correctly within the chosen tool.

---

## Note

PeopleSoft EnterpriseOne Open Data Access was formerly known as J.D. Edwards Open Data Access. References to PeopleSoft EnterpriseOne or EnterpriseOne within this document may appear as J.D. Edwards in the software.

---

---

## Hardware and Software Requirements

To use ODA, you must meet the following hardware and software requirements.

### Hardware Requirements

- IBM-compatible personal computer
- Hard disk with 6 MB of free disk space
- At least 16 MB of random access memory (RAM)

## Software Requirements

To access data with the ODA driver, your system must meet the minimum technical requirements (MTR) for PeopleSoft EnterpriseOne. MTRs are updated for each release and are available online. You must also meet the following software requirements:

- PeopleSoft EnterpriseOne
- The EnterpriseOne Open Data Access driver (JDEOWODA.dll)
- The 32-bit ODBC Driver Manager, version 3.0 or later (ODBC32.dll) (this file is included with the ODBC Database Drivers)
- Microsoft Windows 95 or later, or Windows NT 4.0 or later

The use of the ODA ODBC driver by 16-bit applications on Windows 95 is not supported.

---

## ODBC Component Files

The PeopleSoft EnterpriseOne installation installs the components required by ODBC database drivers. You might also find the following additional files:

| Driver          | File Name    |
|-----------------|--------------|
| ODA Driver      | JDEOWODA.DLL |
| ODA Driver Help | JDEOWODA.HLP |
| Release Notes   | README.TXT   |

---

### Note

OLEDDB is a driver for SQL Server. However, OLEDDB data source is not supported for ODA. If you are using ODA with SQL Server, use ODBC to set up your data source.

---

---

## Open Data Access (ODA) Driver Architecture

The PeopleSoft EnterpriseOne ODA ODBC driver architecture has five components:

- Application  
Front-end Query/Reporting tool that calls the ODA driver to access data from the EnterpriseOne database.
- Manager  
Loads and unloads drivers on behalf of an application. Processes ODBC calls or passes them to the ODA driver.

- **EnterpriseOne ODA Driver**  
Passes some of the ODBC requests directly to the vendor's ODBC driver. If specific data types for EnterpriseOne are used, then the SQL SELECT statement is modified before sending it to the vendor's ODBC driver. After the data is returned from the vendor's ODBC driver, the EnterpriseOne ODA ODBC driver might need to manipulate the data so that it displays correctly in the application.
- **Vendor Driver**  
Processes ODBC function calls and submits SQL requests to the specific data source. If necessary, the driver modifies an application's request so that the request conforms to the syntax supported by the associated DBMS.
- **Data Source**  
The data that you want to access as well as the operating system, DBMS, and network platform for the data.

---

## Adding an ODA Data Source

Although the ODA driver is automatically registered as part of the installation process, you might need to add a data source. You can also add a file data source or a system data source, modify a data source, or delete a data source if needed. You might also need to check the currency value check box when you configure the ODA driver so that you can view currency data in the correct format.

If you use Oracle, you must create another ODBC DSN, named OneWorld ODA Ora, so that you can access Oracle Data source through ODA. Specific information for doing this is included in the online release notes.

---

### ► To add a data source

1. Double-click the Control Panel icon. On Control Panel, double-click the ODBC icon.
2. On the User Data Sources dialog box, click Add.
3. On Add Data Source, choose the EnterpriseOne Open Data Access driver from the Installed ODBC Drivers list and click Finish.
4. On Configure Data Source, enter the following information to set up the data source and then click OK.
  - **Data Source Name**  
Specify the name that you want to call the EnterpriseOne Open Data Access driver.
  - **Description**  
Specify the description of the driver that you are adding. The Description entry cannot exceed 79 characters.

5. On the Connect form, turn on one or more of the following options:
  - **Convert User Defined Codes**  
Choose this option to return the associated description of the user defined field instead of the user defined code. The associated description is more descriptive because it is a text description instead of a code that is used for the user defined code. The default value is to display the associated description instead of the user defined code.
  - **Convert Currency Values**  
Choose this option to convert currency fields to the correct values.
  - **Use Long Table/Business View Names**  
Choose this option to view long table/view names.
  - **Use Long Column Names**  
Choose this option to view long column names.
6. On the Connect form, turn on one or more of the following Table/Business View Display Options:
  - **Tables Only**  
Choose this option to view only EnterpriseOne tables.
  - **Business Views Only**  
Choose this option to view only EnterpriseOne business views.
  - **Tables and Business Views**  
Choose this option to view both EnterpriseOne tables and EnterpriseOne business views.
7. To customize the list of functions that are enabled in ODA, click Advanced.  
Advanced configuration is optional. If you choose not to customize the list of functions enabled in ODA, a default list of settings is used.

---

## Adding a File Data Source

You can also add a file data source.

### ► To add a file data source

---

1. Double-click the Control Panel icon. On Control Panel, double-click the ODBC icon.
2. On User Data Sources click the DSN tab.
3. On File Data Sources, click Add.
4. On Add Data Source, choose the EnterpriseOne Open Data Access driver from the Installed ODBC Drivers list and click Finish.

5. On Configure Data Source, enter the following information to set up the data source and click OK.
  - Data Source Name  
Specify the name that you want to call the EnterpriseOne Open Data Access driver.
  - Description  
Specify the description of the driver that you are adding. The Description entry cannot exceed 79 characters.
6. On the Connect form, turn on one or more of the following options:
  - Convert User Defined Codes  
Choose this option to return the associated description of the user defined field instead of the user defined code. The associated description is more descriptive because it is a text description instead of a code that is used for the user defined code. The default value is to display the associated description instead of the user defined code.
  - Convert Currency Values  
Choose this option to convert currency fields to the correct values.
  - Use Long Table/Business View Names  
Choose this option to view long table/view names.
  - Use Long Column Names  
Choose this option to view long column names.
7. On the Connect form, turn on one or more of the following Table/Business View Display Options:
  - Tables Only  
Choose this option to view only EnterpriseOne tables.
  - Business Views Only  
Choose this option to view only EnterpriseOne business views.
  - Tables and Business Views  
Choose this option to view both EnterpriseOne tables and EnterpriseOne business views.
8. To customize the list of functions that are enabled in ODA, click Advanced.  
Advanced configuration is optional. If you choose not to customize the list of functions enabled in ODA, a default list of settings is used.

---

## Adding a System Data Source

A data source can be set up with a system data source name (DSN) that can be used by more than one user on the same machine. The system DSN can also be used by a system-wide service, which can then gain access to the data source even if no user is logged on to the machine.

---

### ► To add a system data source

---

1. Double-click the Control Panel icon. On Control Panel, double-click the ODBC icon.
2. On Data Sources, click the System DSN tab, and then click Add.
3. On System Data Sources, click Add.
4. On Add Data Source, choose the EnterpriseOne Open Data Access driver from the Installed ODBC Drivers list and click Finish.
5. On Configure Data Source, enter the following information to set up the data source and click OK.
  - Data Source Name - specify the name that you want to call the EnterpriseOne Open Data Access driver.
  - Description - specify the description of the driver that you are adding. (Note the Description entry cannot exceed 79 characters.)
6. On the Connect form, enable one or more of the following options:
  - Convert User Defined Codes - Use this option to return the associated description of the user defined field instead of the user defined code. The associated description is more descriptive, because it is a text description instead of a code that is used for the user defined code. The default is to display the associated description instead of the user defined code.
  - Convert Currency Values - Use this option to convert currency fields to the correct values.
  - Use Long Table/Business View Names - Use this option to view long table/view names.
  - Use Long Column Names - Use this option to view long column names.
7. On the Connect form, enable one or more of the following Table/Business View Display Options
  - Tables Only - Use this option to view only EnterpriseOne tables.
  - Business Views Only - Use this option to view only EnterpriseOne business views.
  - Tables and Business Views - Use this option to view both EnterpriseOne tables and EnterpriseOne business views.
8. To customize the list of functions that are enabled in ODA, click Advanced.

Advanced configuration is optional. If you choose not to customize the list of functions enabled in ODA, a default list of settings is used.

---

# Modifying a Data Source

You can also modify a data source.

## ► To modify a data source

---

1. Double-click the Control Panel icon. On Control Panel, double-click the ODBC icon.
2. On User Data Sources, File Data Sources, or System Data Sources, choose a data source from the available list.
3. Click Configure.
4. On Configure Data Source, enter the following information to set up the data source and then click OK.
  - **Data Source Name**  
Specify the name that you want to call the EnterpriseOne Open Data Access driver.
  - **Description**  
Specify the description of the driver that you are adding. The Description entry cannot exceed 79 characters.
5. On the Connect form, turn on one or more of the following options:
  - **Convert User Defined Codes**  
Choose this option to return the associated description of the user defined field instead of the user defined code. The associated description is more descriptive because it is a text description instead of a code that is used for the user defined code. The default is to display the associated description instead of the user defined code.
  - **Convert Currency Values**  
Choose this option to convert currency fields to the correct values.
  - **Use Long Table/Business View Names**  
Choose this option to view long table/view names.
  - **Use Long Column Names**  
Choose this option to view long column names.
6. On the Connect form, turn on one or more of the following Table/Business View Display Options:
  - **Tables Only**  
Choose this option to view only EnterpriseOne tables.
  - **Business Views Only**  
Choose this option to view only EnterpriseOne business views.

- Tables and Business Views

Choose this option to view both EnterpriseOne tables and EnterpriseOne business views.

7. To customize the list of functions that are enabled in ODA, click Advanced.

The Advanced configuration is optional. If you choose not to customize the list of functions enabled in ODA, a default list of settings is used.

---

## Deleting a Data Source

You can delete a data source.

### ► To delete a data source

---

1. Double-click the Control Panel icon. On Control Panel, double-click the ODBC icon.
2. On User Data Sources, File Data Sources, or System Data Sources, choose the data source you want to delete from the Data Sources list.
3. Click Remove, and then click Yes to confirm the deletion.

---

## Using Keywords in the Connection String

You can use C programming language to write database applications that directly invoke SQL APIs supported by ODA, such as SQLDriverConnect and SQLBrowseConnect. The following table lists keywords that you use in the connection string when you write your own database applications:

| Key                | Value                 | Description                 | Input Connection String                                                                                            | Output Connection String                       |
|--------------------|-----------------------|-----------------------------|--------------------------------------------------------------------------------------------------------------------|------------------------------------------------|
| CONVERTUDC         | YIN<br>(default to N) | Convert UDC or not          | Optional. If not in the connection string, load from INI/registry settings<br><br>(EnterpriseOne ODA DSN settings) | From the input string or INI/registry settings |
| CONVERTCURRENCY    | YIN<br>(default to N) | Convert currency or not     |                                                                                                                    |                                                |
| SHIFTDECIMALS      | YIN<br>(default to Y) | Use decimal shift or not    |                                                                                                                    |                                                |
| CONVERTJULIANDATES | YIN<br>(default to Y) | Convert Julian dates or not |                                                                                                                    |                                                |
| DISPLAYOPTIONS     | 0 1 2 (no default)    | Display TBLE, BSFN or both  |                                                                                                                    |                                                |

|                |                       |                                   |                                                    |                                                      |
|----------------|-----------------------|-----------------------------------|----------------------------------------------------|------------------------------------------------------|
| LONGTABLENAMES | YIN<br>(default to Y) | Use long names for tables or not  |                                                    |                                                      |
| LONGCOLUMNAMES | YIN<br>(default to Y) | Use long names for columns or not |                                                    |                                                      |
| UID            | <string>              | User ID                           | Required by JDEDriverConnect (SQL_DRIVER_NOPROMPT) | The same as the input if not overwritten by OW login |
| PWD            | <string>              | Password                          |                                                    |                                                      |
| ENVIRONMENT    | <string>              | Environment                       |                                                    |                                                      |
| DBQ            | <string>              | The same as the ENVIRONMENT       | Work as ENVIRONMENT, if ENVIRONMENT not specified  | Removed if ENVIRONMENT exist                         |
| DSN            | <string>              | Data source                       | Optional. Default to "DEFAULT" if invalid          | Overwritten by login                                 |

If you use the Microsoft Analysis Service tool, you can use connection string keywords to create a new data source. The following example shows how to write a connection string:

DSN=OneWorld ODA;DBQ=ADEVHP02;

---

## Working with ODA

Once the ODA driver is properly installed and an ODBC data source is established, you can use the functionality of the ODA driver. When a SQL connection is established, the environment of the current connection is stored in the system as the database name. SQLGetInfo can access this value later or it can be used for future connections.

You can use the following specific PeopleSoft EnterpriseOne features with EnterpriseOne ODA:

### Long Table and Business View Names

Long table and business view names allow you to see a descriptive name when you view an object list. You can use either the descriptive names or the original EnterpriseOne object name in the SELECT statement.

---

#### Note

This option might not be available for all third-party products (for example, ShowCase STRATEGY products prior to the 2.0 release or Crystal Reports) because the long names contain special characters that are not handled correctly by these tools.

---

## Long Column Names

Long column names allow you to see a descriptive name when viewing any columns list. You can still use either the descriptive names or the original EnterpriseOne column name. For example, you can use either of the following statements to retrieve information from the Address Book Master table (F0101):

- SELECT ABAN8 from the Address Book Master table (F0101)
- SELECT AddressNumber from the Address Book Master table (F0101)

## Julian Date

Julian date modifies all references to Julian date columns to convert the date to an SQL-92 standard date. The EnterpriseOne Julian date is converted to a standard date value that can be used in date calculations. This feature allows you to use duration or other date calculations in both the SELECT (result data), WHERE, and HAVING clauses and the ORDER BY clause.

The SQL SELECT statement is modified to before a data calculation to convert the EnterpriseOne Julian date column to a standard date. The modification to the SQL SELECT statement is based on the data source that is being accessed because of driver differences in handling date calculations. If the original column value is zero, the date conversion will result in a date value of 1899-12-31. To remove these values, the following condition should be added to the WHERE clause in the SELECT statement where DATECOL is the EnterpriseOne Julian date column:

```
"DATECOL <> {d `1899-12-31'}"
```

## Decimal Shifting

All references to decimal shifted columns are modified to shift the decimal point to cause the result data to be correct. This feature allows SQL statements containing complex expressions, aggregates, and filtering to run and return accurate results.

The SQL SELECT statement is modified to divide the column by the appropriate number of decimal places so that the data is returned correctly and to make “compare operators” work for filtering.

## Currency

Currency columns are limited to single-column references in the selected columns list. Returned data is converted using the standard EnterpriseOne currency conversion routines. All other references to the currency column in the SQL statement are passed through to the native driver. You must understand how the currency column is used to make effective use of filtering.

Before selected columns are returned, the EnterpriseOne Open Data Access driver converts any currency columns to the correct value. Currency columns used in the WHERE or HAVING clause are processed based on the nonconverted currency value. Currency columns in the GROUP BY or ORDER BY clause are grouped and sorted by the nonconverted currency value.

## Media Object

The Media object column, TXVC, in the Media Objects (F00165) storage is limited to single-column references in the selected columns list. ODA returns media data in plain text or rich text format (RTF) and truncates other binary data, such as an image. The size limitation of the text or RTF is 30,000 characters, and text will be truncated when it reaches this limitation.

## Column Security

When column security is active, any reference to restricted columns causes an error to be returned when the SELECT statement is examined, including the use of \* (asterisk—selecting all columns) in the select clause, as defined by the SQL-92 standards. You will receive an error if you are not authorized for all of the columns in the table.

## Row Security

When row security is active, the statement is modified to include the appropriate WHERE clause for filtering secured rows. You will only see rows that you are authorized to access along with getting accurate results using aggregate functions—for example, SUM or AVG.

## User Defined Codes

When user defined codes (UDCs) are enabled, you see the associated description instead of the internal code when the column data is returned. This processing affects only the returned data and has no effect on the other parts of the Select statement—for example, Where, Order By. This is an optional setting that can be configured when you set up the driver.

Before the UDC is returned to you, the EnterpriseOne Open Data Access driver converts the code to the associated description. The UDC columns used in the WHERE or HAVING clause are selected based on the nonconverted code and the UDC columns referenced in the GROUP BY and ORDER BY clause are grouped and sorted by the nonconverted code.

---

# Running a Query Using Microsoft Excel

You can use Microsoft Excel to create and run a query.

### ► To run a query using Microsoft Excel

---

1. From the Data menu, choose Get External Data.
2. Choose Create New Query.
3. On the Databases tab, choose the appropriate data source (for example, EnterpriseOne Local or EnterpriseOne ODA).

Because Excel uses file data sources, the ODA data source you set up in the 32-bit ODBC Administrator will not appear on the list of databases. You should create a File-type Data Source by choosing <New Data Source> and following the procedures for setting up a data source.

When you choose the ODA data source, you might need to log on to EnterpriseOne to use the ODA driver. Once you log on, you will not see the Solution Explorer because it is only activated so that the ODA driver can check security and environment mappings.

The Excel Query Wizard will display a list of available tables in the EnterpriseOne data source. Expanding any table name shows the available columns or fields in each table. If you are using the ODA driver, you will see long descriptions of each field (for example, DateUpdated). If not, you will see the alpha codes for the fields (for example ABUPMJ).

4. To translate field/column names from the EnterpriseOne alpha codes, use the Data Field Display Text (F9202) table. Choose all rows and sort (on FRDTAI) to create a cross-reference.

The first two letters of all EnterpriseOne column names are the application code, and the remaining letters are in this table as a suffix.

5. Finish building your query with Query Wizard and save the query.
6. You can then run your query and review it in Excel or MS Query.

After you run a query from Excel, if you ask to see the results in MS Query, you get the results back quickly because it selects a page at a time. If you are working with a large result set, you should close EnterpriseOne and any applications that take up a lot of memory so that you can navigate through the records faster. If you ask for the query to be returned directly to a spreadsheet instead of into MS Query, it might be a long time before you see any rows because Excel does not show a page at a time.

To verify the outcome of each query, you should run each one first using the non-ODA EnterpriseOne data source and then use the ODA data source and compare the results.

---

## ODA Error Messages

Following is a list of the errors that you can receive from the PeopleSoft EnterpriseOne Open Data Access driver. The messages are placed in the ODBC error message queue where the application can retrieve them using the standard ODBC error mechanism. The EnterpriseOne messages look like the following:

[J.D. Edwards][OneWorldODA Driver]MESSAGE TEXT

### Configuration Request Error

This error might occur when you add a new data source if you do not provide enough information for the driver and it cannot show a configuration dialog.

You must either pass enough information to the driver or allow the driver to prompt for more information.

### Option Value Changed

This is an informational message that occurs when you attempt to set a connection or statement option to a value that the driver does not accept. The driver then changes the value to an acceptable default and uses this message to let you know that the value has changed.

The EnterpriseOne Open Data Access driver changes values in the following areas.

- Setting the rowset size to a value other than one. The driver currently only supports single row rowsets.
- Setting the login time out to a value other than zero. The driver currently only supports zero in this option, which means, timeout disabled.

### **Data Source Name Is Not Valid**

The data source you entered is not a valid ODBC data source name. This error occurs when you are adding a new data source or configuring an existing data source. You must enter a name that follows the ODBC data source naming convention.

### **Data Source Does Not Exist**

This error occurs when you attempt to use a data source that does not exist. You must enter the name of an existing data source. If you get this error when you attempt to connect to a data source, you might need to create a default data source.

### **Unable to Allocate Memory**

The PeopleSoft EnterpriseOne Open Data Access driver was not able to allocate enough memory to continue. You must close some applications and try the operation again. Make sure that you meet the minimum system requirements.

### **Invalid Type of Request**

You attempted to use a configuration option that is unknown to the driver. The driver supports the following options when configuring data sources.

- Adding a data source
- Configuring a data source
- Removing a data source

### **Data Truncated**

The conversion of column data resulted in a truncation of the value. You should allocate more room for the column data to avoid this informational message.

### **Syntax Error or Access Violation**

The statement contained a syntax error and no further information is available.

### **Unable to Display Connection Dialog**

The driver encountered an error when attempting to display the connection dialog.

### **Cross System Joins Not Supported**

This error occurs in one of two situations:

- You referenced tables that are contained on multiple systems in the PeopleSoft EnterpriseOne environment. The EnterpriseOne Open Data Access driver currently supports tables referenced on a single system.
- You referenced a business view that contains multiple tables that reside on multiple systems.

You must make sure that you are referencing tables on a single system or a business view that contains tables on a single system.

### **Unable to Connect to the PeopleSoft EnterpriseOne Environment**

The driver could not establish a connection to the PeopleSoft EnterpriseOne environment. This connection is required before a successful connection can be made to this driver.

**Internal Data Conversion Error**

The driver encountered an unknown error during data conversion.

**Internal Execution Error**

The driver experienced an unexpected error during a statement execution.

**User Defined Code Columns Can Only Be in Simple Column References**

A user attempted to use a User Defined Code column in a complex expression. The PeopleSoft Enterprise Open Data Access driver only allows such columns to be simple references.

**Currency Columns Can Only Be in Simple Column References**

A user attempted to use a Currency column in a complex expression. The PeopleSoft EnterpriseOne Open Data Access driver only allows such columns to be simple references.

**Media Object Columns Can Only Be in Simple Column References**

A user attempted to use a Media Object column in a complex expression. The PeopleSoft EnterpriseOne Open Data Access driver only allows such columns to be simple references.

**Column Security Violation**

You attempted to use a column you are not authorized to use. You must remove references to those columns that are secured.

**Invalid Cursor State**

You attempted an operation that was not valid for the state that the driver is in, for example:

- You attempted to bind a column prior to preparing or executing a statement.
- You attempted to execute a statement while there are pending results.
- You attempted to get data from the driver prior to preparing or executing a statement.
- You attempted to prepare a statement while there are pending results.

**Invalid Column Number**

You attempted to access a column that was not part of the statements results.

**Driver Does Not Support the Requested Conversion**

An attempt was made to convert a column to a data type not supported by the PeopleSoft EnterpriseOne Open Data Access driver.

**Invalid Date/Time String**

An attempt to convert a character column to a date, time, or timestamp value failed because the character column did not contain a valid format.

**Invalid Numeric String**

An attempt to convert a character column to a numeric value failed because the character column did not contain a valid numeric value.

**Numeric Value Out of Range**

An attempt to convert a column to a numeric value failed because the output data type could not accommodate the value in the column. You should use the default data type or choose a data type that can accommodate the column value.

**Data Returned for One or More Columns was Truncated**

An attempt to convert a column to a numeric value caused a truncation of decimal digits. The output data type could not accommodate the value in the column. You should use the default data type or choose a data type that can accommodate the column value.

**The Data Cannot be Converted**

An attempt to convert a column value failed because the input type could not be converted to output type. You should use the default data type.

**Statement Must Be a SELECT**

The PeopleSoft EnterpriseOne Open Data Access driver is read-only and allows only SELECT statements.

**Attempt to Fetch Before the First Row**

An attempt was made to fetch before the beginning of results. The attempt resulted in the first rowset being fetched.

**Option Value Changed**

An attempt was made to set a connection, statement, or scroll options to a value that was not allowed. The PeopleSoft EnterpriseOne Open Data Access driver substituted a similar value.

**Fractional Truncation**

An attempt to convert a column to a numeric value succeeded with a loss of fractional digits because the output data type could not accommodate the value in the column. You should use the default data type or choose a data type that can accommodate the column value.

**Driver Not Capable**

An attempt was made to set a connection, statement, or scroll option that the driver does not allow.

**Multiple Business Views Referenced**

An attempt was made to reference more than one business view in a single SELECT statement. The PeopleSoft EnterpriseOne Open Data Access driver restricts the SELECT statement to contain only one business view.

**Unable to Open Table or Business View**

The PeopleSoft EnterpriseOne Open Data Access driver was unable to locate the table or business view in the EnterpriseOne database or could not get information pertaining to the table or business view.

**Server Connection Failed**

The PeopleSoft EnterpriseOne Open Data Access driver was unable to establish a connection to the server referenced by the tables or business view in the SELECT statement.

**Business View Contains Invalid Join**

The Business View definition contains a join condition that could not be processed by the PeopleSoft EnterpriseOne Open Data Access driver.

**Business View Contains Unsupported UNION Operator**

The Business View definition contains the UNION operator, which could not be processed by the PeopleSoft EnterpriseOne Open Data Access driver.

# XML Format Examples (All Parameters)

The following examples are used for specific formats.

---

## Example: Inbound Sales Order XML Format (All Parameters)

This example illustrates an inbound XML format with all of the parameters.

```
"<?xml version='1.0' ?>
<jdeRequest type='callmethod' user='userid' pwd='password' environment='environment'
>
<callMethod name=' GetLocalComputerId' app='NetCommerce' runOnError='no'>
<params>
<param name=' szMachineKey'id='m2' >< /param>
<params>
<callMethod>
<callMethod name=' F4211FSBeginDoc' app='NetCommerce' runOnError='no'>
<params>
<param name='mnCMJobNumber' id='j1'></param>
<param name='cMDocAction' >A</param>
<param name='cCMProcessEdits' <1>/param>
<param name='szCMComputerID' idref='c2' >< /param>
<param name='cCMErrorConditions' >value< /param>
<param name='cCMUpdateWriteToWF' >value< /param>
<param name='szCMProgramID' >value< /param>
<param name='szCMVersion' >value< /param>
<param name='szOrderCo' <value< /param>
<param name='mnOrderNo' >value< /param>
<param name='szOrderType' >value< /param>
<param name='szBusinessUnit' >value< /param>
<param name='szOriginalOrderCo' >value< /param>
<param name='szOriginalOrderNo' >value< /param>
<param name='szOriginalOrderType' >value< /param>
```

```
<param name='mnAddressNumber' >value< /param>
<param name='mnShipToNo' >value< /param>
<param name='jdRequestedDate' >value< /param>
<param name='jdOrderDate' >value< /param>
 <param name='jdPromisedDate' >value< /param>
<param name='jdCancelDate' >value< /param>
<param name='szReference' >value< /param>
<param name='szDeliveryInstructions1' >value< /param>
<param name='szDeliveryInstructions2' >value< /param>
<param name='szPrintMesg' >value< /param>
 <param name='szPaymentTerm' >value< /param>
<param name='cPaymentInstrument' >value< /param>
 <param name='szAdjustmentSchedule' >value< /param>
 <param name='mnTradeDiscount' >value< /param>
 <param name='szTaxExplanationCode' >value< /param>
 <param name='szTaxArea' >value< /param>
 <param name='szCertificate' >value< /param>
 <param name='cAssociatedText' >value< /param>
 <param name='szHoldOrdersCode' >value< /param>
<param name='cPricePickListYN' >value< /param>
<param name='mnInvoiceCopies' >value< /param>
<param name='mnBuyerNumber' >value< /param>
<param name='mnCarrier' >value< /param>
<param name='szRouteCode' >value< /param>
<param name='szStopCode' >value< /param>
<param name='szZoneNumber' >value< /param>
<param name='szFreightHandlingCode' >value< /param>
<param name='cApplyFreightYN' >value< /param>
<param name='mnCommissionCode1' >value< /param>
<param name='mnCommissionRate1' >value< /param>
<param name='mnCommissionCode2' >value< /param>
```

```

<param name='mnCommissionRate2'>value</param>
<param name='szWeightDisplayUOM'>value</param>
<param name='szVolumeDisplayUOM'>value</param>
<param name='szAuthorizationNo'>value</param>
<param name='szCreditBankAcctNo'>value</param>
<param name='jdCreditBankExpiredDate'>value</param>
<param name='cMode'>value</param>
<param name='szCurrencyCode'>value</param>
<param name='mnExchangeRate'>value</param>
<param name='szOrderedBy'>value</param>
<param name='szOrderTakenBy'>value</param>
<param name='szUserReservedCode'>value</param>
<param name='jdUserReservedDate'>value</param>
<param name='mnUserReservedAmnt'>value</param>
<param name='mnUserReservedNo'>value</param>
<param name='szUserReservedRef'>value</param>
<param name='jdDateUpdated'>value</param>
<param name='szUserID'>value</param>
<param name='szWKBaseCurrency'>value</param>
<param name='cWKAdvancedPricingYN'>value</param>
<param name='szWKCreditMesg'>value</param>
<param name='szWKTempCreditMesg'>value</param>
<param name='cWKInvalidSalesOrderNo'>value</param>
<param name='cWKSourceOfData'>blank</param>
<param name='cWKProcMode'>blank</param>
<param name='mnWKSuppressProcess'>0</param>
<param name='mnSODDocNo'>value</param>
<param name='szSODDocType'>value</param>
<param name='szSODOrderCo'>value</param>
<param name='mnTriangulationRateFrom'>value</param>
<param name='mnTriangulationRateTo'>value</param>

```

```

<param name='cCurrencyConversionMethod' >value</param>
<param name='cRetrieveOrderNo' >value</param>
<param name='szPricingGroup' >value</param>
<param name='cCommitInvInED' >value</param>
<param name='cSpotRateAllowed' >value</param>
<param name='cGenericChar2_EV02' >value</param>
<param name='szGenericString1_DL01' >value</param>
<param name='szGenericString2_DL02' >value</param>
<param name='mnGenericMathNumeric1_MATH01' >value</param>
<param name='mnGenericMathNumeric2_MATH02' >value</param>
<param name='szLongAddressNumberShipto' >value</param>
<param name='szLongAddressNumber' >value</param>
<param name='mnProcessID' >value</param>
<param name='mnTransactionID' >value</param>
</params>
<onError abort='yes'>\
<callMethod name='F4211ClearWorkFile'app='NetCommerce' runOnError='yes'>
<params>
<param name='mnJobNo' idref='j1'></param>
<param name='szComputerID' idref='c2'></param>
<param name='mnFromLineNo' >value</param>
<param name='mnThruLineNo' >value</param>
<param name='cClearHeaderWF' >value</param>
<param name='cClearDetailWF' >value</param>
<param name='szProgramID' >value</param>
<param name='mnWKRelatedOrderProcess' >value</param>
<param name='szCMVersion' >value</param>
<param name='cGenericChar1_EV01' >value</param>
<param name='szGenericString1_DL01' >value</param>
<param name='mnSODRelatedJobNumber' >value</param>
<param name='mnProcessID' >value</param>

```

```

<param name='mnTransactionID' >value</param>
</params>
</callMethod>
</onError>
</callMethod>
<callMethod name=' F4211FSEditLine'app='NetCommerce' runOnError='yes'>
<params>
<param name='mnCMJobNo ` idref='j1'></param>
<param name='cCMLineAction'>value</param>
<param name='cCMProcessEdits ' >value</param>
<param name='cCMWriteToWFFlag ' >value</param>
<param name='cCMReedWrittenToWF ' >value</param>
<param name='szCMComputerID' idref='c2'></param>
<param name='cCMErrConditions' >value</param>
<param name='szOrderCo' >value</param>
<param name='mnOrderNo' >value</param>
<param name='szOrderType' >value</param>
<param name='mnLineNo' >value</param>
<param name='szBusinessUnit' >value</param>
<param name='mnShipToNo' >value</param>
<param name='jdRequestedDate' >value</param>
<param name='jdPromisedDate' >value</param>
<param name='jdCancelDate' >value</param>
<param name='jdPromisedDlvryDate' >value</param>
<param name='szItemNo' >value</param>
<param name='szLocation'>value</param>
<param name='szLotNo' >value</param>
<param name='szDescription1' >value</param>
<param name='szDescription2' >value</param>
<param name='szLineType' >value</param>
<param name='szLastStatus' >value</param>

```

```

<param name='szNextStatus' >value</param>
<param name='mnQtyOrdered' >value</param>
<param name='mnQtyShipped' >value</param>
<param name='mnQtyBackordered' >value</param>
<param name='mnQtyCanceled' >value</param>
<param name='mnExtendedPrice' >value</param>
<param name='mnExtendedCost' >value</param>
<param name='szPrintMesg' >value</param>
<param name='cPaymentInstrument' >value</param>
<param name='szAdjustmentSchedule' >value</param>
<param name='cSalesTaxableYN' >value</param>
<param name='cAssociatedText' >value</param>
<param name='szTransactionUOM' >value</param>
<param name='szPricingUOM' >value</param>
<param name='mnItemWeight' >value</param>
<param name='szWeightUOM' >value</param>
<param name='mnForeignUnitPrice' >value</param>
<param name='mnForeignExtPrice' >value</param>
<param name='mnForeignUnitCost' >value</param>
<param name='mnForeignExtCost' >value</param>
<param name='szPricingCategoryLevel' >value</param>
<param name='mnDiscountFactor' >value</param>
<param name='mnCMLineNo' >value</param>
<param name='szCMProgramID' >value</param>
<param name='szCMVersion' >value</param>
<param name='mnSupplierNo' >value</param>
<param name='szRelatedKitItemNo' >value</param>
<param name='mnKitMasterLineNo' >value</param>
<param name='mnComponentLineNo' >value</param>
<param name='mnRelatedKitComponent' >value</param>
<param name='mnNoOfCpntPerParent' >value</param>

```

```

<param name='cOverridePrice' >value</param>
<param name='cOverrideCost' >value</param>
<param name='szUserID' >value</param>
<param name='jdDateUpdated' >value</param>
<param name='mnWKOrderTotal' >value</param>
<param name='mnWKForeignOrderTotal' >value</param>
<param name='mnWKTotCost' >value</param>
<param name='mnWKForeignTotCost' >value</param>
<param name='cWKProcessingType' >value</param>
<param name='cWKSourceOfData' >value</param>
<param name='cWKCheckAvailability' >value</param>
<param name='mnLastLineNoAssigned' >value</param>
<param name='cStockingType' >value</param>
<param name='szOriginalOrderKeyCo' >value</param>
<param name='szOriginalOrderNo' >value</param>
<param name='szOriginalOrderType' >value</param>
<param name='mnOriginalOrderLineNo' >value</param>
<param name='cParentItemMethodOfPriceCalc' >value</param>
<param name='szLandedCost' >value</param>
<param name='mnWKSuppressProcess' >value</param>
<param name='mnShortItemNo' >value</param>
<param name='mnWKRelatedOrderProcess' >value</param>
<param name='mnSODLineNo' >value</param>
<param name='mnPriceAdjRevLevel' >value</param>
<param name='szSalesOrderFlags' >value</param>
<param name='mnSODDocNo' >value</param>
<param name='szSODDocType' >value</param>
<param name='szSODOrderCo' >value</param>
<param name='szTransferOrderToBranch' >value</param>
<param name='mnDomesticDetachedAdj' >value</param>
<param name='mnForeignDetachedAdj' >value</param>

```

```
<param name='mnSODWFLineNo' >value</param>
<param name='szGeneric2CharString' >value</param>
<param name='mnTOEPOExchangeRate' >value</param>
<param name='szTOEPOCurrencyCode' >value</param>
<param name='mnDRPKeyId' >value</param>
<param name='mnSoldToCust' >value</param>
<param name='szF4201BranchPlant' >value</param>
<param name='szSoldToCurrencyCode' >value</param>
<param name='cConsolidationFlag' >value</param>
<param name='jdPriceEffectiveDate' >value</param>
<param name='mnWOWFLineNo' >value</param>
<param name='mnLineNoIncrement' >value</param>
<param name='mnParentWFLineNo' >value</param>
<param name='cStatusInWarehouse' >value</param>
<param name='cBypassCommitments' >value</param>
<param name='szProductSource' >value</param>
<param name='szProductSourceType' >value</param>
<param name='mnSequenceNumber' >value</param>
<param name='szAgreementNumber' >value</param>
<param name='mnAgreementSupplement' >value</param>
<param name='mnAgreementsFound' >value</param>
<param name='szModeOfTransport' >value</param>
<param name='szDutyStatus' >value</param>
<param name='szLineofBusiness' >value</param>
<param name='jdPromisedShip' >value</param>
<param name='szEndUse' >value</param>
<param name='mnTOEPOExchangeRate' >value</param>
<param name='szPriceCode1' >value</param>
<param name='szPriceCode2' >value</param>
<param name='szPriceCode3' >value</param>
<param name='szItemFlashMessage' >value</param>
```

```

<param name='szCompanyKeyRelated' >value</param>
<param name='szRelatedPoSoNumber' >value</param>
<param name='szRelatedOrderType' >value</param>
<param name='mnRelatedPoSoLineNo' >value</param>
<param name='cGenericChar3' >value</param>
<param name='mnProfitMargin' >value</param>
<param name='mnQuantityAvailable' >value</param>
<param name='cRequestScheduleFlag' >value</param>
<param name='cOrderProcessType' >value</param>
<param name='cGenericChar2' >value</param>
<param name='mnSODRelatedJobNumber' >value</param>
<param name='szGenericString' >value</param>
<param name='mnCarrier' >value</param>
<param name='szGenericString2_DL02' >value</param>
<param name='mnGenericMathNumeric1_MATH01' >value</param>
<param name='mnGenericMathNumeric2_MATH02' >value</param>
<param name='mnItemVolume_ITVL' >value</param>
<param name='szVolumeUOM_VLUM' >value</param>
<param name='szRevenueBusinessUnit' >value</param>
<param name='szCustomerPO_VR01' >value</param>
<param name='szReference2Vendor_VR02' >value</param>
<param name='mnProcessID' >value</param>
<param name='mnTransactionID' >value</param>
</params>
<onError abort='no'>\
</onError>
</callMethod>
<callMethod name='F4211FSEndDoc'app='NetCommerce' runOnError='no'>
<params>
<param name='mnCMJobNo' idref='j1'></param>
<param name='mnSalesOrderNo' >value</param>

```

```

<param name='szCMComputerID' idref='c2' >< /param>
<param name='cCMErrCondition' >value</param>
<param name='szOrderType' >value</param>
<param name='szKeyCompany' >value</param>
<param name='mnOrderTotal' >value</param>
<param name='mnForeignOrderTotal' >value</param>
<param name='szBaseCurrencyCode' >value</param>
<param name='szProgramID' >value</param>
<param name='szWorkstationID' >value</param>
<param name='szCMProgramID' >value</param>
<param name='szCMVersion' >value</param>
<param name='mnTimeOfDay' >value</param>
<param name='mnTotalCost' >value</param>
<param name='mnForeignTotalCost' >value</param>
<param name='cSuppressRlvBlnktFlag' >value</param>
<param name='cWKSkipProcOptions' >value</param>
<param name='mnWKRelatedOrderProcess' >value</param>
<param name='cCMUseWorkFiles' >value</param>
<param name='mnEDIDocNo' >value</param>
<param name='szEDIKeyCo' >value</param>
<param name='szEDIDocType' >value</param>
<param name='cCMPProcessEdits' >value</param>
<param name='cGenericChar2' >value</param>
<param name='mnSODRelatedJobNumber' >value</param>
<param name='cGenericChar1_EV01' >value</param>
<param name='mnGenericMathNumeric2_MATH02' >value</param>
<param name='szGenericString1_DL01' >value</param>
<param name='szGenericString2_DL02' >value</param>
<param name='mnProcessID' >value</param>
<param name='mnTransactionID' >value</param>
<params/>

```

```

<onError abort='no'>\
<callMethod name=' F4211ClearWorkFile'app='NetCommerce'
runOnError='yes'>
<params>
 <param name='mnJobNo` idref='j1`></param>
 <param name='szComputerID' idref='c2`></param>
 <param name='mnFromLineNo' >value</param>
 <param name='mnThruLineNo' >value</param>
 <param name='cClearHeaderWF' >value</param>
 <param name='cClearDetailWF' >value</param>
 <param name='szProgramID' >value</param>
 <param name='mnWKRelatedOrderProcess' >value</param>
 <param name='szCMVersion' >value</param>
 <param name='cGenericChar1_EV01' >value</param>
 <param name='szGenericString1_DL01' >value</param>
 <param name='mnSODRelatedJobNumber' >value</param>
 <param name='mnProcessID' >value</param>
 <param name='mnTransactionID' >value</param>
</params>
</callMethod>
</onError>
</callMethod>
<returnParams version='value' message='failure Destination='queueName'
successDestination='queueName'>
 <param name='long description' idref='value'/param>
</returnParams>
</onError>
<callMethod name=' F4211ClearWorkFile'app='NetCommerce'
runOnError='yes'>
<params>
 <param name='mnJobNo` idref='j1`></param>

```

```

<param name='szComputerID' idref='c2'></param>
<param name='mnFromLineNo'>value</param>
<param name='mnThruLineNo'>value</param>
<param name='cClearHeaderWF'>value</param>
<param name='cClearDetailWF'>value</param>
<param name='szProgramID'>value</param>
<param name='mnWKRelatedOrderProcess'>value</param>
<param name='szCMVersion'>value</param>
<param name='cGenericChar1_EV01'>value</param>
<param name='szGenericString1_DL01'>value</param>
 <param name='mnSODRelatedJobNumber'>value</param>
<param name='mnProcessID'>value</param>
<param name='mnTransactionID'>value</param>
</params>
</callMethod>
</onError>
</jdeRequest>

```

## Example: Outbound Customer Create XML Request and Response Format (All Parameters)

The following examples, both as request and response, illustrate the outbound XML format with all of the parameters.

```

□ This format will return all columns for the customer master (F0101Z2) □
<?xml version='1.0' ?>
<jdeRequest type='trans' user='user' pwd='password' environment='environment' session='' sessionid='300'>
 <transaction action='transactionInfo' type='JDEAB'>
 <key>
 <column name='EdiUserId'>value</column>
 <column name='EdiBatchNumber'>value</column>
 <column name='EdiTransactNumber'>value</column>
 </key>
 </transaction>
</jdeRequest>

```

```

<?xml version='1.0' encoding='utf-8' ?>
<jdeResponse type='trans' user='user' session='session' environment='env'>
 <transaction type='JDEAB' action='transactionInfo'>
 <returnCode code='0'>XML Request OK</returnCode>
 <key>
 <column name='EdiUserId'></column>
 <column name='EdiBatchNumber'></column>
 </key>
 <table name='F0101Z2' type='detail'>
 <column name='EdiUserId'></column>
 <column name='EdiBatchNumber'></column>
 <column name='EdiTransactNumber'></column>
 <column name='EdiLineNumber'></column>
 <column name='EdiDocumentType'></column>
 <column name='TypeTransaction'></column>
 <column name='EdiTranslationFormat'></column>
 <column name='EdiTransmissionDate'></column>
 <column name='DirectionIndicator'></column>
 <column name='EdiDetailLinesProcess'></column>
 <column name='EdiSuccessfullyProcess'></column>
 <column name='TradingPartnerId'></column>
 <column name='TransactionAction'></column>
 <column name='AddressNumber'></column>
 <column name='AlternateAddressKey'></column>
 <column name='TaxId'></column>
 <column name='NameAlpha'></column>
 <column name='DescripCompressed'></column>
 <column name='CostCenter'></column>
 <column name='StandardIndustryCode'></column>
 <column name='LanguagePreference'></column>
 <column name='AddressType1'></column>
 <column name='CreditMessage'></column>
 <column name='PersonCorporationCode'></column>
 <column name='AddressType2'></column>
 <column name='AddressType3'></column>
 <column name='AddressType4'></column>
 <column name='AddressTypeReceivables'></column>
 <column name='AddressType5'></column>
 <column name='AddressTypePayables'></column>
 <column name='AddTypeCode4Purch'></column>
 </table>
 </transaction>
</jdeResponse>

```

```

<column name='MiscCode3'></column>
<column name='AddressTypeEmployee'></column>
<column name='SubledgerInactiveCode'></column>
<column name='DateBeginningEffective'></column>
<column name='AddressNumber1st'></column>
<column name='AddressNumber2nd'></column>
<column name='AddressNumber3rd'></column>
<column name='AddressNumber4th'></column>
<column name='AddressNumber6th'></column>
<column name='AddressNumber5th'></column>
<column name='ReportCodeAddBook001'></column>
<column name='ReportCodeAddBook002'></column>
<column name='ReportCodeAddBook003'></column>
<column name='ReportCodeAddBook004'></column>
<column name='ReportCodeAddBook005'></column>
<column name='ReportCodeAddBook006'></column>
<column name='ReportCodeAddBook007'></column>
<column name='ReportCodeAddBook008'></column>
<column name='ReportCodeAddBook009'></column>
<column name='ReportCodeAddBook010'></column>
<column name='ReportCodeAddBook011'></column>
<column name='ReportCodeAddBook012'></column>
<column name='ReportCodeAddBook013'></column>
<column name='ReportCodeAddBook014'></column>
<column name='ReportCodeAddBook015'></column>
<column name='ReportCodeAddBook016'></column>
<column name='ReportCodeAddBook017'></column>
<column name='ReportCodeAddBook018'></column>
<column name='ReportCodeAddBook019'></column>
<column name='ReportCodeAddBook020'></column>
<column name='CategoryCodeAddressBook2'></column>
<column name='CategoryCodeAddressBk22'></column>
<column name='CategoryCodeAddressBk23'></column>
<column name='CategoryCodeAddressBk24'></column>
<column name='CategoryCodeAddressBk25'></column>
<column name='CategoryCodeAddressBk26'></column>
<column name='CategoryCodeAddressBk27'></column>
<column name='CategoryCodeAddressBk28'></column>
<column name='CategoryCodeAddressBk29'></column>
<column name='CategoryCodeAddressBk30'></column>

```

```

<column name='GIBankAccount'></column>
<column name='TimeScheduledIn'></column>
<column name='DateScheduledIn'></column>
<column name='ActionMessageControl'></column>
<column name='NameRemark'></column>
<column name='CertificateTaxExempt'></column>
<column name='TaxId2'></column>
<column name='Kanjialpha'></column>
<column name='UserReservedCode'></column>
<column name='UserReservedDate'></column>
<column name='UserReservedAmount'></column>
<column name='UserReservedNumber'></column>
<column name='UserReservedReference'></column>
<column name='NameMailing'></column>
<column name='SecondaryMailingName'></column>
<column name='AddressLine1'></column>
<column name='AddressLine2'></column>
<column name='AddressLine3'></column>
<column name='AddressLine4'></column>
<column name='ZipCodePostal'></column>
<column name='City'></column>
<column name='Country'></column>
<column name='State'></column>
<column name='County Address'></column>
<column name='PhoneAreaCode1'></column>
<column name='PhoneNumber'></column>
<column name='PhoneNumberTyp1'></column>
<column name='PhoneAreaCode2'></column>
<column name='PhoneNumber1'></column>
<column name='PhoneNumberTyp2'></column>
<column name='TransactionOriginator'></column>
<column name='UserId'></column>
<column name='ProgramId'></column>
<column name='WorkStationId'></column>
<column name='DateUpdated'></column>
<column name='TimeOfDay'></column>
<column name='TimeLastUpdated'></column>
</table>
</transaction>
</jdeResponse>

```

# XML Format Examples (Default Values)

The following example shows the EnterpriseOne default values.

---

## Example: Inbound Sales Order XML Format

The following sales order entry example shows the minimum required parameters. PeopleSoft EnterpriseOne recommends that you start with the minimum required values and test them to ensure your system is working. After you are confident the minimum required values are working properly, you can add other values.

```
<?xml version="1.0" encoding="utf-8" ?>
..<jdeRequest type="callmethod" user="JDE" pwd="JDE" environment="PRD733">
.....<callMethod name="GetLocalComputerId" app="NetComm" runOnError="no">
 <params>
 <param name="szMachineKey" id="2" />
 </params>
 <onError abort="yes" />
.....</callMethod>
.....<callMethod name="F4211FSBeginDoc" app="NetComm" runOnError="no">
 <params>
 <param name="szCMComputerID" idref="2" />
 <param name="szOrderType">S4</param>
 <param name="szBusinessUnit">M30</param>
 <param name="mnAddressNumber">4242</param>
 </params>
 <onError abort="yes">
.....<callMethod name="F4211ClearWorkFile" app="NetComm" runOnError="yes">
 <params>
 <param name="mnJobNo" idref="1" />
 <param name="szComputerID" idref="2" />
 <param name="cClearHeaderWF">2</param>
 <param name="cClearDetailWF">2</param>
 </params>
.....</callMethod>
 </onError>
.....</callMethod>
.....<callMethod name="F4211FSEditLine" app="NetComm" runOnError="yes">
 <params>
 <param name="mnCMJobNo" idref="1" />
 <param name="szCMComputerID" idref="2" />
 <param name="szBusinessUnit">M30</param>
 <param name="szItemNo">1001</param>
 </params>
 <onError abort="no" />
.....</callMethod>
.....<callMethod name="F4211FSEndDoc" app="NetComm" runOnError="no">
 <params>
 <param name="mnCMJobNo" idref="1" />
 <param name="szCMComputerID" idref="2" />
```

```

 </params>
 <onError abort="no">
.....<callMethod name="F4211ClearWorkFile" app="NetComm" runOnError="yes">
 <params>
 <param name="mnJobNo" idref="1" />
 <param name="szComputerID" idref="2" />
 <param name="mnFromLineNo">0</param>
 <param name="mnThruLineNo">0</param>
 <param name="cClearHeaderWF">2</param>
 <param name="cClearDetailWF">2</param>
 <param name="szProgramID">NetComm</param>
 <param name="szCMVersion">ZJDE0001</param>
 </params>
.....</callMethod>
 </onError>
.....</callMethod>
.....<returnParams failureDestination="ERROR.Q" successDestination="SUCCESS.Q" runOnError="yes" />
 <onError abort="yes">
.....<callMethod name="F4211ClearWorkFile" app="NetComm" runOnError="yes">
 <params>
 <param name="mnJobNo" idref="1" />
 <param name="szComputerID" idref="2" />
 <param name="mnFromLineNo">0</param>
 <param name="mnThruLineNo">0</param>
 <param name="cClearHeaderWF">2</param>
 <param name="cClearDetailWF">2</param>
 <param name="szProgramID">NetComm</param>
 <param name="szCMVersion">ZJDE0001</param>
 </params>
.....</callMethod>
 </onError>
</jdeRequest>

```

# XML Format Examples (Events)

The following examples are XML samples for Z and real-time events.

---

## Example: Z Events XML Format

The following example illustrates a Z file event XML document.

```
<?xml version='1.0' encoding='utf-8' ?>
<jdeResponse type='trans' user='JDE' environment='XDEVNIS2'>
 <transaction type='JDESC' action='transactionInfo'>
 <returnCode code='0'>XML Request OK</returnCode>
 <key>
 <EdiUserId>KW6803955</EdiUserId>
 <EdiBatchNumber>16319</EdiBatchNumber>
 <EdiTransactNumber>106053</EdiTransactNumber>
 </key>
 <F4201Z1 type='header'>
 <EdiUserId>KW6803955</EdiUserId>
 <EdiBatchNumber>16319</EdiBatchNumber>
 <EdiTransactNumber>106053</EdiTransactNumber>
 <EdiLineNumber>1.000</EdiLineNumber>
 <EdiDocumentType>SO</EdiDocumentType>
 <TypeTransaction>JDESC</TypeTransaction>
 <EdiTranslationFormat> </EdiTranslationFormat>
 <EdiTransmissionDate></EdiTransmissionDate>
 <DirectionIndicator>2</DirectionIndicator>
 <EdiDetailLinesProcess>0</EdiDetailLinesProcess>
 <EdiSuccessfullyProcess>Y</EdiSuccessfullyProcess>
 <TradingPartnerId> </TradingPartnerId>
 <TransactionAction>UA</TransactionAction>
 <CompanyKeyOrderNo>00200</CompanyKeyOrderNo>
 <DocumentOrderInvoiceE>6559</DocumentOrderInvoiceE>
 <OrderType>SO</OrderType>
 <OrderSuffix>000</OrderSuffix>
 </F4201Z1>
 </transaction>
</jdeResponse>
```

```

<CostCenter> M30</CostCenter>
<Company>00200</Company>
<CompanyKeyOriginal> </CompanyKeyOriginal>
<OriginalPoSoNumber> </OriginalPoSoNumber>
<OriginalOrderType> </OriginalOrderType>
<CompanyKeyRelated> </CompanyKeyRelated>
<RelatedPoSoNumber> </RelatedPoSoNumber>
<RelatedOrderType> </RelatedOrderType>
<AddressNumber>4242</AddressNumber>
<AddressNumberShipTo>4242</AddressNumberShipTo>
<AddressNumberParent>4242</AddressNumberParent>
<DateRequestedJulian>2005/05/05</DateRequestedJulian>
<DateTransactionJulian>2005/05/05</DateTransactionJulian>
<PromisedDeliveryDate>2005/05/05</PromisedDeliveryDate>
<DateOriginalPromise>2005/05/05</DateOriginalPromise>
<ActualDeliveryDate></ActualDeliveryDate>
<CancelDate></CancelDate>
<DatePriceEffectiveDate>2005/05/05</DatePriceEffectiveDate>
<DatePromisedPickJu>2005/05/05</DatePromisedPickJu>
<DatePromisedShipJu></DatePromisedShipJu>
<Reference1> </Reference1>
<Reference2Vendor> </Reference2Vendor>
<DeliveryInstructLine1> </DeliveryInstructLine1>
<DeliveryInstructLine2> </DeliveryInstructLine2>
<PrintMessage1> </PrintMessage1>
<PaymentTermsCode01> </PaymentTermsCode01>
<PaymentInstrumentA> </PaymentInstrumentA>

```

```

<PriceAdjustmentScheduleN> </PriceAdjustmentScheduleN>
<PricingGroup>PREFER</PricingGroup>
<DiscountTrade>.000</DiscountTrade>
<PercentRetainage1>.000</PercentRetainage1>
<TaxArea1>DEN</TaxArea1>
<TaxExplanationCode1>S</TaxExplanationCode1>
<CertificateTaxExempt> </CertificateTaxExempt>
<AssociatedText> </AssociatedText>
<PriorityProcessing>0</PriorityProcessing>
<BackordersAllowedYN>Y</BackordersAllowedYN>
<SubstitutesAllowedYN>Y</SubstitutesAllowedYN>
<HoldOrdersCode> </HoldOrdersCode>
<PricePickListYN>Y</PricePickListYN>
<InvoiceCopies>0</InvoiceCopies>
<NatureOfTransaction> </NatureOfTransaction>
<BuyerNumber>0</BuyerNumber>
<Carrier>0</Carrier>
<ModeOfTransport> </ModeOfTransport>
<ConditionsOfTransport> </ConditionsOfTransport>
<RouteCode> </RouteCode>
<StopCode> </StopCode>
<ZoneNumber> </ZoneNumber>
<ContainerID> </ContainerID>
<FreightHandlingCode> </FreightHandlingCode>
<ApplyFreightYN>Y</ApplyFreightYN>
<ApplyFreight> </ApplyFreight>
<FreightCalculatedYN> </FreightCalculatedYN>

```

```

<MergeOrdersYN> </MergeOrdersYN>
<CommissionCode1>6001</CommissionCode1>
<RateCommission1>5.000</RateCommission1>
<CommissionCode2>0</CommissionCode2>
<RateCommission2>.000</RateCommission2>
<ReasonCode> </ReasonCode>
<PostQuantities> </PostQuantities>
<AmountOrderGross>134.97</AmountOrderGross>
<AmountTotalCost>.00</AmountTotalCost>
<UnitOfMeasureWhtDisp> </UnitOfMeasureWhtDisp>
<UnitOfMeasureVolDisp> </UnitOfMeasureVolDisp>
<AuthorizationNoCredit> </AuthorizationNoCredit>
<AcctNoCrBank> </AcctNoCrBank>
<DateExpired></DateExpired>
<SubledgerInactiveCode> </SubledgerInactiveCode>
<CorrespondenceMethod> </CorrespondenceMethod>
<CurrencyMode>F</CurrencyMode>
<CurrencyCodeFrom>BEF</CurrencyCodeFrom>
<CurrencyConverRateOv>33.8180588</CurrencyConverRateOv>
<LanguagePreference>E</LanguagePreference>
<AmountForeignOpen>4564.42</AmountForeignOpen>
<AmountForeignTotalC>.00</AmountForeignTotalC>
<OrderedBy> </OrderedBy>
<OrderTakenBy> </OrderTakenBy>
<UserReservedCode> </UserReservedCode>
<UserReservedDate></UserReservedDate>
<UserReservedAmount>.00</UserReservedAmount>

```

```

<UserReservedNumber>0</UserReservedNumber>
<UserReservedReference> </UserReservedReference>
<UserId>KW6803955</UserId>
<ProgramId> </ProgramId>
<WorkStationId>STI5</WorkStationId>
<DateUpdated>2000/08/22</DateUpdated>
<TimeOfDay>134435</TimeOfDay>
</F4201Z1>
<F4211Z1 type='detail'>
 <EdiUserId>KW6803955</EdiUserId>
 <EdiBatchNumber>16319</EdiBatchNumber>
 <EdiTransactNumber>106053</EdiTransactNumber>
 <EdiLineNumber>1.000</EdiLineNumber>
 <EdiDocumentType>SO</EdiDocumentType>
 <TypeTransaction>JDESC</TypeTransaction>
 <EdiTranslationFormat> </EdiTranslationFormat>
 <EdiTransmissionDate></EdiTransmissionDate>
 <DirectionIndicator>2</DirectionIndicator>
 <EdiDetailLinesProcess>0</EdiDetailLinesProcess>
 <EdiSuccessfullyProcess>N</EdiSuccessfullyProcess>
 <TradingPartnerId> </TradingPartnerId>
 <TransactionAction>UA</TransactionAction>
 <CompanyKeyOrderNo>00200</CompanyKeyOrderNo>
 <DocumentOrderInvoiceE>6559</DocumentOrderInvoiceE>
 <OrderType>SO</OrderType>
 <LineNumber>1.000</LineNumber>
 <OrderSuffix>000</OrderSuffix>

```

```

<CostCenter> M30</CostCenter>
<Company>00200</Company>
<CompanyKeyOriginal> </CompanyKeyOriginal>
<OriginalPoSoNumber> </OriginalPoSoNumber>
<OriginalOrderType> </OriginalOrderType>
<OriginalLineNumber>.000</OriginalLineNumber>
<CompanyKeyRelated> </CompanyKeyRelated>
<RelatedPoSoNumber> </RelatedPoSoNumber>
<RelatedOrderType> </RelatedOrderType>
<RelatedPoSoLineNo>.000</RelatedPoSoLineNo>
<ContractNumberDistributi> </ContractNumberDistributi>
<ContractSupplementDistri>0</ContractSupplementDistri>
<ContractBalancesUpdatedY> </ContractBalancesUpdatedY>
<AddressNumber>4242</AddressNumber>
<AddressNumberShipTo>4242</AddressNumberShipTo>
<AddressNumberParent>4242</AddressNumberParent>
<DateRequestedJulian>2005/05/05</DateRequestedJulian>
<DateTransactionJulian>2005/05/05</DateTransactionJulian>
<PromisedDeliveryDate>2005/05/05</PromisedDeliveryDate>
<DateOriginalPromisde>2005/05/05</DateOriginalPromisde>
<ActualDeliveryDate></ActualDeliveryDate>
<DateInvoiceJulian></DateInvoiceJulian>
<CancelDate></CancelDate>
<DtForGLAndVouch1></DtForGLAndVouch1>
<DateReleaseJulian>2005/05/05</DateReleaseJulian>
<DatePriceEffectiveDate>2005/05/05</DatePriceEffectiveDate>
<DatePromisedPickJu>2005/05/05</DatePromisedPickJu>

```

```

<DatePromisedShipJu></DatePromisedShipJu>
<Reference1> </Reference1>
<Reference2Vendor> </Reference2Vendor>
<IdentifierShortItem>60003</IdentifierShortItem>
<Identifier2ndItem>1001</Identifier2ndItem>
<Identifier3rdItem>1001</Identifier3rdItem>
<Location> </Location>
<Lot> </Lot>
<FromGrade> </FromGrade>
<ThruGrade> </ThruGrade>
<FromPotency>.000</FromPotency>
<ThruPotency>.000</ThruPotency>
<DaysPastExpiration>0</DaysPastExpiration>
<DescriptionLine1>Bike Rack - Trunk Mount</DescriptionLine1>
<DescriptionLine2> </DescriptionLine2>
<LineType>S</LineType>
<StatusCodeNext>540</StatusCodeNext>
<StatusCodeLast>520</StatusCodeLast>
<CostCenterHeader> M30</CostCenterHeader>
<ItemNumberRelatedKit> </ItemNumberRelatedKit>
<LineNumberKitMaster>.000</LineNumberKitMaster>
<ComponentNumber>.0</ComponentNumber>
<RelatedKitComponent>0</RelatedKitComponent>
<NumOfCpntPerParent>0</NumOfCpntPerParent>
<SalesReportingCode1> </SalesReportingCode1>
<SalesReportingCode2> </SalesReportingCode2>
<SalesReportingCode3> </SalesReportingCode3>

```

```

<SalesReportingCode4> </SalesReportingCode4>
<SalesReportingCode5> </SalesReportingCode5>
<PurchasingReportCode1> </PurchasingReportCode1>
<PurchasingReportCode2> </PurchasingReportCode2>
<PurchasingReportCode3> </PurchasingReportCode3>
<PurchasingReportCode4>240</PurchasingReportCode4>
<PurchasingReportCode5> </PurchasingReportCode5>
<UnitOfMeasureAsInput>EA</UnitOfMeasureAsInput>
<UnitsTransactionQty>3</UnitsTransactionQty>
<UnitsQuantityShipped>3</UnitsQuantityShipped>
<UnitsQuanBackorHeld>0</UnitsQuanBackorHeld>
<UnitsQuantityCanceled>0</UnitsQuantityCanceled>
<UnitsQuantityFuture>0</UnitsQuantityFuture>
<UnitsOpenQuantity>0</UnitsOpenQuantity>
<QuantityShippedToDate>0</QuantityShippedToDate>
<QuantityRelieved>0</QuantityRelieved>
<CommittedHS>S</CommittedHS>
<OtherQuantity12> </OtherQuantity12>
<AmtPricePerUnit2>44.9900</AmtPricePerUnit2>
<AmountExtendedPrice>134.97</AmountExtendedPrice>
<AmountOpen1>.00</AmountOpen1>
<PriceOverrideCode> </PriceOverrideCode>
<TemporaryPriceYN> </TemporaryPriceYN>
<UnitOfMeasureEntUP>EA</UnitOfMeasureEntUP>
<AmtListPricePerUnit>44.9900</AmtListPricePerUnit>
<AmountUnitCost>32.1000</AmountUnitCost>
<AmountExtendedCost>96.30</AmountExtendedCost>

```

```

<CostOverrideCode> </CostOverrideCode>
<ExtendedCostTransfer>.0000</ExtendedCostTransfer>
<PrintMessage1> </PrintMessage1>
<PaymentTermsCode01> </PaymentTermsCode01>
<PaymentInstrumentA> </PaymentInstrumentA>
<BasedonDate> </BasedonDate>
<DiscountTrade>.000</DiscountTrade>
<TradeDiscountOld>.0000</TradeDiscountOld>
<PriceAdjustmentScheduleN> </PriceAdjustmentScheduleN>
<PricingCategory> </PricingCategory>
<PricingCategoryLevel1> </PricingCategoryLevel1>
<DiscountFactor>1.0000</DiscountFactor>
<DiscountFactorTypeOr> </DiscountFactorTypeOr>
<DiscntApplicationType> </DiscntApplicationType>
<DiscountCash>.000</DiscountCash>
<CompanyKey> </CompanyKey>
<DocVoucherInvoiceE>0</DocVoucherInvoiceE>
<DocumentType> </DocumentType>
<OriginalDocumentNo>0</OriginalDocumentNo>
<OriginalDocumentType> </OriginalDocumentType>
<DocumentCompanyOriginal> </DocumentCompanyOriginal>
<PickSlipNumber>0</PickSlipNumber>
<DeliveryNumber>0</DeliveryNumber>
<PromationNumber> </PromationNumber>
<DraftNumber>0</DraftNumber>
<TaxableYN>N</TaxableYN>
<TaxArea1>DEN</TaxArea1>

```

```

<TaxExplanationCode1>S</TaxExplanationCode1>
<AssociatedText> </AssociatedText>
<PriorityProcessing>0</PriorityProcessing>
<ResolutionCodeBC> </ResolutionCodeBC>
<BackordersAllowedYN>Y</BackordersAllowedYN>
<SubstitutesAllowedYN>Y</SubstitutesAllowedYN>
<PartialShipmntsAllowY>Y</PartialShipmntsAllowY>
<LineofBusiness> </LineofBusiness>
<EndUse> </EndUse>
<DutyStatus> </DutyStatus>
<CommodityCode> </CommodityCode>
<NatureOfTransaction> </NatureOfTransaction>
<PrimaryLastVendorNo>4343</PrimaryLastVendorNo>
<BuyerNumber>8444</BuyerNumber>
<Carrier>0</Carrier>
<ModeOfTransport> </ModeOfTransport>
<ConditionsOfTransport> </ConditionsOfTransport>
<RouteCode> </RouteCode>
<StopCode> </StopCode>
<ZoneNumber> </ZoneNumber>
<ContainerID> </ContainerID>
<FreightHandlingCode> </FreightHandlingCode>
<ApplyFreightYN>Y</ApplyFreightYN>
<ApplyFreight> </ApplyFreight>
<FreightCalculatedYN> </FreightCalculatedYN>
<RateCodeFreightMisc> </RateCodeFreightMisc>
<RateTypeFreightMisc> </RateTypeFreightMisc>

```

```

<ShippingCommodityClass> </ShippingCommodityClass>
<ShippingConditionsCode> </ShippingConditionsCode>
<SerialNumberLot> </SerialNumberLot>
<UnitOfMeasurePrimary>EA</UnitOfMeasurePrimary>
<UnitsPrimaryQtyOrder>3</UnitsPrimaryQtyOrder>
<UnitOfMeasureSecondary>EA</UnitOfMeasureSecondary>
<UnitsSecondaryQtyOr>3</UnitsSecondaryQtyOr>
<UnitOfMeasurePricing>EA</UnitOfMeasurePricing>
<AmountUnitWeight>240.0000</AmountUnitWeight>
<WeightUnitOfMeasure>OZ</WeightUnitOfMeasure>
<AmountUnitVolume>6.7500</AmountUnitVolume>
<VolumeUnitOfMeasure>FC</VolumeUnitOfMeasure>
<RepriceBasketPriceCat> </RepriceBasketPriceCat>
<OrderRepriceCategory> </OrderRepriceCategory>
<OrderRepricedIndicator> </OrderRepricedIndicator>
<InventoryCostingMeth>07</InventoryCostingMeth>
<AllocateByLot> </AllocateByLot>
<GIClass>IN30</GIClass>
<Century>20</Century>
<FiscalYear1>5</FiscalYear1>
<LineStatus> </LineStatus>
<SalesOrderStatus01> </SalesOrderStatus01>
<SalesOrderStatus02> </SalesOrderStatus02>
<SalesOrderStatus03> </SalesOrderStatus03>
<SalesOrderStatus04> </SalesOrderStatus04>
<SalesOrderStatus05> </SalesOrderStatus05>
<SalesOrderStatus06> </SalesOrderStatus06>

```

```
<SalesOrderStatus07> </SalesOrderStatus07>
<SalesOrderStatus08> </SalesOrderStatus08>
<SalesOrderStatus09> </SalesOrderStatus09>
<SalesOrderStatus10> </SalesOrderStatus10>
<SalesOrderStatus11> </SalesOrderStatus11>
<SalesOrderStatus12> </SalesOrderStatus12>
<SalesOrderStatus13> </SalesOrderStatus13>
<SalesOrderStatus14> </SalesOrderStatus14>
<SalesOrderStatus15> </SalesOrderStatus15>
<Salesperson1>6001</Salesperson1>
<SalespersonCommission1>5.000</SalespersonCommission1>
<Salesperson2>0</Salesperson2>
<SalespersonCommission2>.000</SalespersonCommission2>
<ApplyCommissionYN>Y</ApplyCommissionYN>
<CommissionCategory> </CommissionCategory>
<ReasonCode> </ReasonCode>
<GrossWeight>.0000</GrossWeight>
<UnitOfMeasureGrossWt> </UnitOfMeasureGrossWt>
<AcctNoInputMode> </AcctNoInputMode>
<AccountId> </AccountId>
<PurchasingCostCenter> </PurchasingCostCenter>
<ObjectAccount> </ObjectAccount>
<Subsidiary> </Subsidiary>
<LedgerType> </LedgerType>
<Subledger> </Subledger>
<SubledgerType> </SubledgerType>
<CodeLocationTaxStat> </CodeLocationTaxStat>
```

```

<PriceCode1> </PriceCode1>
<PriceCode2> </PriceCode2>
<PriceCode3> </PriceCode3>
<StatusInWarehouse> </StatusInWarehouse>
<WoOrderFreezeCode> </WoOrderFreezeCode>
<CorrespondenceMethod> </CorrespondenceMethod>
<CurrencyCodeFrom>BEF</CurrencyCodeFrom>
<CurrencyConverRateOv>33.8180588</CurrencyConverRateOv>
<AmountListPriceForeign>1521.4745</AmountListPriceForeign>
<AmtForPricePerUnit>1521.4745</AmtForPricePerUnit>
<AmountForeignExtPrice>4564.42</AmountForeignExtPrice>
<AmountForeignUnitCost>1085.5597</AmountForeignUnitCost>
<AmountForeignExtCost>3256.68</AmountForeignExtCost>
<UserReservedCode> </UserReservedCode>
<UserReservedDate></UserReservedDate>
<UserReservedAmount>.00</UserReservedAmount>
<UserReservedNumber>0</UserReservedNumber>
<UserReservedReference> </UserReservedReference>
<TransactionOriginator>KW6803955</TransactionOriginator>
<UserId>KW6803955</UserId>
<ProgramId>XMLtest</ProgramId>
<WorkStationId>ST15</WorkStationId>
<DateUpdated>2000/08/22</DateUpdated>
<TimeOfDay>134435</TimeOfDay>
</F4211Z1>
<F49211Z1 type='additionalHeader'>
 <EdiUserId>KW6803955</EdiUserId>

```

```

<EdiBatchNumber>16319</EdiBatchNumber>
<EdiTransactNumber>106053</EdiTransactNumber>
<EdiLineNumber>1.000</EdiLineNumber>
<EdiDocumentType>SO</EdiDocumentType>
<TypeTransaction>JDESC</TypeTransaction>
<EdiTranslationFormat> </EdiTranslationFormat>
<EdiTransmissionDate></EdiTransmissionDate>
<DirectionIndicator>2</DirectionIndicator>
<EdiDetailLinesProcess>0</EdiDetailLinesProcess>
<EdiSuccessfullyProcess>N</EdiSuccessfullyProcess>
<TradingPartnerId> </TradingPartnerId>
<TransactionAction>UA</TransactionAction>
<DocumentOrderInvoiceE>6559</DocumentOrderInvoiceE>
<OrderType>SO</OrderType>
<CompanyKeyOrderNo>00200</CompanyKeyOrderNo>
<LineNumber>1.000</LineNumber>
<CostCenterTrip> </CostCenterTrip>
<TripNumber>0</TripNumber>
<DateLoaded></DateLoaded>
<DispatchGrp> </DispatchGrp>
<BulkPackedFlag>P</BulkPackedFlag>
<Distance>0</Distance>
<UnitOfMeasure> </UnitOfMeasure>
<DeferredEntriesFlag> </DeferredEntriesFlag>
<AmountDeferredCost>.0000</AmountDeferredCost>
<AmountForeignDeferredCos>.0000</AmountForeignDeferredCos>
<AmountDeferredRevenue>.0000</AmountDeferredRevenue>

```

```

<AmountForeignDeferredRe> 0000</AmountForeignDeferredRe>
<AaiTableNumber>0</AaiTableNumber>
<ScheduledInvoiceDate></ScheduledInvoiceDate>
<InvoiceCycleCode> </InvoiceCycleCode>
<LoadConfirmDate></LoadConfirmDate>
<TimeLoad>0</TimeLoad>
<DeliveryConfirmDate></DeliveryConfirmDate>
<UnitsPrimaryCommittedQua>0</UnitsPrimaryCommittedQua>
<UnitofMeasureCommittedQu> </UnitofMeasureCommittedQu>
<Temperature>.00</Temperature>
<StrappingTemperatureUnit> </StrappingTemperatureUnit>
<Density>.00</Density>
<DensityTypeAtStandardTem> </DensityTypeAtStandardTem>
<DensityTemperature>.00</DensityTemperature>
<DensityTemperatureUnit> </DensityTemperatureUnit>
<VolumeCorrectionFactors>.0000</VolumeCorrectionFactors>
<PriceatAmbiantorStandard>A</PriceatAmbiantorStandard>
<PricingBasedOnDate> </PricingBasedOnDate>
<UnitsInvoiceQuantity>0</UnitsInvoiceQuantity>
<StockTotalinPrimaryUOM>0</StockTotalinPrimaryUOM>
<UnitofMeasure6> </UnitofMeasure6>
<AmbientResult>0</AmbientResult>
<UnitofMeasure3> </UnitofMeasure3>
<WeightResult>0</WeightResult>
<UnitofMeasure5> </UnitofMeasure5>
<VendorFreightCalculatedY> </VendorFreightCalculatedY>
<CustomerFreightCalculate> </CustomerFreightCalculate>

```

<AmountCustomerFreightCha>.0000</AmountCustomerFreightCha>  
<AmountVendorFreightCharg>.0000</AmountVendorFreightCharg>  
<PrimaryVehicleId> </PrimaryVehicleId>  
<RegistrationLicenseNumbe> </RegistrationLicenseNumbe>  
<CostCenterArDefault> </CostCenterArDefault>  
<FlightNumber> </FlightNumber>  
<Destination> </Destination>  
<AircraftType> </AircraftType>  
<Origin> </Origin>  
<TimeElapsed>0</TimeElapsed>  
<ShipmentNumberB73>0</ShipmentNumberB73>  
<AddressNumberIssued>6074</AddressNumberIssued>  
<PaymentTermsCode01> </PaymentTermsCode01>  
<DocVoucherInvoiceE>0</DocVoucherInvoiceE>  
<DocumentType> </DocumentType>  
<CompanyKey> </CompanyKey>  
<CurrencyConverRateOv>-1.0000000</CurrencyConverRateOv>  
<CurrencyCodeFrom> </CurrencyCodeFrom>  
<TaxArea1>DEN</TaxArea1>  
<TaxExplanationCode1> </TaxExplanationCode1>  
<ForeignDomesticFlag> </ForeignDomesticFlag>  
<FuelingPort> </FuelingPort>  
<RegistrationIdentificati> </RegistrationIdentificati>  
<DeliveryLocationN> </DeliveryLocationN>  
<AuthorizationName> </AuthorizationName>  
<NameAlpha> </NameAlpha>  
<MeterTicket1> </MeterTicket1>

```

<UnitsBeginningThroughput>0</UnitsBeginningThroughput>
<ClosingReading1>0</ClosingReading1>
<MeterTicket2> </MeterTicket2>
<UnitsBeginningThroughput2>0</UnitsBeginningThroughput2>
<ClosingReading2>0</ClosingReading2>
<MeterTicket3> </MeterTicket3>
<UnitsBeginningThroughput3>0</UnitsBeginningThroughput3>
<ClosingReading3>0</ClosingReading3>
<DateArrival></DateArrival>
<TimeArrival>0</TimeArrival>
<DateDeparture></DateDeparture>
<TimeDeparture>0</TimeDeparture>
<DateStartJobJulian></DateStartJobJulian>
<TimeBeginningHHMM>0</TimeBeginningHHMM>
<DateEnding></DateEnding>
<TimeStopHHMM>0</TimeStopHHMM>
<FutureUse01t> </FutureUse01t>
<FutureUse02t> </FutureUse02t>
<FutureUse03t> </FutureUse03t>
<FutureUse04> </FutureUse04>
<FutureUse05> </FutureUse05>
<FutureUseCode> </FutureUseCode>
<FutureUseQuantityt>0</FutureUseQuantityt>
<FutureUseDate></FutureUseDate>
<FutureUseUnitofMeasure> </FutureUseUnitofMeasure>
<UserReservedCode> </UserReservedCode>
<UserReservedDate></UserReservedDate>

```

```

<UserReservedAmount>.00</UserReservedAmount>
<UserReservedNumber>0</UserReservedNumber>
<UserReservedReference> </UserReservedReference>
<TransactionOriginator> </TransactionOriginator>
<UserId>KW6803955</UserId>
<ProgramId>XMLtest</ProgramId>
<WorkStationId>STI5</WorkStationId>
<DateUpdated>2000/08/22</DateUpdated>
<TimeOfDay>134435</TimeOfDay>
</F49211Z1>
</transaction>
</jdeResponse>

```

---

## Example: Real-Time Events Template

The following provides an example of the real-time events template. The example template might not correspond to the exact event that your application uses. Your event might include values that are not in the example template.

```

//The event must be described in the jdeResponse type element. The attribute type
is always "realTimeEvent". The attributes for user and environment always
correspond to the username and environment that generated the event.

<?xml version="1.0" encoding="utf-8" ?>
<jdeResponse type="realTimeEvent" user="" session="28980548.1019684006"
 environment="">
 <event>
 <header>

//Code for the header information follows. <eventVersion> is always 1.0, <type>
corresponds to the event type, <application> corresponds to the application that
created the event, and <version> to the version of the application. The <sessionID>
is unique for every event. The <scope> is the value of the argument scope that was
sent to the real-time event API during creation of the event. The <codepage>
element is for encoding of the elements; in the sample utf-8 is used. The remaining
header elements are self-explanatory.

 <eventVersion>1.0</eventVersion>
 <type>RTSOOUT</type>
 <user />
 <application />
 <version />
 <sessionID />
 <environment />
 <host />

```

```

<sequenceID />
<date />
<time />
<scope />
<codepage>utf-8</codepage>
</header>

//The body contains details that describe one data structure for each element. The
body contains the date of creation, the name of the file that is creating the data
structure, time of creation, and the DSTMPL name of the PeopleSoft EnterpriseOne
data structure. Type is type of partial event (added as an argument to jdeIEO-
_EventAdd), executionOrder increases in the real generated event from 1 to
elementCount, and parameterCount is the number of fields in the data structure. In
the following example code, there are three data structures: D34A1050C, D4202150C,
and D4202150B. Each data structure is followed by detail elements. When you create
an event, the element value is the value of the field, for example: <szNameAlpha
type="String">ABC</szNameAlpha >

<body elementCount="3">
<detail date="" name="" time="" type="" DSTMPL="D34A1050C"
 executionOrder="" parameterCount="25">
 <szNameAlpha type="String" />
 <mnParentAddressNumber type="Double" />
 <szSecondItemNumber type="String" />
 <szThirdItemNumber type="String" />
 <cPriorityProcessing type="Character" />
 <cBackOrdersAllowed type="Character" />
 <cOrderShippedFlag type="Character" />
 <cTransferDirectShipFlag type="Character" />
 <cCommitted type="Character" />
 <mnDaysBeforeExpiration type="Double" />
 <szPurchaseCategoryCode1 type="String" />
 <szPurchaseCategoryCode2 type="String" />
 <szPurchaseCategoryCode3 type="String" />
 <szPurchaseCategoryCode4 type="String" />
 <szRelatedOrderNumber type="String" />
 <szRelatedOrderType type="String" />
 <szRelatedOrderKeyCompany type="String" />
 <szPlanningUnitOfMeasure type="String" />
 <mnPlanningQuantity type="Double" />
 <cAPSFlag type="Character" />
 <cAPSSupplyDemandFlag type="Character" />
 <jdDateUpdated type="Date" />
 <mnTimeUpdated type="Double" />
 <szShipComplete type="String" />
 <mnRelatedOrderLineNumber type="Double" />
</detail>
<detail date="" name="" time="" type="" DSTMPL="D4202150C"
 executionOrder="" parameterCount="94">
 <cOrderAction type="Character" />
 <szOrderType type="String" />
 <szOrderCompany type="String" />
 <mnLineNumber type="Double" />
 <szDetailBranchPlant type="String" />
 <mnShipToAddressNumber type="Double" />

```

```

<jdTransactionDate type="Date" />
<jdRequestedDate type="Date" />
<jdScheduledPickDate type="Date" />
<jdPromisedShipDate type="Date" />
<jdPromisedDeliveryDate type="Date" />
<jdCancelDate type="Date" />
<jdPriceEffectiveDate type="Date" />
<mnQuantityOrdered type="Double" />
<mnQuantityShipped type="Double" />
<mnQuantityBackOrdered type="Double" />
<mnQuantityCanceled type="Double" />
<szTransactionUnitOfMeasure type="String" />
<mnUnitPrice type="Double" />
<mnExtendedPrice type="Double" />
<mnForeignUnitPrice type="Double" />
<mnForeignExtPrice type="Double" />
<cPriceOverrideCode type="Character" />
<cTaxableYN type="Character" />
<szPriceAdjustmentSchedule type="String" />
<mnDiscountPercentage type="Double" />
<szPaymentTerms type="String" />
<cPaymentInstrument type="Character" />
<szCurrencyCode type="String" />
<szItemNumber type="String" />
<mnShortItemNumber type="Double" />
<szDescriptionLine1 type="String" />
<szDescriptionLine2 type="String" />
<szLineType type="String" />
<szLastStatus type="String" />
<szNextStatus type="String" />
<szLocation type="String" />
<szLot type="String" />
<szLineofBusiness type="String" />
<szEndUse type="String" />
<szDutyStatus type="String" />
<szPrintMessage1 type="String" />
<szFreightHandlingCode type="String" />
<mnItemWeight type="Double" />
<szWeightUnitOfMeasure type="String" />
<szModeOfTransport type="String" />
<mnCarrier type="Double" />
<szSubledger type="String" />
<cSubledgerType type="Character" />
<szPriceCode1 type="String" />
<szPriceCode2 type="String" />
<szPriceCode3 type="String" />
<szSalesReportingCode1 type="String" />
<szSalesReportingCode2 type="String" />
<szSalesReportingCode3 type="String" />
<szSalesReportingCode4 type="String" />
<szSalesReportingCode5 type="String" />
<szOriginalPoSoNumber type="String" />
<szOriginalOrderType type="String" />
<szOriginalOrderCompany type="String" />
<mnOriginalOrderLineNumber type="Double" />

```

```

<jdDateUpdated type="Date" />
<mnTimeOfDay type="Double" />
<mnPickSlipNumber type="Double" />
<mnInvoiceDocNumber type="Double" />
<szInvoiceDocType type="String" />
<szInvoiceDocCompany type="String" />
<szUserReservedCode type="String" />
<jdUserReservedDate type="Date" />
<mnUserReservedNumber type="Double" />
<mnUserReservedAmount type="Double" />
<szUserReservedReference type="String" />
<mnUnitCost type="Double" />
<mnExtendedCost type="Double" />
<mnForeignUnitCost type="Double" />
<mnForeignExtCost type="Double" />
<mnOrderNumber type="Double" />
<szSupplierReference type="String" />
<jdOriginalPromisdDate type="Date" />
<mnAdjustmentRevisionLevel type="Double" />
<mnLastIndex type="Double" />
<szRelatedPoSoNumber type="String" />
<szRelatedOrderType type="String" />
<szRelatedOrderCompany type="String" />
<mnRelatedPoSoLineNo type="Double" />
<szPricingUnitOfMeasure type="String" />
<szTaxArea type="String" />
<szTaxExplanationCode type="String" />
<szPartnerItemNo type="String" />
<szCatalogItem type="String" />
<szUPCNumber type="String" />
<szShipToDescriptive type="String" />
<szSoldToDescriptive type="String" />
<szProductItem type="String" />
</detail>
<detail date="" name="" time="" type="" DSTMPL="D4202150B"
 executionOrder="" parameterCount="66">
 <cOrderAction type="Character" />
 <mnOrderNumber type="Double" />
 <szOrderType type="String" />
 <szOrderCompany type="String" />
 <szHeaderBranchPlant type="String" />
 <szCompany type="String" />
 <szOriginalPoSoNumber type="String" />
 <szOrderedBy type="String" />
 <szOrderTakenBy type="String" />
 <mnSoldToAddressNumber type="Double" />
 <szSoldToNameMailing type="String" />
 <szSoldToAddressLine1 type="String" />
 <szSoldToAddressLine2 type="String" />
 <szSoldToAddressLine3 type="String" />
 <szSoldToAddressLine4 type="String" />
 <szSoldToZipCode type="String" />
 <szSoldToCity type="String" />
 <szSoldToCounty type="String" />
 <szSoldToState type="String" />

```

```

<szSoldToCountry type="String" />
<mnShipToAddressNumber type="Double" />
<szShipToNameMailing type="String" />
<szShipToAddressLine1 type="String" />
<szShipToAddressLine2 type="String" />
<szShipToAddressLine3 type="String" />
<szShipToAddressLine4 type="String" />
<szShipToZipCode type="String" />
<szShipToCity type="String" />
<szShipToCounty type="String" />
<szShipToState type="String" />
<szShipToCountry type="String" />
<jdTransactionDate type="Date" />
<jdRequestedDate type="Date" />
<jdCancelDate type="Date" />
<szReference type="String" />
<szDeliveryInstructLine1 type="String" />
<szDeliveryInstructLine2 type="String" />
<szPrintMessage type="String" />
<szFreightHandlingCode type="String" />
<mnCommissionCode1 type="Double" />
<mnCommissionCode2 type="Double" />
<mnRateCommission1 type="Double" />
<mnRateCommission2 type="Double" />
<mnDiscountTrade type="Double" />
<szPaymentTerms type="String" />
<cPaymentInstrument type="Character" />
<szCurrencyCode type="String" />
<mnCurrencyConverRate type="Double" />
<szTaxArea type="String" />
<szTaxExplanationCode type="String" />
<mnOrderTotal type="Double" />
<mnForeignOrderTotal type="Double" />
<szUserReservedCode type="String" />
<jdUserReservedDate type="Date" />
<mnUserReservedAmount type="Double" />
<mnUserReservedNumber type="Double" />
<szUserReservedReference type="String" />
<szHoldCode type="String" />
<cQuoteFlag type="Character" />
<jdScheduledPickDate type="Date" />
<jdPromisedShipDate type="Date" />
<jdOriginalPromisdDate type="Date" />
<cCurrencyMode type="Character" />
<szShipToDescriptive type="String" />
<szSoldToDescriptive type="String" />
<cPublishToXPiXFlag type="Character" />
</detail>
</body>
</event>
</jdeResponse>

```

The following table shows the mapping between EnterpriseOne types and events:

CHAR	Character
STRING	String
MATH_numeric	Double
JDEDATE	Dat"
SHORT	Int
INT	Int
USHORT	Int
LONG	Long
ULONG	Long
ID	"Long
ID2	Long
BOOL	BOOL

# EnterpriseOne PeopleBooks Glossary

<b>“as of” processing</b>	A process that is run at a specific point in time to summarize item transactions.
<b>52 period accounting</b>	A method of accounting that uses each week as a separate accounting period.
<b>account site</b>	In the invoice process, the address to which invoices are mailed. Invoices can go to a different location or account site from the statement.
<b>active window</b>	The window that contains the document or display that will be affected by current cursor movements, commands, and data entry in environments that are capable of displaying multiple on-screen windows.
<b>ActiveX</b>	A technology and set of programming tools developed by Microsoft Corporation that enable software components written in different languages to interact with each another in a network environment or on a web page. The technology, based on object linking and embedding, enables Java applet-style functionality for Web browsers as well as other applications (Java is limited to Web browsers at this time). The ActiveX equivalent of a Java applet is an ActiveX control. These controls bring computational, communications, and data manipulation power to programs that can “contain” them—for example, certain Web browsers, Microsoft Office programs, and anything developed with Visual Basic or Visual C++.
<b>activity</b>	In Advanced Cost Accounting, an aggregation of actions performed within an organization that is used in activity-based costing.
<b>activity driver</b>	A measure of the frequency and intensity of the demands that are placed on activities by cost objects. An activity driver is used to assign costs to cost objects. It represents a line item on the bill of activities for a product or customer. An example is the number of part numbers, which is used to measure the consumption of material-related activities by each product, material type, or component. The number of customer orders measures the consumption of order-entry activities by each customer. Sometimes an activity driver is used as an indicator of the output of an activity, such as the number of purchase orders that are prepared by the purchasing activity. See also cost object.
<b>activity rule</b>	The criteria by which an object progresses from a given point to the next in a flow.
<b>actual cost</b>	Actual costing uses predetermined cost components, but the costs are accumulated at the time that they occur throughout the production process.
<b>adapter</b>	A component that connects two devices or systems, physically or electronically, and enables them to work together.
<b>add mode</b>	The condition of a form where a user can enter data into it.
<b>advanced interactive executive</b>	An open IBM operating system that is based on UNIX.
<b>agent</b>	A program that searches through archives or other repositories of information on a topic that is specified by the user.

<b>aging</b>	A classification of accounts by the time elapsed since the billing date or due date. Aging is divided into schedules or accounting periods, such as 0-30 days, 31-60 days, and so on.
<b>aging schedule</b>	A schedule that is used to determine whether a payment is delinquent and the number of days which the payment is delinquent.
<b>allegato IVA clienti</b>	In Italy, the term for the A/R Annual VAT report.
<b>allegato IVA fornitori</b>	In Italy, the term for the A/P Annual VAT report.
<b>application layer</b>	The seventh layer of the Open Systems Interconnection Reference Model, which defines standards for interaction at the user or application program level.
<b>application programming interface (API)</b>	A set of routines that is used by an application program to direct the performance of procedures by the computer's operating system.
<b>AS/400 Common</b>	A data source that resides on an AS/400 and holds data that is common to the co-existent library, allowing PeopleSoft EnterpriseOne to share information with PeopleSoft World.
<b>assembly inclusion rule</b>	A logic statement that specifies the conditions for using a part, adjusting the price or cost, performing a calculation, or using a routing operation for configured items.
<b>audit trail</b>	The detailed, verifiable history of a processed transaction. The history consists of the original documents, transaction entries, and posting of records and usually concludes with a report.
<b>automatic return</b>	A feature that allows a user to move to the next entry line in a detail area or to the first cell in the next row in several applications.
<b>availability</b>	The expression of the inventory amount that can be used for sales orders or manufacturing orders.
<b>available inventory</b>	The quantity of product that can be promised for sale or transfer at a particular time, considering current on-hand quantities, replenishments in process, and anticipated demand.
<b>back office</b>	The set of enterprise software applications that supports the internal business functions of a company.
<b>backhaul</b>	The return trip of a vehicle after delivering a load to a specified destination. The vehicle can be empty or the backhaul can produce less revenue than the original trip. For example, the state of Florida is considered a backhaul for many other states—that is, many trucking companies ship products into the state of Florida, but most of them cannot fill a load coming out of Florida or they charge less. Hence, trucks coming out of Florida are either empty or produce less revenue than the original trip.
<b>balance forward</b>	The cumulative total of inventory transactions that is used in the Running Balance program. The system does not store this total. You must run this program each time that you want to review the cumulative inventory transactions total.
<b>balance forward receipt application method</b>	A receipt application method in which the receipt is applied to the oldest or newest invoices in chronological order according to the net due date.

<b>bank tape (lock box) processing</b>	The receipt of payments directly from a customer's bank via customer tapes for automatic receipt application.
<b>base location</b>	[In package management] The topmost location that is displayed when a user launches the Machine Identification application.
<b>basket discount</b>	A reduction in price that applies to a group or "basket" of products within a sales order.
<b>basket repricing</b>	A rule that specifies how to calculate and display discounts for a group of products on a sales order. The system can calculate and display the discount as a separate sales order detail line, or it can discount the price of each item on a line-by-line basis within the sales order.
<b>batch job</b>	A job submitted to a system and processed as a single unit with no user interaction.
<b>batch override</b>	An instruction that causes a batch process to produce output other than what it normally would produce for the current execution only.
<b>batch process</b>	A type of process that runs to completion without user intervention after it has been started.
<b>batch program</b>	A program that executes without interacting with the user.
<b>batch version</b>	A version of a report or application that includes a set of user-defined specifications, which control how a batch process runs.
<b>batch/lot tracking</b>	The act of identifying where a component from a specific lot is used in the production of goods.
<b>batch/mix</b>	A manufacturing process that primarily schedules short production runs of products.
<b>batch-of-one processing</b>	A transaction method that allows a client application to perform work on a client workstation, and then submit the work all at once to a server application for further processing. As a batch process is running on the server, the client application can continue performing other tasks. See also direct connect, store-and-forward.
<b>binary large object (BLOB)</b>	A collection of binary data stored as a single entity in a [file].
<b>binder clip</b>	See paper clip.
<b>black products</b>	Products that are derived from the low or heavy end of the distillation process—for example, diesel oils and fuel oils. See also white products.
<b>blend note</b>	Document that authorizes a blending activity, and describes both the ingredients for the blend and the blending steps that occur.
<b>blend off</b>	Reworking off-specification material by introducing a small percentage back into another run of the same product.
<b>blind execution</b>	The mode of execution of a program that does not require the user to review or change the processing options set for the program, and does not require user intervention after the program has been launched.

<b>boleto</b>	In Brazil, the document requesting payment by a supplier or a bank on behalf of a supplier.
<b>bolla doganale</b>	VAT-Only Vouchers for Customs. In Italy, a document issued by the customs authority to charge VAT and duties on extra-EU purchasing.
<b>bookmark</b>	A shortcut to a location in a document or a specific place in an application or application suite.
<b>bordero &amp; cheque</b>	In Brazil, bank payment reports.
<b>broker</b>	A program that acts as an intermediary between clients and servers to coordinate and manage requests.
<b>BTL91</b>	In the Netherlands, the ABN/AMRO electronic banking file format that enables batches with foreign automatic payment instructions to be delivered.
<b>budgeted volume</b>	A statement of planned volumes (capacity utilization) upon which budgets for the period have been set.
<b>bunkering</b>	A rate per ton or a sum of money that is charged for placing fuel on board; can also mean the operation itself.
<b>business function</b>	An encapsulated set of business rules and logic that can normally be re-used by multiple applications. Business functions can execute a transaction or a subset of a transaction (check inventory, issue work orders, and so on). Business functions also contain the APIs that allow them to be called from a form, a database trigger, or a non-EnterpriseOne application. Business functions can be combined with other business functions, forms, event rules, and other components to make up an application. Business functions can be created through event rules or third-generation languages, such as C. Examples of business functions include Credit Check and Item Availability.
<b>business function event rule</b>	Encapsulated, reusable business logic that is created by using through event rules rather than C programming. Contrast with embedded event rule. See also event rule.
<b>business object library</b>	[In interoperability] The repository that stores EnterpriseOne business objects, which consist of Java or CORBA objects.
<b>business unit</b>	A financial entity that is used to track the costs, revenue, or both, of an organization. A business unit can also be defined as a branch/plant in which distribution and manufacturing activities occur. Additionally, in manufacturing setup, work centers and production lines must be defined as business units; but these business unit types do not have profit/loss capability.
<b>business view</b>	Used by EnterpriseOne applications to access data from database tables. A business view is a means for selecting specific columns from one or more tables with data that will be used in an application or report. It does not select specific rows and does not contain any physical data. It is strictly a view through which data can be handled.
<b>business view design aid (BDA)</b>	An EnterpriseOne GUI tool for creating, modifying, copying, and printing business views. The tool uses a graphical user interface.

<b>buy-back crude</b>	In foreign producing oil countries, that portion of the host government's share of "participation crude" which it permits the company holding a concession to "buy back."
<b>CAB</b>	In Italy, the bank branch code or branch ID. A five-digit number that identifies any agency of a specific bank company in Italy.
<b>cadastro de pessoas físicas</b>	Cadastro de pessoas físicas. In Brazil, the federal tax ID for a person.
<b>category code</b>	A code that identifies a collection of objects sharing at least one common attribute.
<b>central object</b>	A software component that resides on a central server.
<b>central objects merge</b>	A process that blends a customer's modifications with the objects in a current release with objects in a new release.
<b>central server</b>	A computer that has been designated to contain the originally installed version of the software (central objects) for deployment to client computers.
<b>certificate input</b>	See direct input.
<b>certificate of analysis (COA)</b>	A document that is a record of all of the testing which has been performed against an item, lot, or both, plus the test results for that item and lot.
<b>change management</b>	[In software development] A process that aids in controlling and tracking the evolution of software components.
<b>change order</b>	In PeopleSoft, an addendum to the original purchase order that reflects changes in quantities, dates, or specifications in subcontract-based purchasing. A change order is typically accompanied by a formal notification.
<b>chargeback</b>	A receipt application method that generates an invoice for a disputed amount or for the difference of an unpaid receipt.
<b>chart</b>	EnterpriseOne term for tables of information that appear on forms in the software. See forms.
<b>check-in location</b>	The directory structure location for the package and its set of replicated objects. This location is usually \\deploymentserver\release\path_code\package\packagename. The subdirectories under this path are where the central C components (source, include, object, library, and DLL file) for business functions are stored.
<b>checksum value</b>	A computed value that depends on the contents of a block of data, and that is transmitted or stored with the data to detect whether errors have occurred in the transmission or storage.
<b>class</b>	[In object-oriented programming] A category of objects that share the same characteristics.
<b>clean cargo</b>	Term that refers to cargoes of gasoline and other refined products. See also dirty cargo.
<b>client access</b>	The ability to access data on a server from a client machine.
<b>client machine</b>	Any machine that is connected to a network and that exchanges data with a server.

<b>client workstation</b>	A network computer that runs user application software and is able to request data from a server.
<b>ClieOp03</b>	In the Netherlands, the euro-compliant uniform electronic banking file format that enables batches with domestic automatic direct debit instructions and batches with domestic payment instructions to be delivered.
<b>ClieOp2</b>	In the Netherlands, the uniform electronic banking file format that enables batches with domestic automatic direct debit instructions and batches with domestic payment instructions to be delivered.
<b>cluster</b>	Two or more computers that are grouped together in such a way that they behave like a single computer.
<b>co-existence</b>	A condition where two or more applications or application suites access one or more of the same database tables within the same enterprise.
<b>cold test</b>	The temperature at which oil becomes solid. Generally considered to be 5 degrees F lower than the pour point.
<b>commitment</b>	The number of items that are reserved to fill demand.
<b>common object request broker architecture</b>	An object request broker standard that is endorsed by the Object Management Group.
<b>compa-ratio</b>	An employee's salary divided by the midpoint amount for the employee's pay grade.
<b>component changeout</b>	See component swap.
<b>component object model (COM)</b>	A specification developed by Microsoft for building software components that can be assembled into programs or add functionality to existing programs running on Microsoft Windows platforms. COM components can be written in a variety of languages, although most are written in C++, and can be unplugged from a program at runtime without having to recompile the program.
<b>component swap</b>	In Equipment/Plant Management, the substitution of an operable component for one that requires maintenance. Typically, you swap components to minimize equipment downtime while servicing one of the components. A component swap can also mean the substitution of one parent or component item for another in its associated bill of material.
<b>conference room pilot environment</b>	An EnterpriseOne environment that is used as a staging environment for production data, which includes constants and masters tables such as company constants, fiscal date patterns, and item master. Use this environment along with the test environment to verify that your configuration works before you release changes to end-users.
<b>configurable network computing (CNC)</b>	An application architecture that allows interactive and batch applications that are composed of a single code base to run across a TCP/IP network of multiple server platforms and SQL databases. The applications consist of re-usable business functions and associated data that can be configured across the network dynamically. The overall objective for businesses is to provide a future-proof environment that enables them to change organizational structures, business processes, and technologies independently of each other.

<b>configurable processing engine</b>	Handles all “batch” processes, including reporting, Electronic Data Exchange (EDIt) transactions, and data duplication and transformation (for data warehousing). This ability does not mean that it exists only on the server; it can be configured to run on desktop machines (Windows 95 and NT Workstation) as well.
<b>configuration management</b>	A rules-based method of ordering assemble-to-order or make-to-order products in which characteristics of the product are defined as part of the Sales Order Entry process. Characteristics are edited by using Boolean logic, and then translated into the components and routing steps that are required to produce the product. The resulting configuration is also priced and costed, based on the defined characteristics.
<b>configured item segment</b>	A characteristic of a configured item that is defined during sales order entry. For example, a customer might specify a type of computer hard drive by stating the number of megabytes of the hard drive, rather than a part number.
<b>consuming location</b>	The point in the manufacturing routing where a component or subassembly is used in the production process. In kanban processing, the location where the kanban container materials are used in the manufacturing process and the kanban is checked out for replenishment.
<b>contra/clearing account</b>	A G/L account used by the system to offset (balance) journal entries. For example, you can use a contra/clearing account to balance the entries created by allocations.
<b>contribution to profit</b>	Selling price of an item minus its variable costs.
<b>control table</b>	A table that controls the program flow or plays a major part in program control.
<b>control table workbench</b>	During the Installation Workbench process, Control Table Workbench runs the batch applications for the planned merges that update the data dictionary, user defined codes, menus, and user overrides tables.
<b>control tables merge</b>	A process that blends a customer’s modifications to the control tables with the data that accompanies a new release.
<b>corrective work order</b>	A work order that is used to formally request unscheduled maintenance and communicate all of the details pertaining to the requested maintenance task.
<b>corrective work order</b>	A work order that is used to formally request unscheduled maintenance and communicate all of the details pertaining to the requested maintenance task.
<b>cost assignment</b>	Allocating resources to activities or cost objects.
<b>cost component</b>	An element of an item’s cost—for example, material, labor, or overhead.
<b>cost object</b>	Any customer, product, service, contract, project, or other work unit for which you need a separate cost measurement.
<b>cost rollup</b>	A simulated scenario in which work center rates, material costs, and labor costs are used to determine the total cost of an item.
<b>costing elements</b>	The individual classes of added value or conversion costs. These elements are typically materials, such as raw and packaging; labor and machine costs; and overhead, such as fixed and variable. Each corporation defines the necessary detail of product costs by defining and tracking cost categories and subcategories.

<b>credit memo</b>	A negative amount that is used to correct a customer's statement when he or she is overcharged.
<b>credit notice</b>	The physical document that is used to communicate the circumstances and value of a credit order.
<b>credit order</b>	A credit order is used to reflect products or equipment that is received or returned so that it can be viewed as a sales order with negative amounts. Credit orders usually add the product back into inventory. This process is linked with delivery confirmation.
<b>cross segment edit</b>	A logic statement that establishes the relationship between configured item segments. Cross segment edits are used to prevent ordering of configurations that cannot be produced.
<b>crude oil assay</b>	A procedure for determining the distillation curve and quality characteristics of a crude oil.
<b>cumulative update</b>	A version of software that includes fixes and enhancements that have been made since the last release or update.
<b>currency relationships</b>	When converting amounts from one currency to another, the currency relationship defines the from currency and the to currency in PeopleSoft software. For example, to convert amounts from German marks to the euro, you first define a currency relationship between those two currencies.
<b>currency restatement</b>	The process of converting amounts from one currency into another currency, generally for reporting purposes. It can be used, for example, when many currencies must be restated into a single currency for consolidated reporting.
<b>current cost</b>	The cost that is associated with an item at the time a parts list and routing are attached to a work order or rate schedule. Current cost is based on the latest bill of material and routing for the item.
<b>customer pricing rules</b>	In Procurement, the inventory pricing rules that are assigned to a supplier. In Sales, inventory pricing rules that are assigned to a customer.
<b>D.A.S. 2 Reporting (DAS 2 or DADS 1)</b>	In France, the name of the official form on which a business must declare fees and other forms of remuneration that were paid during the fiscal year.
<b>data dictionary</b>	A dynamic repository that is used for storing and managing a specific set of data item definitions and specifications.
<b>data source workbench</b>	During the Installation Workbench process, Data Source Workbench copies all of the data sources that are defined in the installation plan from the Data Source Master and Table and Data Source Sizing tables in the Planner data source to the System - release number data source. It also updates the Data Source Plan detail record to reflect completion.
<b>data structure</b>	A description of the format of records in a database such as the number of fields, valid data types, and so on.
<b>data types</b>	Supplemental information that is attached to a company or business unit. Narrative type contains free-form text. Code type contains dates, amounts, and so on.

<b>datagram</b>	A self-contained packet of information that is forwarded by routers, based on their address and the routing table information.
<b>date pattern</b>	A period of time that is set for each period in standard and 52-period accounting and forecasting.
<b>DCE</b>	See distributed computing environment.
<b>DEB</b>	See déclaration d'échange de biens.
<b>debit memo</b>	In Accounts Payable, a voucher that is entered with a negative amount. Enter this type of voucher when a supplier sends you a credit so that you can apply the amount to open vouchers when you issue payment to the supplier.
<b>debit memo</b>	A form that is issued by a customer, requesting an adjustment of the amount, which is owed to the supplier.
<b>debit statement</b>	A list of debit balances.
<b>de-blend</b>	When blend off does not result in a product that is acceptable to customers. The further processing of product to adjust specific physical and chemical properties to within specification ranges. See also blend off.
<b>déclaration d'échange de biens (DEB)</b>	The French term that is used for the Intrastat report.
<b>delayed billing</b>	The invoicing process is delayed until the end of a designated period.
<b>delta load</b>	A batch process that is used to compare and update records between specified environments.
<b>denominated-in currency</b>	The company currency in which financial reports are based.
<b>deployment server</b>	A server that is used to install, maintain, and distribute software to one or more enterprise servers and client workstations.
<b>detail</b>	The specific information that makes up a record or transaction. Contrast with summary.
<b>detail information</b>	Information that primarily relates to individual lines in a sales or purchase order.
<b>direct connect</b>	A transaction method in which a client application communicates interactively and directly with a server application. See also batch-of-one immediate, store-and-forward.
<b>direct input</b>	The system calculates the net units when you enter gross volume, temperature, and gravity or density. This data is generally entered during product receiving from the certificate that is prepared by an independent inspector.
<b>direct ship orders</b>	A purchase order that is issued to a third-party supplier who designates the destination as the customer. A direct ship sales order is also created for the customer. Direct ship orders occur when a product is not available from a company-owned or company-operated source, so the system creates an order to ship the product from a third-party source directly to the customer. Sometimes referred to as a drop ship or third-party supply.
<b>direct usage</b>	Consumption of resources that are attributable to specific production runs because the resources were directly issued to the schedule/order.

<b>director</b>	An EnterpriseOne user interface that guides a user interactively through an EnterpriseOne process.
<b>dirty cargo</b>	Term that refers to crude oil cargoes or other non-refined petroleum cargoes. See also clean cargo.
<b>dispatch planning</b>	Efficient planning and scheduling of product deliveries. Considerations include: Dispatch groups Scheduled delivery date Scheduled delivery time Preferred delivery date Preferred delivery time Average delivery time for that geographical location Available resources Special equipment requirements at the product's source or destination.
<b>displacement days</b>	The number of days that are calculated from today's date by which you group vouchers for payment. For example, if today's date is March 10 and you specify three displacement days, the system includes vouchers with a due date through March 13 in the payment group. Contrast with pay-through date.
<b>display sequence</b>	A number that the system uses to re-order a group of records on the form.
<b>distributed computing environment (DCE)</b>	A set of integrated software services that allows software which is running on multiple computers to perform seamless and transparently to the end-users. DCE provides security, directory, time, remote procedure calls, and files across computers running on a network.
<b>distributed data processing</b>	Processing in which some of the functions are performed across two or more linked facilities or systems.
<b>distributed database management system (DDBMS)</b>	A system for distributing a database and its control system across many geographically dispersed machines.
<b>do not translate (DNT)</b>	A type of data source that must exist on the AS/400 because of BLOB restrictions.
<b>double-byte character set (DBCS)</b>	A method of representing some characters by using one byte and other characters by using two bytes. Double-byte character sets are necessary to represent some characters in the Japanese, Korean, and Chinese languages.
<b>downgrade profile</b>	A statement of the hierarchy of allowable downgrades. Includes substitutions of items, and meeting tighter specifications for those products with wider or overlapping specification ranges.
<b>DTA</b>	Datenträgeraustausch. A Swiss payment format that is required by Telekurs (Payserv).
<b>dual pricing</b>	To provide prices for goods and services in two currencies. During the euro transition period, dual pricing between the euro and Economic and Monetary Union (EMU) member currencies is encouraged.

<b>dynamic link library (DLL)</b>	A set of program modules that are designed to be invoked from executable files when the executable files are run, without having to be linked to the executable files. They typically contain commonly used functions.
<b>dynamic partitioning</b>	The ability to dynamically distribute logic or data to multiple tiers in a client/server architecture.
<b>economy of scale</b>	A phenomenon whereby larger volumes of production reduce unit cost by distributing fixed costs over a larger quantity. Variable costs are constant; but fixed costs per unit are reduced, thereby reducing total unit cost.
<b>edit mode</b>	A processing mode or condition where the user can alter the information in a form.
<b>edit rule</b>	A method that is used for formatting user entries, validating user entries, or both, against a predefined rule or set of rules.
<b>embedded event rule</b>	An event rule that is specific to a particular table or application. Examples include form-to-form calls, hiding a field that is based on a processing option value, or calling a business function. Contrast with business function event rule. See also event rule.
<b>employee work center</b>	A central location for sending and receiving all EnterpriseOne messages (system and user-generated), regardless of the originating application or user. Each user has a mailbox that contains workflow and other messages, including Active Messages. With respect to workflow, the Message Center is MAPI compliant and supports drag-and-drop work reassignment, escalation, forward and reply, and workflow monitoring. All messages from the message center can be viewed through EnterpriseOne messages or Microsoft Exchange.
<b>Emulator</b>	An item of software or firmware that allows one device to imitate the functioning of another.
<b>encapsulation</b>	The ability to confine access to and manipulation of data within an object to the procedures that contribute to the definition of that object.
<b>engineering change order (ECO)</b>	A work order document that is used to implement and track changes to items and resulting assemblies. The document can include changes in design, quantity of items required, and the assembly or production process.
<b>enhanced analysis database</b>	A database containing a subset of operational data. The data on the enhanced analysis database performs calculations and provides summary data to speed generation of reports and query response times. This solution is appropriate when external data must be added to source data, or when historical data is necessary for trend analysis or regulatory reporting. See also duplicated database, enterprise data warehouse.
<b>enterprise server</b>	A computer containing programs that collectively serve the needs of an enterprise rather than a single user, department, or specialized application.
<b>EnterpriseOne object</b>	A re-usable piece of code that is used to build applications. Object types include tables, forms, business functions, data dictionary items, batch processes, business views, event rules, versions, data structures, and media objects. See also object.

<b>EnterpriseOne process</b>	Allows EnterpriseOne clients and servers to handle processing requests and execute transactions. A client runs one process, and servers can have multiple instances of a process. EnterpriseOne processes can also be dedicated to specific tasks (for example, workflow messages and data replication) to ensure that critical processes do not have to wait if the server is particularly busy.
<b>EnterpriseOne web development computer</b>	A standard EnterpriseOne Windows developer computer with the additional components installed:  Sun's JDK 1.1.  JFC (0.5.1).  Generator Package with Generator.Java and JDECOM.dll.  R2 with interpretive and application controls/form.
<b>environment workbench</b>	During the Installation Workbench process, Environment Workbench copies the environment information and Object Configuration Manager tables for each environment from the Planner data source to the System release number data source. It also updates the Environment Plan detail record to reflect completion.
<b>equivalent fuel</b>	A barrel of equivalent fuel supplies six million BTUs of heat. Fuel gas quantities are usually calculated as equivalent fuel barrels in economic calculations for refinery operations.
<b>escalation monitor</b>	A batch process that monitors pending requests or activities, and restarts or forwards them to the next step or user after they have been inactive for a specified amount of time.
<b>ESR</b>	Einzahlungsschein mit Referenznummer. A pay slip with a reference number.
<b>event rule</b>	[In EnterpriseOne] A logic statement that instructs the system to perform one or more operations that are based on an activity that can occur in a specific application, such as entering a form or exiting a field.
<b>exit bar</b>	[In EnterpriseOne] The tall pane with icons in the left portion of many EnterpriseOne program windows.
<b>facility</b>	An entity within a business for which you want to track costs. For example, a facility might be a warehouse location, job, project, work center, or branch/plant. Sometimes referred to as a business unit.
<b>fast path</b>	[In EnterpriseOne] A command prompt that allows the user to move quickly among menus and applications by using specific commands.
<b>file handle</b>	A temporary reference (typically a number) that is assigned to a file which has been opened by the operating system and is used throughout the session to access the file.
<b>file server</b>	A computer that stores files to be accessed by other computers on the network.
<b>find/browse</b>	A type of form used to:  Search, view, and select multiple records in a detail area.  Delete records.  Exit to another form.  Serve as an entry point for most applications.

<b>firm planned order (FPO)</b>	A work order that has reached a user defined status. When this status is entered in the processing options for the various manufacturing programs, messages for those orders are not exploded to the components.
<b>fiscal date pattern</b>	A representation of the beginning date for the fiscal year and the ending date for each period in that year.
<b>fix/inspect</b>	A type of form used to view, add, or modify existing records. A fix/inspect form has no detail area.
<b>fixed quantity</b>	A term that indicates the bill of material relationship between a parent item and its components or ingredients. When a bill of material component has a fixed quantity relationship to its parent, the amount of the component does not change when the software calculates parts list requirements for different work order quantities. Contrast with variable quantity.
<b>flexible account numbers</b>	The format of account numbers for journal entries. The format that you set up must be the three segments:  Business unit.  Object.  Subsidiary.
<b>form design aid (FDA)</b>	The EnterpriseOne GUI development tool for building interactive applications and forms.
<b>form exit</b>	[In EnterpriseOne] An option that is available as a button on the Form Exit bar or as a selection in the Form menu. It allows users to open an interconnected form.
<b>form interconnection</b>	Allows one form to access and pass data to another form. Form interconnections can be attached to any event; however, they are normally used when a button is clicked.
<b>form type</b>	The following form types are available in EnterpriseOne:  Find/browse.  Fix/inspect.  Header detail.  Headerless detail.  Message.  Parent/child.  Search/select.
<b>form-to-form call</b>	A request by a form for data or functionality from one of the connected forms.
<b>framework</b>	[In object-oriented systems] A set of object classes that provide a collection of related functions for a user or piece of software.
<b>frozen cost</b>	The cost of an item, operation, or process after the frozen update program is run; used by the Manufacturing Accounting system.
<b>frozen update program</b>	A program that freezes the current simulated costs, thereby finalizing them for use by the Manufacturing Accounting system.

<b>globally unique identifier (GUI)</b>	A 16-byte code in the Component Object Model that identifies an interface to an object across all computers and networks.
<b>handle</b>	[In programming] A pointer that contains the address of another pointer, which, in turn, contains the address of the desired object.
<b>hard commitment</b>	The number of items that are reserved for a sales order, work order, or both, from a specific location, lot, or both.
<b>hard error</b>	An error that cannot be corrected by a given error detection and correction system.
<b>header</b>	Information at the beginning of a table or form. Header information is used to identify or provide control information for the group of records that follows.
<b>header information</b>	Information that pertains to the entire order.
<b>hover help</b>	A help function that provides contextual information or instructions when a cursor moves over a particular part of the interface element for a predefined amount of time.
<b>ICMS</b>	Imposto sobre circulação de mercadoria e serviços. In Brazil, a state tax that is applied to the movement of merchandise and some services.
<b>ICMS Substituto</b>	Imposto sobre circulação de mercadoria e serviços substituto. In Brazil, the ICMS tax that is charged on interstate transactions, or on special products and clients.
<b>ICMS Substituto-Markup</b>	See imposto sobre circulação de mercadoria e serviços substituto-markup.
<b>imposto de renda (IR)</b>	Brazilian income tax.
<b>imposto sobre produtos industrializados</b>	In Brazil, a federal tax that applies to manufactured goods (domestic and imported).
<b>imposto sobre services (ISS)</b>	In Brazil, tax on services.
<b>inbound document</b>	A document that is received from a trading partner using Electronic Data Interface (EDI). This document is also referred to as an inbound transaction.
<b>indented tracing</b>	Tracking all lot numbers of intermediates and ingredients that are consumed in the manufacture of a given lot of product, down through all levels of the bill of material, recipe, or formula.
<b>indexed allocations</b>	A procedure that allocates or distributes expenses, budgets, adjustments, and so on, among business units, based on a fixed percentage.
<b>indirect measurement</b>	Determining the quantity on-hand by:  Measuring the storage vessels and calculating the content's balance quantity.  or  Theoretically calculating consumption of ingredients and deducting them from the on-hand balance.

<b>indirect usage</b>	Determining what should have been used by multiplying receipt quantity of the parent times the quantity per statement in the formula, recipe, or bill of material. This transaction typically affects both consumption on schedule as well as issue from on-hand balances.
<b>in-process rework</b>	Recycling a semi processed product that does not meet acceptable standards. Further processing takes the product out of a given operation and sends it back to the beginning of that operation or a previous operation (for example, unreacted materials).   Rework that is detected prior to receipt of finished goods and corrected during the same schedule run.
<b>INPS withholding tax</b>	Instituto Nazionale di Previdenza Sociale withholding tax. In Italy, a 12% social security withholding tax that is imposed on payments to certain types of contractors. This tax is paid directly to the Italian social security office.
<b>inscrição estadual</b>	ICMS tax ID. In Brazil, the state tax ID.
<b>inscrição municipal</b>	ISS tax ID. In Brazil, the municipal tax ID.
<b>integrated toolset</b>	Unique to EnterpriseOne is an industrial-strength toolset that is embedded in the already comprehensive business applications. This toolset is the same toolset that is used by PeopleSoft to build EnterpriseOne interactive and batch applications. Much more than a development environment, however, the EnterpriseOne integrated toolset handles reporting and other batch processes, change management, and basic data warehousing facilities.
<b>integrity test</b>	A process that is used to supplement a company's internal balancing procedures by locating and reporting balancing problems and data inconsistencies.
<b>interbranch sales order</b>	A sales order that is used for transactions between branch/plants other than the selling branch/plant.
<b>Interoperability</b>	The ability of different computer systems, networks, operating systems, and applications to work together and share information.
<b>inventory pricing rule</b>	A discount method that is used for purchases from suppliers and sales to customers. The method is based on effectivity dates, up-to quantities, and a factor by which you can mark up or discount the price or cost.
<b>inventory turn</b>	The number of times that the inventory cycles, or turns over, during the year. A frequently used method to compute inventory turnover is to divide the annual costs of sales by the average inventory level.
<b>invoice</b>	An itemized list of goods that are shipped or services that are rendered, stating quantities, prices, fees, shipping charges, and so on. Companies often have their invoices mailed to a different address than where they ship products. In such cases, the bill-to address differs from the ship-to address.
<b>IP</b>	See imposto sobre produtos industrializados.
<b>IR</b>	See imposto de renda.
<b>IServer Service</b>	Developed by PeopleSoft, this Internet server service resides on the Web server and is used to speed up delivery of the Java class files from the database to the client.

<b>ISS</b>	See imposto sobre servicios.
<b>jargon</b>	An alternate data dictionary item description that EnterpriseOne or PeopleSoft World displays, based on the product code of the current object.
<b>java application server</b>	A component-based server that resides in the middle-tier of a server-centric architecture and provides middleware services for security and state maintenance, along with data access and persistence.
<b>JDBNET</b>	A database driver that allows heterogeneous servers to access each other's data.
<b>jde.ini</b>	A PeopleSoft file (or member for AS/400) that provides the runtime settings that are required for EnterpriseOne initialization. Specific versions of the file or member must reside on every machine that is running EnterpriseOne, including workstations and servers.
<b>JDE.LOG</b>	The main diagnostic log file of EnterpriseOne. Always located in the root directory on the primary drive. Contains status and error messages from the startup and operation of EnterpriseOne.
<b>JDEBASE Database Middleware</b>	PeopleSoft proprietary database middleware package that provides two primary benefits:       1. Platform-independent APIs for multidatabase access. These APIs are used in two ways:      a. By the interactive and batch engines to dynamically generate platform-specific SQL, depending on the data source request.     b. As open APIs for advanced C business function writing. These APIs are then used by the engines to dynamically generate platform-specific SQL.           2. Client-to-server and server-to-server database access. To accomplish this access, EnterpriseOne is integrated with a variety of third-party database drivers, such as Client Access 400 and open database connectivity (ODBC).
<b>JDECallObject</b>	An application programming interface that is used by business functions to invoke other business functions.
<b>JDEIPC</b>	Communications programming tools that are used by server code to regulate access to the same data in multiprocess environments, communicate and coordinate between processes, and create new processes.
<b>JDENET</b>	PeopleSoft proprietary middleware software. JDENET is a messaging software package.
<b>JDENET communications middleware</b>	PeopleSoft proprietary communications middleware package for EnterpriseOne. It is a peer-to-peer, message-based, socket-based, multiprocess communications middleware solution. It handles client-to-server and server-to-server communications for all EnterpriseOne supported platforms.
<b>just in time installation (JITI)</b>	EnterpriseOne's method of dynamically replicating objects from the central object location to a workstation.
<b>just in time replication (JITR)</b>	EnterpriseOne's method of replicating data to individual workstations. EnterpriseOne replicates new records (inserts) only at the time that the user needs the data. Changes, deletes, and updates must be replicated using Pull Replication.

<b>Kagami</b>	In Japan, summarized invoices that are created monthly (in most cases) to reduce the number of payment transactions.
<b>latitude</b>	The X coordinate of the location of an item in the warehouse. The system can use latitude, longitude, and height when suggesting locations for putaway, replenishment, and picking.
<b>laytime (or layhours)</b>	The amount of time that is allotted to a tanker at berth to complete loading or discharging cargo. This time is usually expressed in running hours, and is fixed by prior agreement between the vessel owner and the company that is chartering the vessel. Laytime is stipulated in the charter, which states exactly the total of number of hours that are granted at both loading and unloading ports, and indicates whether such time is reversible. A statement of “Seventy-Two Hours, Reversible” means that a total of 72 hours is granted overall at both ports, and any time saved at one port can be applied as a credit at the other port.   For example, if the vessel uses only 32 hours instead of 36 hours to load cargo, it can apply an additional four hours to the 36 hours allotted at the discharge port. Such considerations are important for purposes of computing demurrage.
<b>leading zeros</b>	A series of zeros that certain facilities in PeopleSoft systems place in front of a value that is entered. This situation normally occurs when you enter a value that is smaller than the specified length of the field. For example, if you enter 4567 in a field that accommodates eight numbers, the facility places four zeros in front of the four numbers that you enter. The result appears as 00004567.
<b>ledger type</b>	A code that designates a ledger which is used by the system for a particular purpose. For example, all transactions are recorded in the AA (actual amounts) ledger type in their domestic currency. The same transactions can also be stored in the CA (foreign currency) ledger type.
<b>level break</b>	The position in a report or text where a group of similar types of information ends and another one begins.
<b>libro IVA</b>	Monthly VAT report. In Italy, the term for the report that contains the detail of invoices and vouchers that were registered during each month.
<b>line of business</b>	A description of the nature of a company’s work; also a tool to control the relationship with that customer, including product pricing.
<b>linked service type</b>	A service type that is associated with a primary service type. Linked service types can be cancelled, and the maintenance tasks are performed when the primary service type to which they are linked comes due. You can specify whether the system generates work orders for linked service types, as well as the status that the system assigns to work orders that have already been generated. Sometimes referred to as associated service types. See also primary service type and service type.
<b>livro razao</b>	In Brazil, a general ledger report.
<b>load balancing</b>	The act of distributing the number of processes proportionally to all servers in a group to maximize overall performance.
<b>location workbench</b>	During the Installation Workbench process, Location Workbench copies all locations that are defined in the installation plan from the Location Master table in the Planner data source to the System data source.

<b>log files</b>	Files that track operations for a process or application. Reviewing log files is helpful for troubleshooting problems. The file extension for log files is .LOG.
<b>logic data source</b>	Any code that provides data during runtime.
<b>logical compartment</b>	One of two ways that is identified in the transportation constants to display compartments on vehicles. Logical display numbers the compartments sequentially.   For example, if two vehicles are on a trip and each vehicle has three compartments, the logical display is 1,2,3,4,5,6.
<b>logical file</b>	A set of keys or indices that is used for direct access or ordered access to the records in a physical file. Several logical files can have different accesses to a physical.
<b>logical shelf</b>	A logical, not physical, location for inventory that is used to track inventory transactions in loan/borrow, or exchange agreements with other companies. See also logical warehouse.
<b>logical warehouse</b>	Not a physical warehouse containing actual inventory, but a means for storing and tracking information for inventory transactions in loan/borrow, or exchange agreements with other companies.
<b>longitude</b>	The Y coordinate of the location of an item in the warehouse. The system can use latitude, longitude, and height when suggesting locations for putaway, replenishment, and picking.
<b>LSV</b>	Lastschriftverfahren. A Swiss auto debit format that is required by Telekurs (Payserv).
<b>mail merge</b>	A mass-mail facility that takes names, addresses, and (sometimes) pertinent facts about recipients and merges the information into a form letter or a similarly basic document.
<b>mailmerge workbench</b>	[In EnterpriseOne] An application that merges Microsoft Word 6.0 (or higher) word-processing documents with EnterpriseOne records to automatically print business documents.
<b>main fuels</b>	Usually refers to bulk fuel products, but sometimes includes packaged products.
<b>maintenance loop</b>	See maintenance route.
<b>maintenance route</b>	A method of performing PMs for multiple pieces of equipment from a single preventive maintenance work order. A maintenance route includes pieces of equipment that share one or more identical maintenance tasks which can be performed at the same time for each piece of equipment. Sometimes referred to as maintenance loop.
<b>maintenance work order</b>	In PeopleSoft EnterpriseOne systems, a term that is used to distinguish work orders created for the performance of equipment and plant maintenance from other work orders, such as manufacturing work orders, utility work orders, and engineering change orders.

<b>manufacturing and distribution planning</b>	Planning that includes resource and capacity planning, and material planning operations. Resource and capacity planning allows you to prepare a feasible production schedule that reflects your demand forecasts and production capability. Material Planning Operations provides a short-range plan to cover material requirements that are needed to make a product.
<b>mapping</b>	A set of instructions that describes how one data structure passes data to another.
<b>master business function</b>	An interactive master file that serves as a central location for adding, changing, and updating information in a database.
<b>master business function</b>	A central system location for standard business rules about entering documents, such as vouchers, invoices, and journal entries. Master business functions ensure uniform processing according to guidelines that you establish.
<b>master table</b>	A database table that is used to store data and information that is permanent and necessary to the system's operation. Master tables might contain data such as paid tax amounts, supplier names, addresses, employee information, and job information.
<b>matching document</b>	A document that is associated with an original document to complete or change a transaction. For example, a receipt is the matching document of an invoice.
<b>media object</b>	An electronic or digital representation of an object.
<b>media storage objects</b>	Files that use one of the following naming conventions that are not organized into table format: Gxxx, xxxGT, or GTxxx.
<b>memory violation</b>	An error that occurs as the result of a memory leak.
<b>menu selection</b>	An option on a menu that initiates a software function directly.
<b>message center</b>	A central location for sending and receiving all EnterpriseOne messages (system- and user-generated), regardless of the originating application or user.
<b>messaging application programming interface (MAPI)</b>	An architecture that defines the components of a messaging system and how they behave. It also defines the interface between the messaging system and the components.
<b>metal content</b>	A series of properties of a blended product that help to determine its suitability for a prescribed purpose.
<b>metals management</b>	The process of maintaining information about the location and status of durable product containers such as liquid petroleum gas (LPG) cylinders.
<b>mobile inventory</b>	Inventory that is transferred from a depot to a barge or truck for milk-run deliveries.
<b>modal</b>	A restrictive or limiting interaction that is created by a given condition of operation. Modal often describes a secondary window that restricts a user's interaction with other windows. A secondary window can be modal with respect to its primary window or to the entire system. A modal dialog box must be closed by the user before the application continues.

<b>model work order</b>	For scheduled preventive maintenance or for a condition-based alert, a model work order functions as a template for the creation of other work orders. You can assign model work orders to service types and condition-based alerts. When the service type comes due or the alert is generated, the system automatically generates a work order that is based on information from the model work order.
<b>modeless</b>	Not restricting or limiting interaction. Modeless often describes a secondary window that does not restrict a user's interaction with other windows. A modeless dialog box stays on the screen and is available for use at any time, but also permits other user activities.
<b>multiple stocking locations</b>	Authorized storage locations for the same item number at locations, in addition to the primary stocking location.
<b>multitier architecture</b>	A client/server architecture that allows multiple levels of processing. A tier defines the number of computers that can be used to complete some defined task.
<b>named event rules (NER)</b>	Also called business function event rules. Encapsulated, re-usable business logic that is created by using event rules, rather than C programming.
<b>national language support (NLS)</b>	Mechanisms that are provided to facilitate internationalization of both system and application user interfaces.
<b>natureza da operação</b>	Transaction nature. In Brazil, a code that classifies the type of commercial transaction to conform to the fiscal legislation.
<b>negative pay item</b>	An entry in an account that indicates a prepayment. For example, you might prepay a supplier before goods are sent or prepay an employee's forecasted expenses for a business trip. The system stores these pending entries, assigning them a minus quantity as debit amounts in a designated expense account. After the prepaid goods are received or the employee submits an expense report, entering the actual voucher clears all of the negative pay items by processing them as regular pay items. Note that a negative pay item can also result from entering a debit memo (A/P) or a credit memo (A/R).
<b>net added cost</b>	The cost to manufacture an item at the current level in the bill of material. Thus, for manufactured parts, the net added cost includes labor, outside operations, and cost extras applicable to this level in the bill of material, but not materials (lower-level items). For purchased parts, the net added cost also includes the cost of materials.
<b>next status</b>	The next step in the payment process for payment control groups. The next status can be either WRT (write) or UPD (update).
<b>node</b>	A termination point for two or more communications links. A node can serve as the control location for forwarding data among the elements of a network or multiple networks, as well as performing other networking and, in some cases, local processing.
<b>non-inventory items</b>	See non-stock items.
<b>non-list price</b>	A price for bulk products that is determined by its own algorithms, such as a rolling average or commodity price plus.
<b>non-prime product</b>	A manufactured product with revenue potential that is less than the product planned for, or scheduled to be produced.

<b>non-stock items</b>	Items that the system does not account for as part of the inventory. For example, office supplies, or packaging materials can be non-stock items.
<b>nota fiscal</b>	In Brazil, a legal document that must accompany all commercial transactions.
<b>nota fiscal fatura</b>	In Brazil, a nota fiscal and invoice information.
<b>notula</b>	In Italy, the process whereby a business does not recognize value added tax until the payment of a voucher.
<b>object configuration manager (OCM)</b>	EnterpriseOne's object request broker and the control center for the runtime environment. It keeps track of the runtime locations for business functions, data, and batch applications. When one of these objects is called, the Object Configuration Manager directs access to it by using defaults and overrides for a given environment and user.
<b>object embedding</b>	When an object is embedded in another document, an association is maintained between the object and the application that created it; however, any changes made to the object are also only kept in the compound document. See also object linking.
<b>object librarian</b>	A repository of all versions, applications, and business functions that are re-usable in building applications.
<b>object linking</b>	When an object is linked to another document, a reference is created with the file in which the object is stored, as well as with the application that created it. When the object is modified, either from the compound document or directly through the file in which it is saved, the change is reflected in that application as well as anywhere it has been linked. See also object embedding.
<b>object linking and embedding (OLE)</b>	A technology for transferring and sharing information among applications by allowing the integration of objects from diverse applications, such as graphics, charts, spreadsheets, text, or an audio clip from a sound program. OLE is a compound document standard that was developed by Microsoft Corporation. It enables you to create objects with one application, and then link or embed them in a second application. Embedded objects retain their original format and links to the application that created them. See also object embedding, object linking.
<b>object management workbench (OMW)</b>	The change management system that is used for EnterpriseOne development.
<b>object-based technology (OBT)</b>	A technology that supports some of the main principles of object-oriented technology:  Classes.  Polymorphism.I  Inheritance.  Encapsulation.

<b>object-oriented technology (OOT)</b>	Brings software development past procedural programming into a world of reusable programming that simplifies development of applications. Object orientation is based on the following principles:  Classes.  Polymorphism.I  Inheritance.  Encapsulation.
<b>offsetting account</b>	An account that reduces the amount of another account to provide a net balance. For example, a credit of 200 to a cash account might have an offsetting entry of 200 to an A/P Trade (liability) account.
<b>open database connectivity (ODBC)</b>	Defines a standard interface for different technologies to process data between applications and different data sources. The ODBC interface comprises set of function calls, methods of connectivity, and representation of data types that define access to data sources.
<b>open systems interconnection (OSI)</b>	The OSI model was developed by the International Standards Organization (ISO) in the early 1980s. It defines protocols and standards for the interconnection of computers and network equipment.
<b>order detail line</b>	A part of an order that contains transaction information about a service or item being purchased or sold, such as quantity, cost, price, and so on.
<b>order hold</b>	A flag that stops the processing of an order because it has exceeded the credit or budget limit, or has another problem.
<b>order-based pricing</b>	Pricing strategy that grants reductions in price to a customer. It is based upon the contents and relative size (volume or value) of the order as a whole.
<b>outbound document</b>	A document that is sent to a trading partner using EDI. This term is also referred to as an outbound transaction.
<b>outturn</b>	The quantity of oil that is actually received into a buyer's storage tanks when a vessel is unloaded. For various reasons (vaporization, clingage to vessel tank walls, and so on), the amount of a product pumped into shore tankage at unloading is often less than the quantity originally loaded onto the vessel, as certified by the Bill of Lading. Under a delivered or CIF outturn transaction, the buyer pays only for the barrels actually "turned out" by the vessel into storage.  When a buyer is paying CIF Bill of Lading figures, a loss of 0.5% of total cargo volume is considered normal. Losses in excess of 0.5%, however, are either chargeable to the seller or are covered by specialized insurance that covers partial, as well as total, loss of the cargo.
<b>overhead</b>	In the distillation process, that portion of the charge that leaves the top of the distillation column as vapor. This definition is strictly as it relates to ECS.
<b>override conversion method</b>	A method of calculating exchange rates that is set up between two specific currencies. For those specific currencies, this method overrides the conversion method in General Accounting Constants and does not allow inverse rates to be used when calculating currency amounts.

<b>package / package build</b>	A collection of software that is grouped into a single entity for modular installation. EnterpriseOne objects are installed to workstations in packages from the deployment server. A package can be compared to a bill of material or kit that indicates the necessary objects for that workstation and where the installation program can find them on the deployment server. It is a point-in-time “snapshot” of the central objects on the deployment server.
<b>package location</b>	The directory structure location for the package and its set of replicated objects. This location is usually \\deployment server\release\path_code\package\ package name. The replicated objects for the package are placed in the subdirectories under this path. This location is also where the package is built or stored.
<b>package workbench</b>	During the Installation Workbench process, Package Workbench transfers the package information tables from the Planner data source to the System - release number data source. It also updates the Package Plan detail record to reflect completion.
<b>packaged products</b>	Products that, by their nature, must be delivered to the customer in containers which are suitable for discrete consumption or resale.
<b>pane/panel</b>	A resizable subarea of a window that contains options, components, or other related information.
<b>paper clip</b>	An icon that is used to indicate that a media object is attached to a form or record.
<b>parent/child form</b>	A type of form that presents parent/child relationships in an application on one form:   The left portion of the form presents a tree view that displays a visual representation of a parent/child relationship.   The right portion of the form displays a detail area in browse mode. The detail area displays the records for the child item in the tree.   The parent/child form supports drag and drop functionality.
<b>parent/child relationship</b>	See parent/component relationship.
<b>parent/component relationship</b>	1. In Capital Asset Management, the hierarchical relationship of a parent piece of equipment to its components. For example, a manufacturing line could be a parent and the machinery on the line could be components of the line. In addition, each piece of machinery could be a parent of still more components.   2. In Product Data Management, a hierarchical relationship of the components and subassemblies of a parent item to that parent item. For example, an automobile is a parent item; its components and subassemblies include: engine, frame, seats, and windows.   Sometimes referred to as parent/child relationship.
<b>partita IVA</b>	In Italy, a company fiscal identification number.
<b>pass-through</b>	A process where data is accepted from a source and forwarded directly to a target without the system or application performing any data conversion, validation, and so on.
<b>pay on consumption</b>	The method of postponing financial liability for component materials until you issue that material to its consuming work order or rate schedule.

<b>payment group</b>	A system-generated group of payments with similar information, such as a bank account. The system processes all of the payments in a payment group at the same time.
<b>PeopleSoft database</b>	See JDEBASE Database Middleware.
<b>performance tuning</b>	The adjustments that are made for a more efficient, reliable, and fast program.
<b>persistent object</b>	An object that continues to exist and retains its data beyond the duration of the process that creates it.
<b>pervasive device</b>	A type of intelligent and portable device that provides a user with the ability to receive and gather information anytime, from anywhere.
<b>planning family</b>	A means of grouping end items that have similarity of design or manufacture.
<b>plug-in</b>	A small program that plugs into a larger application to provide added functionality or enhance the main application.
<b>polymorphism</b>	A principle of object-oriented technology in which a single mnemonic name can be used to perform similar operations on software objects of different types.
<b>portal</b>	A Web site or service that is a starting point and frequent gateway to a broad array of on-line resources and services.
<b>Postfinance</b>	A subsidiary of the Swiss postal service. Postfinance provides some banking services.
<b>potency</b>	Identifies the percent of an item in a given solution. For example, you can use an 80% potent solution in a work order that calls for 100% potent solution, but you would use 25% more, in terms of quantity, to meet the requirement ( $100 / 80 = 1.25$ ).
<b>preference profile</b>	The ability to define default values for specified fields for a user defined hierarchy of items, item groups, customers, and customer groups. In Quality Management setup, this method links test and specification testing criteria to specific items, item groups, customers, or customer groups.
<b>preflush</b>	A work order inventory technique in which you deduct (relieve) materials from inventory when the parts list is attached to the work order or rate schedule.
<b>preventive maintenance cycle</b>	The sequence of events that make up a preventive maintenance task, from its definition to its completion. Because most preventive maintenance tasks are commonly performed at scheduled intervals, parts of the preventive maintenance cycle repeat, based on those intervals.
<b>preventive maintenance schedule</b>	The combination of service types that apply to a specific piece of equipment, as well as the intervals at which each service type is scheduled to be performed.
<b>primary service type</b>	A service type to which you can link related service types. For example, for a particular piece of equipment, you might set up a primary service type for a 1000-hour inspection and a linked service type for a 500-hour inspection. The 1000-hour inspection includes all of the tasks performed at 500 hours. When a primary service type is scheduled to be performed, the system schedules the linked service type. See also linked service type.

<b>pristine environment</b>	An EnterpriseOne environment that is used to test unaltered objects with PeopleSoft demonstration data or for training classes. You must have this environment so you can compare pristine objects that you modify.
<b>processing option</b>	A data structure that allows users to supply parameters that regulate the execution of a batch program or report.
<b>product data management (PDM)</b>	In PeopleSoft EnterpriseOne software, the system that enables a business to organize and maintain information about each item which it manufactures. Features of this system, such as bills of material, work centers, and routings, define the relationships among parents and components, and how they can be combined to manufacture an item. PDM also provides data for other manufacturing systems including Manufacturing Accounting, Shop Floor Management, and Manufacturing and Distribution Planning.
<b>product line</b>	A group of products with similarity in manufacturing procedures, marketing characteristics, or specifications that allow them to be aggregated for planning; marketing; and, occasionally, costing.
<b>product/process definition</b>	A combination of bill of material (recipe, formula, or both) and routing (process list). Organized into tasks with a statement of required consumed resources and produced resources.
<b>production environment</b>	An EnterpriseOne environment in which users operate EnterpriseOne software.
<b>program temporary fix (PTF)</b>	A representation of changes to PeopleSoft software that your organization receives on magnetic tapes or diskettes.
<b>project</b>	[In EnterpriseOne] A virtual container for objects being developed in Object Management Workbench.
<b>projected cost</b>	The target expenditure in added value for material, labor, and so on, during manufacture. See also standard cost.
<b>promotion path</b>	The designated path for advancing objects or projects in a workflow.
<b>protocollo</b>	See registration number.
<b>PST</b>	Provincial sales tax. A tax that is assessed by individual provinces in Canada.
<b>published table</b>	Also called a “Master” table, this is the central copy to be replicated to other machines and resides on the “publisher” machine. The Data Replication Publisher Table (F98DRPUB) identifies all of the published tables and their associated publishers in the enterprise.
<b>publisher</b>	The server that is responsible for the published table. The Data Replication Publisher Table (F98DRPUB) identifies all of the published tables and their associated publishers in the enterprise.
<b>pull replication</b>	One of the EnterpriseOne methods for replicating data to individual workstations. Such machines are set up as pull subscribers that use EnterpriseOne’s data replication tools. The only time that pull subscribers are notified of changes, updates, and deletions is when they request such information. The request is in the form of a message that is sent, usually at startup, from the pull subscriber to the server machine that stores the Data Replication Pending Change Notification table (F98DRPCN).

<b>query by example (QBE)</b>	Located at the top of a detail area, this area is used to search for data to display in the detail area.
<b>rate scheduling</b>	A method of scheduling product or manufacturing families, or both.  Also a technique to determine run times and quantities of each item within the family to produce enough of each individual product to satisfy demand until the family can be scheduled again.
<b>rate type</b>	For currency exchange transactions, the rate type distinguishes different types of exchange rates. For example, you can use both period average and period-end rates, distinguishing them by rate type.
<b>real-time</b>	Pertaining to information processing that returns a result so rapidly that the interaction appears to be instantaneous.
<b>receipt routing</b>	A series of steps that is used to track and move items within the receipt process. The steps might include in-transit, dock, staging area, inspection, and stock.
<b>referential integrity</b>	Ensures that a parent record cannot be deleted from the database when a child record for exists.
<b>regenerable</b>	Source code for EnterpriseOne business functions can be regenerated from specifications (business function names). Regeneration occurs whenever an application is recompiled, either for a new platform or when new functionality is added.
<b>register types and classes</b>	In Italian VAT Summary Reporting, the classification of VAT transactions.
<b>relationship</b>	Links tables together and facilitates joining business views for use in an application or report. Relationships that are created are based on indexes.
<b>relevé d'identité bancaire (RIB)</b>	In France, the term that indicates the bank transit code, account number, and check digit that are used to validate the bank transit code and account number. The bank transit code consists of the bank code and agency code. The account number is alphanumeric and can be as many as 11 characters. PeopleSoft supplies a validation routine to ensure RIB key correctness.
<b>remessa</b>	In Brazil, the remit process for A/R.
<b>render</b>	To include external data in displayed content through a linking mechanism.
<b>repasse</b>	In Brazil, a discount of the ICMS tax for interstate transactions. It is the adjustment between the interstate and the intrastate ICMS tax rates.
<b>replenishment point</b>	The location on or near the production line where additional components or subassemblies are to be delivered.
<b>replication server</b>	A server that is responsible for replicating central objects to client machines.
<b>report design aid (RDA)</b>	The EnterpriseOne GUI tool for operating, modifying, and copying report batch applications.
<b>repost</b>	In Sales, the process of clearing all commitments from locations and restoring commitments, based on quantities from the Sales Order Detail table (F4211).
<b>resident</b>	Pertaining to computer programs or data while they remain on a particular storage device.

<b>retorno</b>	In Brazil, the receipt process for A/R.
<b>RIB</b>	See relevé d'identité bancaire.
<b>ricevute bancarie (RiBa)</b>	In Italy, the term for accounts receivable drafts.
<b>riepilogo IVA</b>	Summary VAT monthly report. In Italy, the term for the report that shows the total amount of VAT credit and debit.
<b>ritenuta d'acconto</b>	In Italy, the term for standard withholding tax.
<b>rollback</b>	[In database management] A feature or command that undoes changes in database transactions of one or more records.
<b>rollup</b>	See cost rollup.
<b>row exit</b>	[In EnterpriseOne] An application shortcut, available as a button on the Row Exit bar or as a menu selection, that allows users to open a form that is related to the highlighted grid record.
<b>runtime</b>	The period of time when a program or process is running.
<b>SAD</b>	The German name for a Swiss payment format that is accepted by Postfinance.
<b>SAR</b>	See software action request.
<b>scalability</b>	The ability of software, architecture, hardware, or a network to support software as it grows in size or resource requirements.
<b>scripts</b>	A collection of SQL statements that perform a specific task.
<b>scrub</b>	To remove unnecessary or unwanted characters from a string.
<b>search/select</b>	A type of form that is used to search for a value and return it to the calling field.
<b>selection</b>	Found on PeopleSoft menus, selections represent functions that you can access from a menu. To make a selection, type the associated number in the Selection field and press Enter.
<b>serialize</b>	To convert a software object into a stream of bytes to store on a disk or transfer across a network.
<b>server map</b>	The server view of the object configuration mapping.
<b>server workbench</b>	During the Installation Workbench process, Server Workbench copies the server configuration files from the Planner data source to the System release number data source. It also updates the Server Plan detail record to reflect completion.
<b>service interval</b>	The frequency at which a service type is to be performed. Service intervals can be based on dates, periods, or statistical units that are user defined. Examples of statistical units are hours, miles, and fuel consumption.
<b>service type</b>	An individual preventive maintenance task or procedure, such as an inspection, lubrication, or overhaul. Service types can apply to a specific piece of equipment or to a class of equipment. You can specify that service types come due based on a predetermined service interval, or whenever the task that is represented by the service type becomes necessary.

<b>servlet</b>	A [small] program that extends the functionality of a Web server by generating dynamic content and interacting with Web clients by using a request-response paradigm.
<b>share path</b>	The network node under which one or more servers or objects reside.
<b>shop floor management</b>	A system that uses data from multiple system codes to help develop, execute, and manage work orders and rate schedules in the enterprise.
<b>silent mode</b>	A method for installing or running a program that does not require any user intervention.
<b>silent post</b>	A type of post that occurs in the background without the knowledge of the user.
<b>simulated cost</b>	After a cost rollup, the cost of an item, operation, or process according to the current cost scenario. This cost can be finalized by running the frozen update program. You can create simulated costs for a number of cost methods—for example, standard, future, and simulated current costs. See also cost rollup.
<b>single-byte character set (SBCS)</b>	An encoding scheme in which each alphabetic character is represented by one byte. Most Western languages, such as English, can be represented by using a single-byte character set.
<b>single-level tracking</b>	Finding all immediate parents where a specific lot has been used (consumed).
<b>single-voyage (spot) charter</b>	An agreement for a single voyage between two ports. The payment is made on the basis of tons of product delivered. The owner of the vessel is responsible for all expenses.
<b>slimer</b>	A script that changes data in a table directly without going through a regular database interface.
<b>smart field</b>	A data dictionary item with an attached business function for use in the Report Design Aid application.
<b>SOC</b>	The Italian term for a Swiss payment format that is accepted by Postfinance.
<b>soft commitment</b>	The number of items that is reserved for sales orders or work orders in the primary units of measure.
<b>soft error</b>	An error from which an operating system or program is able to recover.
<b>software action request (SAR)</b>	An entry in the AS/400 database that is used for requesting modifications to PeopleSoft software.
<b>SOG</b>	The French term for a Swiss payment format that is accepted by Postfinance.
<b>source directory</b>	The path code to the business function source files belonging to the shared library that is created on the enterprise server.
<b>special period/year</b>	The date that determines the source balances for an allocation.

<b>specification merge</b>	The Specification merge is comprised of three merges:   Object Librarian merge (via the Object Management Workbench).   Versions List merge.   Central Objects merge.   The merges blend customer modifications with data that accompanies a new release.
<b>specification table merge workbench</b>	During the Installation Workbench process, Specification Table Merge Workbench runs the batch applications that update the specification tables.
<b>specifications</b>	A complete description of an EnterpriseOne object. Each object has its own specification, or name, which is used to build applications.
<b>spot charter</b>	See single-voyage charter.
<b>spot rates</b>	An exchange rate that is entered at the transaction level. Spot rates are not used on transactions between two EMU member currencies because exchange rates are irrevocably fixed to the euro.
<b>stamp tax</b>	In Japan, a tax that is imposed on drafts payable, receipts over 30000 Japanese yen, and all contracts. The party that issues any of the above documents is responsible for this tax.
<b>standalone</b>	Operating or capable of operating independently of certain other components of a computer system.
<b>standard cost</b>	The expected, or target cost of an item, operation, or process. Standard costs represent only one cost method in the Product Costing system. You can also calculate, for example, future costs or current costs. However, the Manufacturing Accounting system uses only standard frozen costs.
<b>standard costing</b>	A costing method that uses cost units that are determined before production. For management control purposes, the system compares standard costs to actual costs and computes variances.
<b>subprocess</b>	A process that is triggered by and is part of a larger process, and that generally consists of activities.
<b>subscriber table</b>	The Subscriber table (F98DRSUB), which is stored on the Publisher Server with the Data Replication Publisher table (F98DRPUB), that identifies all of the subscriber machines for each published table.
<b>summary</b>	The presentation of data or information in a cumulative or totaled manner in which most of the details have been removed. Many systems offer forms and reports that summarize information which is stored in certain tables. Contrast with detail.
<b>super backflush</b>	To create backflush transactions for material, labor, or both, against a work order at predefined pay points in the routing. By doing so, you can relieve inventory and account for labor amounts at strategic points throughout the manufacturing process.
<b>supersession</b>	Specification that a new product is replacing an active product on a specified effective date.

<b>supplemental data</b>	Additional types of data for customers and suppliers. You can enter supplemental data for information such as notes, comments, plans, or other information that you want in a customer or supplier record. The system maintains this data in generic databases, separate from the standard master tables (Customer Master, Supplier Master, and Address Book Master).
<b>supplying location</b>	The location from which inventory is transferred once quantities of the item on the production line have been depleted. In kanban processing, the supplying location is the inventory location from which materials are transferred to the consuming location when the containers are replenished.
<b>system code</b>	A numeric or alphanumeric designation that identifies a specific system in EnterpriseOne software.
<b>system function</b>	[In EnterpriseOne] A named set of pre-packaged, re-usable instructions that can be called from event rules.
<b>table access management (TAM)</b>	The EnterpriseOne component that handles the storage and retrieval of user defined data. TAM stores information such as data dictionary definitions; application and report specifications; event rules; table definitions; business function input parameters and library information; and data structure definitions for running applications, reports, and business functions.
<b>table conversion workbench</b>	During the Installation Workbench process, Table Conversion Workbench runs the table conversions that change the technical and application tables to the format for the new release of EnterpriseOne. It also updates the Table Conversions and Controls detail records to reflect completion.
<b>table design aid (TDA)</b>	An EnterpriseOne GUI tool for creating, modifying, copying, and printing database tables.
<b>table event rules</b>	Use table event rules to attach database triggers (or programs) that automatically run whenever an action occurs against the table. An action against a table is referred to as an event. When you create an EnterpriseOne database trigger, you must first determine which event will activate the trigger. Then, use Event Rules Design to create the trigger. Although EnterpriseOne allows event rules to be attached to application events, this functionality is application-specific. Table event rules provide embedded logic at the table level.
<b>table handle</b>	A pointer into a table that indicates a particular row.
<b>table space</b>	[In relational database management systems] An abstract collection of containers in which database objects are stored.
<b>task</b>	[In Solution Explorer and EnterpriseOne Menu] A user defined object that can initiate an activity, process, or procedure.
<b>task view</b>	A group of tasks in Solution Explorer or EnterpriseOne Menu that are arranged in a tree structure.
<b>termo de abertura</b>	In Brazil, opening terms for the transaction journal.
<b>termo de encerramento</b>	In Brazil, closing terms for the transaction journal.
<b>three-tier processing</b>	The task of entering, reviewing, approving, and posting batches of transactions.

<b>three-way voucher match</b>	The process of comparing receipt information to supplier's invoices to create vouchers. In a three-way match, you use the receipt records, the purchase order, and the invoice to create vouchers.
<b>threshold percentage</b>	In Capital Asset Management, the percentage of a service interval that you define as the trigger for maintenance to be scheduled. For example, you might set up a service type to be scheduled every 100 hours with a threshold percentage of 90 percent. When the equipment accumulates 90 hours, the system schedules the maintenance.
<b>throughput agreement</b>	A service agreement in which a business partner agrees to store and manage product for another business partner for a specified time period. The second partner actually owns the stock that is stored in the first partner's depot, although the first partner monitors the stock level; suggests replenishments; and unloads, stores, and delivers product to the partner or its customers. The first partner charges a fee for storing and managing the product.
<b>throughput reconciliation</b>	Reconcile confirmed sales figures in a given period with the measured throughput, based on the meter readings. This process is designed to catch discrepancies that are due to transactions not being entered, theft, faulty meters, or some combination of these factors. This reconciliation is the first stage. See also operational reconciliation.
<b>token</b>	[In Object Management Workbench] A flag that is associated with each object which indicates whether you can check out the object.
<b>tolerance range</b>	The amount by which the taxes that you enter manually can vary from the tax that is calculated by the system.
<b>TP monitor</b>	Transaction Processing monitor. A monitor that controls data transfer between local and remote terminals and the applications that originated them. TP monitors also protect data integrity in the distributed environment and can include programs that validate data and format terminal screens.
<b>tracing</b>	The act of researching a lot by going backward, to discover its origin.
<b>tracking</b>	The act of researching a lot by going forward, to discover where it is used.
<b>transaction set</b>	An electronic business transaction (EDI Standard document) composed of segments.
<b>transclude</b>	To include the external data in the displayed content through a linking mechanism.
<b>transfer order</b>	An order that is used to ship inventory between branch/plants within your company and to maintain an accurate on-hand inventory amount. An interbranch transfer order creates a purchase order for the shipping location and a sales order for the receiving location.
<b>translation adjustment account</b>	An optional G/L account used in currency balance restatement to record the total adjustments at a company level.
<b>translator software</b>	The software that converts data from an application table format to an EDI Standard Format, and from EDI Standard Format to application table format. The data is exchanged in an EDI Standard, such as ANSI ASC X12, EDIFACT, UCS, or WINS.

<b>tree structure</b>	A type of graphical user interface that displays objects in a hierarchy.
<b>trigger</b>	Allows you to attach default processing to a data item in the data dictionary. When that data item is used on an application or report, the trigger is invoked by an event which is associated with the data item. EnterpriseOne also has three visual assist triggers:  Calculator.  Calendar.  Search form.
<b>two-way voucher match</b>	The process of comparing purchase order detail lines to the suppliers' invoices to create vouchers. You do not record receipt information.
<b>universal batch engine (UBE)</b>	[In EnterpriseOne] A type of application that runs a noninteractive process.
<b>unnormalized</b>	Data that is a random collection of data elements with repeating record groups scattered throughout. Also see Normalized.
<b>user overrides merge</b>	The User Overrides merge adds new user override records into a customer's user override table.
<b>user-defined code (UDC)</b>	A value that a user has assigned as being a valid entry for a given or specific field.
<b>utility</b>	A small program that provides an addition to the capabilities which are provided by an operating system.
<b>variable numerator allocations</b>	A procedure that allocates or distributes expenses, budgets, adjustments, and so on, among business units, based on a variable.
<b>variable quantity</b>	A term that indicates the bill of material relationship between a parent item and its components or ingredients. When a bill of material component has a variable quantity relationship to its parent, the amount of the component changes when the software calculates parts list requirements for different work order quantities. Contrast with fixed quantity.
<b>variance</b>	1. In Product Costing and Manufacturing Accounting, the difference between the frozen standard cost, the current cost, the planned cost, and the actual cost. For example, the difference between the frozen standard cost and the current cost is an engineering variance. Frozen standard costs come from the Cost Components table, and the current costs are calculated by using the current bill of material, routing, and overhead rates.  2. In Capital Asset Management, the difference between revenue that is generated by a piece of equipment and costs that are incurred by the equipment.
<b>versions list merge</b>	The Versions List merge preserves any non-XJDE and non-ZJDE version specifications for objects that are valid in the new release as well as their processing options data.
<b>VESR</b>	Verfahren Einzahlungsschein mit Referenznummer. The processing of an ESR pay slip with reference line through accounts receivable and accounts payable.
<b>visual assist</b>	Forms that can be invoked from a control to assist the user in determining what data belongs in the control.

<b>voucher logging</b>	The process of entering vouchers without distributing amounts to specific G/L accounts. The system initially distributes the total amount of each voucher to a G/L suspense account, where it is held until you redistribute it to the correct G/L account.
<b>wareki date format</b>	In Japan, a calendar format, such as Showa or Heisei. When a new emperor begins to reign, the government chooses the title of the date format and the year starts over at one. For instance, January 1, 1998, is equal to Heisei 10, January 1st.
<b>wash down</b>	A minor cleanup between similar product runs. Sometimes used in reference to the sanitation process of a food plant.
<b>wchar_t</b>	An internal type of a wide character. Used for writing portable programs for international markets.
<b>web server</b>	A server that sends information as requested by a browser and uses the TCP/IP set of protocols.
<b>work order life cycle</b>	In Capital Asset Management, the sequence of events through which a work order must pass to accurately communicate the progress of the maintenance tasks that it represents.
<b>workfile</b>	A system-generated file that is used for temporary data processing.
<b>workflow</b>	According to the Workflow Management Coalition, workflow means “the automation of a business process, in whole or part, during which documents, information, or tasks are passed from one participant to another for action, according to a set of procedural rules.”
<b>workgroup server</b>	A network server usually containing subsets of data that are replicated from a master network server.
<b>WorldSoftware architecture</b>	The broad spectrum of application design and programming technology that PeopleSoft uses to achieve uniformity, consistency, and complete integration throughout its software.
<b>write payment</b>	A step in processing payments. Writing payments includes printing checks, drafts, and creating a bank tape table.
<b>write-off</b>	A method for getting rid of inconsequential differences between amounts. For example, you can apply a receipt to an invoice and write off the difference. You can write off both overpayments and underpayments.
<b>Z file</b>	For store and forward (network disconnected) user, EnterpriseOne store-and-forward applications perform edits on static data and other critical information that must be valid to process an order. After the initial edits are complete, EnterpriseOne stores the transactions in work tables on the workstation. These work table are called Z files. When a network connection is established, Z files are uploaded to the enterprise server; and the transactions are edited again by a master business function. The master business function then updates the records in your transaction files.
<b>z-process</b>	A process that converts inbound data from an external system into an EnterpriseOne software table or converts outbound data into an interface table for an external system to access.

<b>zusammenfassende meldung</b>	In Germany, the term for the EU Sales Listing.
-------------------------------------	------------------------------------------------

# Index

## A

- Adding records to interface tables, 71
- Advanced planning agent (APAg)
  - overview, 14
- Advanced planning agent (APAg)/Integration
  - overview, 170
- APIs
  - flat files, 83
  - real-time events, 97
  - XAPI events, 107, 120, 121
  - XAPI inbound response, 114, 134
  - XAPI inbound response generation, 124
  - XML XTS, 34
- Autopilot analyzer tool
  - find a business function, 21

## B

- Batch Interface model types, 162
  - advanced planning agent (APAg)/integration, 170
  - electronic data interface (EDI), 169
  - interface tables, 162
  - output stream access (OSA) UBEs, 170
  - table conversion, 170
- Batch interfaces
  - overview, 12, 162
- Benefits, 7
- Business function calls
  - defined, 18
  - finding the right business function, 19, 20, 21
  - overview, 9

## C

- Call object, 50
- Call object error handling, 51
- Call object error text, 51
- Capabilities, 8, 9
  - business function calls, 9, 18
  - events, 10
  - flat files, 10
  - XML, 10
  - XPI, 10
  - Z transactions, 10, 70
- Check for errors, 74
- Connectors
  - overview, 11

- Copying data into outbound interface tables, 165
- create business function documentation, 20
- Creating business function documentation, 20
- Cross reference facility
  - find a business function, 20

## D

- Debug tools
  - find a business function, 21
- Define events
  - XAPI, 112

## E

- EDI
  - overview, 13
- Electronic Data Interface (EDI)
  - overview, 169
- Enterprise XPI. *See* XPI
- Error handling
  - XML dispatch, 47
- Establish session
  - XML element, 24
- Events
  - defining real-time events, 141
  - defining XAPI events, 141
  - defining Z events, 141
  - jde.ini configurations, 139
  - map business function, 136
  - overview, 10, 85
  - real-time events overview, 93
  - real-time events processing, 93
  - reliable delivery, 153
  - reliable delivery set up, 154
  - security for XAPI originator and executor, 138
  - subscribing to real-time events, 149
  - subscribing to XAPI events, 149
  - subscribing to Z events, 149
  - XAPI
    - modify element name, 138
  - XAPI events overview, 105
  - XAPI events processing, 106, 119
  - XAPI inbound response, 113
  - XAPI originator system error handling, 135
  - XAPI response overview, 113
  - Z events overview, 86
  - Z events processing, 86

- Events self-diagnostic utility tool, 155
  - components, 156
  - event generator component, 157
  - event receiver component, 157
  - executing the tool, 158
  - process overview, 156
  - using with database tables, 157
  - XML comparator component, 157
- Example code
  - XML callObject request, 54
  - XML callObject response, 55
  - XML transaction request and response, 59
- Examples
  - sample code to create an XML list, 63
  - sample code to delete data from an XML list, 66
  - sample code to get column information for an XML list, 67
  - sample code to retrieve data from an XML list, 66
  - XML format, 186, 201
  - XML formats
    - events, 203
- Expire session
  - XML element, 25
- Explicit transaction
  - XML element, 25
- Extraction batch process, 165

## F

- Features, 7
- Finding the right business function, 19, 20
  - use autopilot analyzer tool, 21
  - use cross reference facility, 20
  - use debug tools, 21
  - use existing application as model, 20
  - use object management workbench, 20
- Flat files
  - business function, 82
  - converting, 78
  - errors, 82
  - inbound flat file conversion program, 78
  - overview, 10, 76
  - setup, 77
- formats
  - flat files, 76
- Forms
  - Work with Batch Versions, 81

## I

- ID/IDREF support, 52

- Implicit transaction
  - XML element, 25
- Inbound processing using interface tables, 164
- Inbound transaction. *See Z transactions*
- Industry standard support, 17
- Inter-Enterprise XPI. *See XPI*
- Interface table
  - list of processes, 167
- Interface tables
  - adding records, 71
  - extraction batch process, 165
  - inbound processing, 164
  - maintenance, 166
  - outbound processing, 164
  - overview, 13, 162
  - purge data, 75
  - purge records, 166
  - revision application, 166
  - structure, 162
- Interoperability
  - benefits, 7
  - capabilities, 8, 9
  - features, 7
  - industry standard support, 17
  - model, choosing, 15
  - models, 8, 11
  - overview, 7

## J

- jde.ini file settings
  - list-retrieval engine, 67
  - XAPI events, 113
  - XAPI events for EnterpriseOne to EnterpriseOne systems, 135
  - XAPI response, 117
  - XAPI response on an EnterpriseOne client, 118
  - XML CallObject, 54
  - XML Dispatch, 45
  - XML list, 68
  - XML transaction, 58
  - XML XTS, 42
- jde.ini settings
  - Z events, 92
- jdeRequest Type
  - XML element, 23
- jdeResponse Type
  - XML element, 23

## M

- Messaging adapters

- overview, 12
- Models, 8, 11
  - advanced planning agent (APAg), 14
  - batch interfaces, 12
  - connectors, 11
  - EDI, 13
  - interface tables, 13
  - messaging adapters, 12
  - open data access (ODA), 15
  - output stream access (OSA), 14
  - table conversion, 13
- Modify interface table records, 166
- MQSeries. *See* Messaging adapters
- MSMQ. *See* Messaging adapters
- Multiple requests per document, 52

## N

- Name Z transactions, 71

## O

- Object management workbench
  - find a business function, 20
- OCM setup
  - real-time events, 102
  - XAPI events, 112
- ODA
  - add data source, 173
  - add file data source, 174
  - add system data source, 175
  - business view names, 179
  - column security, 180
  - connection string keywords, 178
  - currency, 180
  - decimal shifting, 180
  - delete data source, 177
  - driver architecture, 172
  - error messages, 181
  - hardware and software requirements, 171
  - hardware requirements, 171
  - Julian date, 179
  - long column names, 179
  - long table names, 179
  - media object, 180
  - modify data source, 176
  - ODBC component files, 172
  - row security, 180
  - run Excel query, 181
  - software requirements, 171
  - user defined codes, 180
  - working with, 179
- ODA error messages

- access violation, 183
- attempt to fetch before the first row, 184
- business view contains invalid join, 185
- business view contains unsupported union operator, 185
- column security violation, 183
- configuration request error, 182
- cross system joins not supported, 183
- currency columns can only be simple column references, 183
- data cannot be converted, 184
- data returned for one or more columns was truncated, 184
- data source does not exist, 182
- data source name not valid, 182
- data truncated, 183
- driver does not support requested conversion, 184
- driver not capable, 185
- fractional truncation, 185
- internal data conversion error, 183
- internal execution error, 183
- invalid column number, 184
- invalid cursor state, 184
- invalid date/time string, 184
- invalid numeric string, 184
- invalid request type, 182
- media object columns can only be simple column references, 183
- multiple business views referenced, 185
- numeric value out of range, 184
- option value changed, 182
- option value changed, 185
- server connection failed, 185
- statement must be a select, 184
- syntax error, 183
- unable to allocate memory, 182
- unable to connect to the EnterpriseOne environment, 183
- unable to display connection dialog, 183
- unable to open business view, 185
- unable to open table, 185
- user defined code columns can only be simple column references, 183
- On error handling, 51
- Open Data Access
  - description, 171
- Open data access (ODA)
  - overview, 15
- Outbound batch
  - subsystem business function, 165
- Outbound processing using interface tables, 164

- Outbound transactions. *See* Events
- Output stream access (OSA)
  - overview, 14
- Output stream access (OSA) UBEs
  - overview, 170
- Overview, 7
  - batch interfaces, 12
  - business function calls, 9
  - connectors, 11
  - events, 85
  - Events, 10
  - flat files, 10, 76
  - messaging adapters, 12
  - XML, 10
  - XPI, 10
  - Z transactions, 10

## P

- Prepare/Commit/Rollback
  - XML element, 25

## R

- Real-time events
  - APIs, 97
  - creating a composite event, 99
  - creating an aggregate event, 98
  - creating an event, 97
  - defining, 103, 141
  - example interface call, 97
  - generating, 101
  - jde.ini configuration, 104
  - journaling, 95
  - OCM configuration for events, 102
  - overview, 93
  - processing, 93
  - sequencing, 104
  - subscribing, 149
  - unique ID, 95
- Reliable event delivery, 153
  - set up, 154
- Return NULL values, 53
- review online documentation, 19
- Reviewing online API and business function
  - documentation, 19
- Run a subsystem job, 72
- Run an input batch process, 72

## S

- Selector, 33
- Self-diagnostic test
  - real-time event, 159

- self-diagnostic tool
  - comprehensive system analysis, 161
  - event template, 160
  - subscription services, 160
- Self-diagnostic tool, 155
  - all events, 159
  - customize, 161
  - starting, 158
  - Z event, 159
- Self-diagnsotic tool
  - event list, 160
- Setting up interface tables, 162
- Structure for interface tables, 162
- Subscribing to events, 149
- Subscription information for XAPI events, 113

## T

- Table conversion
  - overview, 13
- Table Conversion
  - overview, 170
- Tasks for generating XAPI events, 111, 135
- TerminateSession
  - XML element, 26
- Troubleshooting
  - XML kernels, 68

## U

- Updating the EnterpriseOne database, 71
- Updating the ERP database, 72
- Using an existing application as a model, 20

## X

- XAPI
  - error handling
    - EnterpriseOne originator system, 135
  - error processing for EnterpriseOne originator
    - and EnterpriseOne executor, 139
- XAPI events
  - APIs, 107, 120, 121
  - defining, 112, 141
  - example create API usage, 108
  - example inbound response parsing API usage
    - from executor, 125
  - example outbound request parsing API usage
    - from originator, 121
  - example XML file, 110
  - example XML from originator, 123
  - example XML response from originator, 133
  - jde.ini file settings, 113

- jde.ini file settings for EnterpriseOne to EnterpriseOne systems, 135
  - journaling, 95
  - OCM configuration for XAPI, 112
  - overview, 105
  - processing, 106, 119
  - routing information, 111
  - subscribing, 149
  - subscription information, 113
  - tasks, 111, 135
  - XAPI inbound response
    - APIs, 114, 134
    - example for creating, 116
    - example parsing API usage, 115
    - processing, 113
  - XAPI inbound response generation
    - APIs, 124
  - XAPI response
    - jde.ini file settings, 117
    - jde.ini file settings for EnterpriseOne client, 118
  - XML
    - APIs for XTS, 34
    - dispatch kernel, 44
    - kernel troubleshooting, 68
    - overview, 10
    - recognizers for XML Dispatch, 44
    - transports for XML Dispatch, 45
    - XML dispatch processing, 44
    - XTS, 28
      - build selector, 33
      - XTS processing, 29
  - XML and EnterpriseOne, 22
  - XML CallObject, 47
    - jde.ini file settings, 54
    - process, 48
    - templates, 47
  - XML CallObject elements, 49
    - call object, 50
    - call object error handling, 51
    - call object error text, 51
    - ID/IDREF support, 52
    - multiple requests per document, 52
    - on error handling, 51
    - return null values, 53
  - XML dispatch
    - jde.ini file settings, 45
  - XML documents
    - EnterpriseOne date standards, 27
    - EnterpriseOne separator standards, 26
    - EnterpriseOne standards, 26
    - formatting, 23
  - XML elements
    - establish session, 24
    - expire session, 25
    - explicit transaction, 25
    - implicit transaction, 25
    - jdeRequest, 23
    - jdeResponse, 23
    - Prepare/Commit/Rollback, 25
    - TerminateSession, 26
  - XML example
    - EnterpriseOne version 1 format, 31
    - native EnterpriseOne format, 29
    - selector creating, 34
  - XML format
    - inbound sales order, 186
      - default values, 201
    - outbound customer create
      - all parameters, 197
    - real-time events template, 220
  - XML list, 60
    - jde.ini file settings, 68
    - list retrieval engine table conversion wrapper, 60
    - process, 61
    - requests, 62
  - XML List
    - creating a list, 63
    - deleting a list, 66
    - get column information for a list, 67
    - retrieve data from a list, 66
  - XML list-retrieval engine
    - jde.ini file settings, 67
  - XML standards
    - creating documents for EnterpriseOne, 26
    - date, 27
    - separators, 26
  - XML system environment settings, 27
  - XML system settings
    - iSeries, 28
    - unix, 27
    - windows and NT, 28
  - XML transaction, 56
    - data request process, 58
    - jde.ini file settings, 58
    - update process, 56
  - XPI
    - overview, 10
  - XTS
    - jde.ini file settings, 42
- ## Z
- ### Z events

- defining, 141
- enabling outbound processing, 88
- flat file cross-reference, 89
- overview, 86
- processing, 86
- processing log table, 90
- purging data from interface tables, 91
- sequencing, 91
- setting up data export controls, 89
- setting up the jde.ini, 92
- subscribing, 149
- subsystem job, 91
- vendor-specific functions, 92
- XML document format, 92

- Z table. *See* Z transaction
- Z transactions, 164
  - adding records to interface tables, 71
  - check for errors, 74
  - input batch process, 71, 72
  - naming, 71
  - overview, 10, 70
  - processing, 70
  - subsystem job, 71
  - subsystem jobs, 72
  - updating EnterpriseOne, 71
  - updating ERP, 72
- Z transactions
  - update confirmation, 74