PeopleSoft®

EnterpriseOne Tools 8.94 PeopleBook: Package Management

November 2004

EnterpriseOne Tools 8.94 PeopleBook: Package Management
SKU E1_TOOLS8.94TPK-B 1104

Open Source Disclosure

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. The following open source software may be used in PeopleSoft products and the following disclaimers are provided.

Apache Software Foundation

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.

OpenSSL

Copyright (c) 1998-2003 The OpenSSL Project. All rights reserved.

THIS SOFTWARE IS PROVIDED BY THE OpenSSL PROJECT "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 OpenSSL PROJECT 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.

SSLeay

Copyright (c) 1995-1998 Eric Young. All rights reserved.

THIS SOFTWARE IS PROVIDED BY ERIC YOUNG "AS IS" AND ANY EXPRESS 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 AUTHOR OR 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.

Loki Library

Copyright (c) 2001 by Andrei Alexandrescu. This code accompanies the book:

Alexandrescu, Andrei. "Modern C++ Design: Generic Programming and Design Patterns Applied". Copyright (c) 2001. Addison-Wesley. Permission to use, copy, modify, distribute and sell this software for any purpose is hereby granted without fee, provided that the above copyright notice appear in all copies and that both that copyright notice and this permission notice appear in supporting documentation.

Contents

About This PeopleBook

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

This preface discusses:

- PeopleSoft application prerequisites.
- PeopleSoft application fundamentals.
- Documentation updates and printed documentation.
- Additional resources.
- Typographical conventions and visual cues.
- Comments and suggestions.
- Common elements in PeopleBooks.

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

PeopleSoft Application Prerequisites

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

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

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

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

PeopleSoft Application Fundamentals

Each application PeopleBook provides implementation and processing information for your PeopleSoft applications. For some applications, additional, essential information describing the setup and design of your system appears in a companion volume of documentation called the application fundamentals PeopleBook. Most PeopleSoft product lines have a version of the application fundamentals PeopleBook. The preface of each PeopleBook identifies the application fundamentals PeopleBooks that are associated with that PeopleBook.

The application fundamentals PeopleBook consists of important topics that apply to many or all PeopleSoft applications across one or more product lines. Whether you are implementing a single application, some combination of applications within the product line, or the entire product line, you should be familiar with the contents of the appropriate application fundamentals PeopleBooks. They provide the starting points for fundamental implementation tasks.

Documentation Updates and Printed Documentation

This section discusses how to:

- Obtain documentation updates.
- Order printed documentation.

Obtaining Documentation Updates

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

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

See Also

PeopleSoft Customer Connection, https://www.peoplesoft.com/corp/en/login.jsp

Ordering Printed Documentation

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

- Web
- Telephone
- Email

Web

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

Telephone

Contact MMA Partners at 877 588 2525.

Email

Send email to MMA Partners at peoplesoftpress@mmapartner.com.

See Also

PeopleSoft Customer Connection, https://www.peoplesoft.com/corp/en/login.jsp

Additional Resources

The following resources are located on the PeopleSoft Customer Connection website:

Resource	Navigation
Application maintenance information	Updates + Fixes
Business process diagrams	Support, Documentation, Business Process Maps
Interactive Services Repository	Interactive Services Repository
Hardware and software requirements	Implement, Optimize + Upgrade, Implementation Guide, Implementation Documentation & Software, Hardware and Software Requirements
Installation guides	Implement, Optimize + Upgrade, Implementation Guide, Implementation Documentation & Software, Installation Guides and Notes
Integration information	Implement, Optimize + Upgrade, Implementation Guide, Implementation Documentation and Software, Pre-built Integrations for PeopleSoft Enterprise and PeopleSoft EnterpriseOne Applications
Minimum technical requirements (MTRs) (EnterpriseOne only)	Implement, Optimize + Upgrade, Implementation Guide, Supported Platforms
PeopleBook documentation updates	Support, Documentation, Documentation Updates
PeopleSoft support policy	Support, Support Policy
Prerelease notes	Support, Documentation, Documentation Updates, Category, Prerelease Notes
Product release roadmap	Support, Roadmaps + Schedules
Release notes	Support, Documentation, Documentation Updates, Category, Release Notes
Release value proposition	Support, Documentation, Documentation Updates, Category, Release Value Proposition
Statement of direction	Support, Documentation, Documentation Updates, Category, Statement of Direction

Resource	Navigation
Troubleshooting information	Support, Troubleshooting
Upgrade documentation	Support, Documentation, Upgrade Documentation and Scripts

Typographical Conventions and Visual Cues

This section discusses:

- Typographical conventions.
- Visual cues.
- Country, region, and industry identifiers.
- Currency codes.

Typographical Conventions

This table contains the typographical conventions that are used in PeopleBooks:

Typographical Convention or Visual Cue	Description
Bold	Indicates PeopleCode function names, business function names, event names, system function names, method names, language constructs, and PeopleCode reserved words that must be included literally in the function call.
Italics	Indicates field values, emphasis, and PeopleSoft or other book-length publication titles. In PeopleCode syntax, italic items are placeholders for arguments that your program must supply. We also use italics when we refer to words as words or letters as letters, as in the following: Enter the letter *O*.
KEY+KEY	Indicates a key combination action. For example, a plus sign (+) between keys means that you must hold down the first key while you press the second key. For ALT+W, hold down the ALT key while you press the W key.
`Monospace font`	Indicates a PeopleCode program or other code example.
“ ” (quotation marks)	Indicate chapter titles in cross-references and words that are used differently from their intended meanings.

Typographical Convention or Visual Cue	Description
. . . (ellipses)	Indicate that the preceding item or series can be repeated any number of times in PeopleCode syntax.
{ } (curly braces)	Indicate a choice between two options in PeopleCode syntax. Options are separated by a pipe (|).
[ ] (square brackets)	Indicate optional items in PeopleCode syntax.
& (ampersand)	When placed before a parameter in PeopleCode syntax, an ampersand indicates that the parameter is an already instantiated object. Ampersands also precede all PeopleCode variables.

Visual Cues

PeopleBooks contain the following visual cues.

Notes

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

Note. Example of a note.

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

Important! Example of an important note.

Warnings

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

Warning! Example of a warning.

Cross-References

PeopleBooks provide cross-references either under the heading “See Also” or on a separate line preceded by the word *See.* Cross-references lead to other documentation that is pertinent to the immediately preceding documentation.

Country, Region, and Industry Identifiers

Information that applies only to a specific country, region, or industry is preceded by a standard identifier in parentheses. This identifier typically appears at the beginning of a section heading, but it may also appear at the beginning of a note or other text.

Example of a country-specific heading: “(FRA) Hiring an Employee”

Example of a region-specific heading: "(Latin America) Setting Up Depreciation"

Country Identifiers

Countries are identified with the International Organization for Standardization (ISO) country code.

Region Identifiers

Regions are identified by the region name. The following region identifiers may appear in PeopleBooks:

- Asia Pacific
- Europe
- Latin America
- North America

Industry Identifiers

Industries are identified by the industry name or by an abbreviation for that industry. The following industry identifiers may appear in PeopleBooks:

- USF (U.S. Federal)
- E&G (Education and Government)

Currency Codes

Monetary amounts are identified by the ISO currency code.

Comments and Suggestions

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

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

Or send email comments to doc@peoplesoft.com.

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

Common Elements Used in PeopleBooks

Address Book Number Enter a unique number that identifies the master record for the entity. An address book number can be the identifier for a customer, supplier, company, employee, applicant, participant, tenant, location, and so on. Depending on the application, the field on the form might refer to the address book number as the customer number, supplier number, or company number, employee or applicant id, participant number, and so on.

As If Currency Code Enter the three-character code to specify the currency that you want to use to view transaction amounts. This code allows you to view the transaction amounts as if they were entered in the specified currency rather than the foreign or domestic currency that was used when the transaction was originally entered.

Batch Number Displays a number that identifies a group of transactions to be processed by the system. On entry forms, you can assign the batch number or the system can assign it through the Next Numbers program (P0002).

Batch Date Enter the date in which a batch is created. If you leave this field blank, the system supplies the system date as the batch date.

Batch Status Displays a code from user-defined code (UDC) table 98/IC that indicates the posting status of a batch. Values are:

Blank: Batch is unposted and pending approval.

A: The batch is approved for posting, has no errors and is in balance, but it has not yet been posted.

D: The batch posted successfully.

E: The batch is in error. You must correct the batch before it can post.

P: The system is in the process of posting the batch. The batch is unavailable until the posting process is complete. If errors occur during the post, the batch status changes to E.

U: The batch is temporarily unavailable because someone is working with it, or the batch appears to be in use because a power failure occurred while the batch was open.

Branch/Plant Enter a code that identifies a separate entity as a warehouse location, job, project, work center, branch, or plant in which distribution and manufacturing activities occur. In some systems, this is called a business unit.

Business Unit Enter the alphanumeric code that identifies a separate entity within a business for which you want to track costs. In some systems, this is called a branch/plant.

Category Code Enter the code that represents a specific category code. Category codes are user-defined codes that you customize to handle the tracking and reporting requirements of your organization.

Company Enter a code that identifies a specific organization, fund, or other reporting entity. The company code must already exist in the F0010 table and must identify a reporting entity that has a complete balance sheet.

Currency Code Enter the three-character code that represents the currency of the transaction. PeopleSoft EnterpriseOne provides currency codes that are recognized by the International Organization for Standardization (ISO). The system stores currency codes in the F0013 table.

Document Company Enter the company number associated with the document. This number, used in conjunction with the document number, document type, and general ledger date, uniquely identifies an original document.

If you assign next numbers by company and fiscal year, the system uses the document company to retrieve the correct next number for that company.

If two or more original documents have the same document number and document type, you can use the document company to display the document that you want.

Document Number — Displays a number that identifies the original document, which can be a voucher, invoice, journal entry, or time sheet, and so on. On entry forms, you can assign the original document number or the system can assign it through the Next Numbers program.

Document Type — Enter the two-character UDC, from UDC table 00/DT, that identifies the origin and purpose of the transaction, such as a voucher, invoice, journal entry, or time sheet. PeopleSoft EnterpriseOne reserves these prefixes for the document types indicated:

P: Accounts payable documents.

R: Accounts receivable documents.

T: Time and pay documents.

I: Inventory documents.

O: Purchase order documents.

S: Sales order documents.

Effective Date — Enter the date on which an address, item, transaction, or record becomes active. The meaning of this field differs, depending on the program. For example, the effective date can represent any of these dates:

- The date on which a change of address becomes effective.
- The date on which a lease becomes effective
- The date on which a price becomes effective.
- The date on which the currency exchange rate becomes effective.
- The date on which a tax rate becomes effective.

Fiscal Period and **Fiscal Year** — Enter a number that identifies the general ledger period and year. For many programs, you can leave these fields blank to use the current fiscal period and year defined in the Company Names & Number program (P0010)

G/L Date (general ledger date) — Enter the date that identifies the financial period to which a transaction will be posted. The system compares the date that you enter on the transaction to the fiscal date pattern assigned to the company to retrieve the appropriate fiscal period number and year, as well as to perform date validations.

Package Management Preface

This guide is written primarily for system administrators and others who manage custom modifications to the environments. Pristine objects and modifications are deployed to servers and workstations through various types of packages.

Understanding Package Management

The Package Management Guide describes how to assemble, build, and deploy packages to servers and workstations, and contains information about these topics:

- Getting Started
- Working with Objects
- Assembling Packages
- Building Packages
- Deploying Packages
- Setting Up Multitier Deployment

CHAPTER 1

Getting Started with Package Management

This chapter introduces the main concepts of Package Management. It discusses:

- The CNC documentation suite
- Package management processes
- Understanding packages
- Types of packages
- Recommendations for developers
- Tracking changed objects
- Ensuring the integrity of the production environment
- Deployment Methods
- Implementing EnterpriseOne packages

The CNC Documentation Suite

The documentation suite for Configuration Planning and Setup is designed for Configurable Networking Computing (CNC) specialists, system administrators, and network and server administrators. This documentation is intended for users who have completed the initial installation and defined the standard data sources, path codes, and environments. Use the Configuration Planning and Setup documentation to make changes or additions to the configuration setup after the initial installation.

The Configuration Planning and Setup suite consists of these guides:

- Configurable Network Computing Implementation Guide. This guide is written primarily for CNC specialists and contains information about these topics:
- Working with middleware
- Understanding and verifying data sources
- Creating path codes and environments
- Working with Object Configuration Manager
- Understanding the different modes of processing
- Configuring the software for a typical customer
- System Administration Guide. This guide is written primarily for system administrators and contains information about these topics:
- Setting up data replication

- Setting up printers
- Working with servers
- Setting up user profiles and security
- Working with data dictionary and vocabulary overrides
- Understanding transaction processing
- Working with media objects and imaging
- Using the universal table browser
- Understanding naming conventions
- Working with the jde.ini file
- Package Management Guide. This guide is written primarily for system administrators and others who manage custom modifications to the environments. The Package Management Guide contains information about these topics:
- Server and Workstation Administration Guide. This guide is written primarily for network administrators and contains information about these topics:
- Snapshot (multi-client installer)
- Server administration
- Troubleshooting the workstation and the server

Although every attempt has been made to organize the information in the Configuration Planning and Setup suite according to related tasks, you might find that the documentation for the various tasks that you need to perform appears in more than one guide. For example, the documentation for setting up path codes, environments, and data sources is in the Configurable Network Computing Implementation Guide, while the documentation for building and deploying packages is in the Package Management Guide.

The Configuration Planning and Setup suite is the central location for all CNC-related tasks except for:

- Initial installation of PeopleSoft EnterpriseOne. See the appropriate installation guide for your platform.
- PeopleSoft EnterpriseOne upgrade. See the appropriate upgrade guide for your platform.
- Network infrastructure and third-party software setup and maintenance. PeopleSoft does not provide documentation for third-party software or hardware; the applicable software or hardware vendor provides this information.

You do not need to understand the installation process completely to perform configuration planning and setup tasks. However, to use the Configuration Planning and Setup guides you must understand what the installation accomplishes.

Package Management Processes

This guide describes how to set up and maintain processes to develop and deploy custom modifications that are created with the PeopleSoft EnterpriseOne tools. You can use the guide to set up an environment in which you can deploy custom modifications that were made with development tools. It also provides information about applying modification rules, transferring objects, checking out development objects, and working with the data dictionary.

If you have already set up your environment and are comfortable with these concepts, you can skip to the sections that describe how to create and schedule packages and how to deploy packages to workstations and servers.

Roles

During an implementation, both customers and consultants typically fill these roles:

- CNC consultant and CNC administrator
- Custom solution consultant and application developer
- Application consultant and application project leader
- Hardware, network, and third-party software consultant and administrator

Typically, both consultants and customers participate in an implementation. Consultants perform these roles:

- CNC consultant
- Custom solution consultant
- Application consultant
- Hardware, network, and third-party software consultant

Customers perform these roles, which parallel the consultant roles:

- CNC administrator
- Application developer
- Application project leader
- Hardware, network, and third-party software administrator

After the implementation, the consultants typically have fewer responsibilities. Therefore, customers must receive adequate training for the roles that they fill.

CNC Consultant and CNC Administrator

The CNC consultant and CNC administrator install PeopleSoft EnterpriseOne and set up environments, users, security, distributed processing, and data replication. They also are responsible for setting up version control and testing various CNC configurations. The CNC consultant and CNC administrator control the deployment of EnterpriseOne software throughout the company.

Custom Solution Consultant and Application Developers

Custom solution consultants resolve business issues by developing applications. Their primary responsibilities include designing the modifications with upgrades in mind; and developing, testing, and introducing the configured software. While the CNC administrator performs the version control functions that build and deploy software, the customer solution consultant must help develop the internal procedures that document the application development cycle for your business.

Application Consultants and Application Project Leaders

After PeopleSoft EnterpriseOne software is installed, configured, and deployed, the application consultants continue in their role as product experts, where they might be called on to troubleshoot problems that arise. Although application consultants do not implement the CNC configurations, they must understand how PeopleSoft EnterpriseOne handles distributed processing, data replication, environments, and so on, because these application issues influence the CNC decisions.

Hardware, Network, and Third-Party Software Consultants and Administrators

Implementing PeopleSoft EnterpriseOne includes many tasks that are outside the scope of PeopleSoft services. Third-party consultants provide these services. They might also work as CNC consultants, network architects, custom modification consultants, and so on.

Understanding Packages

The purpose of a package is to group software modifications so that they can be deployed to workstations and enterprise servers. A package defines and contains the components to deploy. A package can contain everything needed to run the software (as in an initial installation for a new workstation) or only updates or changes to existing applications.

Why packages are needed

As applications, business functions, and other objects change, you need to make those changes available to users within your enterprise. You might also need to set up a new workstation with PeopleSoft EnterpriseOne software.

Packages enable you to deploy software changes and new applications to your users, or to install the software on a workstation for the first time. After you have defined and built a package, you can deploy it using the Client Workstation Installation program or Deployment Director program (P9631).

You might need to update or set up a workstation or an enterprise, logic, or application server with PeopleSoft EnterpriseOne software for a variety of reasons, such as:

- You want to set up a workstation for a new user or role.
- You need to deploy custom solutions to all users or only to selected users.
- You have created a new path code for development purposes, and you need to deploy it.
- You need to rapidly deploy a software fix to a selected group of affected users.
- Disk space is getting low on some of your workstations, and you need to create a minimum configuration.
- You need to update your servers with custom modifications that you have developed using the toolset.

PeopleSoft EnterpriseOne provides solutions to meet all of these needs. First, the system enables you to create a package that defines and contains the location of the components that need to be distributed to your workstations. In order to deploy these components, you must define and build a package.

Types of Packages

You can build these types of packages:

- Full client
- Full server
- Full mobile
- Update client
- Update server

- Update mobile

Full Client Packages

Full packages are static, point-in-time snapshots of the central objects for the path code on which the package is based. A full package contains everything developers need to run to develop in PeopleSoft EnterpriseOne software. Specifically, a full package includes a full set of TAM specifications, a full set of DLLs, all of the .c and .h files for business functions and tables, an INF file that defines where the foundation, data, helps, and packages are located, and other information about the package. Select this package type when you want to create a full workstation configuration.

In a full package, every application for which users are licensed is available to them. Because specifications reside on the workstation, information is processed locally, which eliminates network traffic. When you deploy a full package, a few specifications and tables might be installed just-in-time but the impact on network performance is insignificant.

Full packages are primarily for initial installations and are normally deployed using the Client Workstation Installation program. You can also use the Deployment Director program (P9631) to install a full package on a computer on which EnterpriseOne is already installed.

Full packages can also include development objects such as business function source files, object files, and header files. A user who needs to load development objects has the option to do so at deployment time.

Full Server Packages

Full server packages are the same as client packages except for the following:

- The server package does not include client-specific business functions.
- The server package does not include application specs.
- During the package build process, you can only select to build the run time system code on a server package. The administrator does not have the option to select different system code.
- The server package does not include features or local data.

Full Mobile Packages

Full mobile packages are similar to full packages but contain only a subset of the specs delivered with a full package. A mobile package includes all Business Functions, but includes only those application and UBE objects tagged as “mobile.” To tag objects for mobile packages, the system administrator must manually flag each object as mobile within Object Management Workbench. This tagging is limited to applications and UBEs. The administrator should tag all those applications and UBEs that users needs to perform their assigned tasks within EnterpriseOne.

A full mobile package can only be built if the EnterpriseOne client has at least one environment in the path code for which the package is being built marked as “WAN.” In this case, the package build program uses the OCM for that environment to build a mirror image of all EnterpriseOne tables in a RDBMS format. These tables are then deployed to an EnterpriseOne workstation on a WIN32 client. If the path code has more than one environment marked as “WAN,” the package build program selects the first environment it encounters in the collating sequence of the RDBMS.

When Full Mobile packages are deployed to workstations, the mobile specs are copied to the workstation in RDBMS format, and the OCM for all installed tables is changed to point to the local repository. Users can then access EnterpriseOne applications and perform tasks while disconnected from the network. Once the user reconnects to the network, the data within the local repository is synchronized with the RDBMS database stored on the EnterpriseOne database server. During this process, data is updated in both directions.

Update Client Packages

The update package enables you to update, add to, or refresh your existing full package or full mobile package with changed objects. You can deploy an update package only to a workstation that already has PeopleSoft EnterpriseOne installed on it. The objects in the update package replace those objects on the workstation. All other objects on the workstation are left untouched. The advantage of this type of package is that you can quickly deploy software changes or enhancements.

Unless a package includes applications that do not have specifications, the update package is a point-in-time copy of your central objects for a particular path code. If the update package contains an application without specifications, only that application is *dynamic*; the rest of the package is static. Dynamic means that the system copies the applications directly from the central objects data source at the moment that the user first selects the application.

When a user signs on to EnterpriseOne after receiving one or more update packages, a form that lists the update packages appears. If the user decides to take one or more of these packages, the system loads the user's workstation with all objects in the package. The objects in an update package replace those same objects on the workstation. All other objects on the workstation remain the same.

Like full packages, update packages can include development objects such as business function source files, object specs, and header files. Update package recipients have the option to load development objects at deployment time.

Because of the way just-in-time installation works, performance across a WAN might be slow if the update package contains only applications without specifications. To improve performance on the WAN, include in the package the specifications for each application.

All update packages require a full package (as the parent package) on which the update package is based. By default, the parent package is updated by the update package. When this occurs, all objects in the update package are merged into the parent package.

Note. The jde.ini file on the client includes a setting called UpdateParentPackage that determines whether the parent is updated. If this setting is not present or set to "Y," the parent package is updated by the update package. If this setting is "N," the parent is not updated.

Business function objects in the update package are linked to the corresponding objects in the parent package, and new DLLs are created. Similarly, specifications from the update package are merged into the specifications in the parent package.

The parent package concept applies to both workstations and servers. Parent packages for workstations reside on the deployment server, while server parent packages are kept in the build area for the enterprise server.

Update Server Packages

Update server packages are the same as update client packages, with the following exceptions:

- Update server packages only include the objects described above for full server packages.
- The specs for selected objects are not built into the update package. Instead, they are merged directly into the parent package.

Update Mobile Packages

An update mobile package is update package that is used to update mobile clients. An update mobile package is automatically built when the parent package is a marked as "Mobile."

Recommendations for Developers

When developers install a full package, they must select the option to install development objects. When they install the development objects with a full package, these objects are automatically updated when they install an update package. Developers receive source code, header files, and libraries, and DLLs.

Tracking Changed Objects

Managing modifications requires a practical version control plan for tracking changed objects. You can avoid many software problems by tracking changed objects.

To more easily plan and track your development and to simplify version control, you should build and deploy packages only as often as necessary. If you perform many development changes, you should build and deploy packages on a set schedule to ensure that everyone involved knows when objects are due to be completed and when you are going to build and deploy the package.

Implementing version control might require a staff of information technology professionals. For example, a company has several hundred developers and a complex CNC configuration. To manage version control for multiple developers, the product version control group consists of:

- One manager to oversee coordination within the department
- One supervisor to coordinate the package builds, coordinate object transfers, and troubleshoot problems
- Two server specialists to build server packages
- Four technical specialists to build workstation packages, perform object transfers, and run automated testing before releasing the package to Quality Assurance
- One night operator to build workstation packages, build server packages, and clear build errors

Working with Path Codes

If you are not planning any development projects, you need only three path codes (sets of central objects): PY811, PD811, and PS811. If you plan to modify the software extensively, you also should create a development path code (DV811).

Because each path code requires version control maintenance, you should create only the path codes that you really need. Even when you make extensive software modifications, you should have only these four path codes:

DV811	The path code that you use for routine development. After successfully testing the objects that you develop, transfer them to your PY811 path code using the Object Transfer program, and distribute them to your users using the package build deployment process.
PY811	A path code that contains a practice set of objects that you test during conference room pilot before you transfer objects to production. Use this path code to deploy quick corrections or make minor modifications to objects that you will transfer to production. You also can use this path code to test modifications that were made in the development path code before you transfer the objects to the production path code.
PD811	The production path code from which just-in-time installations and production server objects are deployed. After you test software changes in PY811, transfer them to PD811, and then deploy the changes to your enterprise servers and workstations.
PS811	The set of pristine objects that is included with the software. You should not make changes to this path code other than to apply changes that PeopleSoft documents. Use this path code to compare standard software to any custom software that you have implemented in other path codes. You should keep a copy of this path code so that you have an unchanged copy of PeopleSoft EnterpriseOne in case you need to undo any of your changes.

All path codes share the same Object Librarian tables, the same system data source, and, usually, the same data dictionary. The only tables that are distinct for each path code are the central objects and specifications tables (tables that begin with F987), the Versions List table (F983051), the Processing Option Text table (F98306), and the User Overrides Table (F98950).

Suggested Package Names

It is recommended that you maintain two versions of each package-an A and a B version-so that you can alternate between these versions when you build packages. The advantage of this approach is that users always have a package available to them, even when you are building the latest version of that package. For example, package PRODB would be available to users while you are building PRODA. Then, after you release PRODA, you would build the next package into PRODB, and so on. This setup gives you two full packages (A and B) for production, as illustrated in the following table:

PD811FA	Standard Production Full A
PD811FB	Standard Production Full B

Update packages might be named as follows:

PD811UA	Production Update Package 1
PD811UB	Production Update Package 2

PD811UA	Production Update Package 3
PD811UB	Production Update Package 4

See Also

EnterpriseOne Tools 8.94 PeopleBook: System Administration, "Using Security Workbench," Understanding Security Workbench

Ensuring the Integrity of the Production Environment

As soon as you transfer objects into your PD811 path code, end users can access the changes. Therefore, you should test the modified objects before you transfer them to the production path code.

After you transfer objects to the production path code, they are immediately deployed to end users under these circumstances:

- When a user is set up for just-in-time installation (JITI), the system automatically deploys the object to the user's workstation the first time that the user attempts to access the object.

  Note. JITI affects users who received an update package containing an application without specifications. For more information about JITI, see Understanding Just-in-Time Installation in the Package Management Guide.

- When you build an update package that includes a business function build for that package, the system builds the business function and then globally links it with all other business functions in the parent package.

To ensure that the modifications that you transfer to your production path code are not immediately available to end users, avoid using update packages that contain applications with no specifications. Also, do not transfer business functions into the production path code until you are ready to deploy because, during a package build, a global build of business functions automatically includes the new functions. When you transfer changes into the production path code, they will not be available to users until you build a full or update package.

Working with the Normal Development Process

The following lists provide an overview of how you should perform your normal development cycle.

In the DV811 path code, complete these tasks:

- Make modifications
- Test the modifications
- Transfer the objects to PY811

In the PY811 path code, complete these tasks:

- Build the package
- Test the modifications
- Deploy server objects to the CRP path code on the enterprise server, and then test the objects
- Schedule the package

- Transfer the objects to PD811

In the PD811 path code, complete these tasks:

- Build the package
- Schedule the package
- Deploy the server objects to the PD811 path code on the enterprise server, and then test the objects

A typical development process

The steps that follow are a typical process for modifying objects and deploying them through successive path codes and into the production environment.

1. Check out your objects from the DV811 path code, modify them, test them, and check them back in.
2. Use the SAR system (or any numbering system) to track your changes.

   Always use a SAR number when you check in the objects.
3. If the objects need to reside on the logic server, transfer them to the DV811 path code on that server.
4. Test the objects by comparing them to the objects on the server.
5. Use a SAR number with Object Transfer to transfer the objects to the PY811 path code.

   Use the check-out log to confirm the transfer (optional). The objects are not in production, but they are now available for you to build a test package in the PY811 path code.
6. Build a full or update package.
7. Test the newly built but unreleased package in the PY811 path code.

   You test the package only by comparing it to workstation processes, not to server processes. Although the name of this package will probably be PY811U1 (update package number 1 for the CRP path code), it is a test package because you have not released it to your users.
8. Schedule the update package to deploy to a test machine and test it in an environment that contains CRP objects with CRP data.
9. Deploy server objects to the PY811 path code on the enterprise server and test them.

   If you prefer, you can build the server package and schedule the deployment at the same time that you build and schedule the workstation package. Building these packages simultaneously can save you time, although this method puts a greater load on the server.
10. Schedule the new package to deploy to CRP users.
11. Use a SAR number with Object Transfer to transfer the object to the PD811 path code.

    Use the check-out log to confirm the transfer (optional). The objects are now in the production environment and are available for you to build a package in the PD811 path code.

    Note. If just-in-time installation is enabled, users can access the objects immediately.
12. Build a full or update package for client workstations.
13. Perform a server package build.

    You can transfer the server package now or wait until it has been tested on a workstation.
14. Schedule the new package to deploy to end-user workstations.
15. Deploy the server objects to the PD811 path code on the enterprise server and test them.

If you prefer, you can build the server package and schedule the deployment at the same time that you build and schedule the workstation package.

Developing Short-Term Changes

Sometimes you need to make a simple change to an application that is undergoing major enhancement work in the DV811 path code. When an object has been modified in the DV811 path code, you might encounter unpredictable results if you make a simple change and quickly deploy it. Therefore, you should make the change in both the PY811 and the DV811 path codes, and deploy the change using the PY811 path code. This method enables you to deploy the change quickly to users without interfering with the major enhancement work in the DV811 path code

Deployment Methods

After you have made software changes, the method that you use to deploy those changes to the workstations on your enterprise depends on factors such as the type of package that you typically build and the needs of your users.

PeopleSoft EnterpriseOne offers several deployment programs, each with its own specific purpose and advantages. The method that you select depends mainly on the type of package that you want to deploy.

This section discusses:

- Package Deployment
- Multitier Deployment
- Cumulative and noncumulative update packages
- Comparing deployment methods
- Deploying various types of modifications
- Just-In-Time Installation

Package Deployment

The Deployment Director program (P9631) enables the administrator to specify the date and time that a package is made available to users or groups. The administrator can specify whether the package is mandatory or optional. If a package is mandatory, users who receive the package will be unable to access PeopleSoft EnterpriseOne software until they install the package.

Fat client users who receive a scheduled package can install the package immediately after they log on to EnterpriseOne on the specified date. If they decide to install the package, the installation program launches, and the package loads. The user can also select to install the package later or to decline the installation altogether (unless the package is mandatory).

For server deployment, the P9631 application runs a UBE that automatically deploys the package to the targeted server.

Multitier Deployment

Multitier deployment enables workstations to install software from more than one deployment location and more than one deployment server. It is recommended that you consider multitier deployment if your site has more than 50 workstations performing software installations per day, or if you are deploying PeopleSoft EnterpriseOne software across a WAN connection.

Cumulative and Noncumulative Update Packages

When you use a cumulative update strategy for deploying packages, you have one package that you add to, rebuild, and re-release to users. You do not create a new package each time you have a modification that you want to deploy. To use a cumulative package, follow these steps:

1. Change the package status in order to add the new modifications.
2. Add the changed or new objects to the package.
3. Rebuild the package.
4. Redeploy the package.

When you use a noncumulative update strategy, you create and deploy a different package each time you add or change an object. For example, if you deploy one modification a week for 10 weeks, you would have 10 different packages, each containing only the software change for that week.

Comparing Deployment Methods

Each deployment method has strengths and limitations. To help you decide which method is right for your needs, here are key points about the different methods:

- A new user loading a new machine should use the Install Manager to load a full package, plus any update packages that you have instructed users to load since the last package build. Therefore, you need a manual tracking system to track which update packages must be applied after installing a particular package.
- All update packages must use the Deployment Director program (P9631) to be scheduled to workstations unless you are using the Push Installation feature.
- Full package can also use Deployment Director if PeopleSoft EnterpriseOne is already loaded on a machine. (You can use the Push Installation feature to deploy to a machine that does not have EnterpriseOne installed.)
- Use the Silent Installation program to submit a workstation installation request through command line arguments. Do not use this program for an initial installation.
- Use the Multitier Deployment process to install from more than one deployment location. You should consider this method if you have more than 50 workstations performing software installations per day or if you have users on a WAN.
- Use the Deployment Director program (P9631) when you need to push objects in a server package to enterprise servers.

Deploying Various Types of Modifications

An understanding of which types of objects (and therefore modifications) can be deployed through each package type will help you select the appropriate package for the changes you deploy.

The following table illustrates which types of changes are installed with a full package, an update package with specifications, and an update package with no specifications.

Modification	Full Package	Update Package with Specs, Business Functions, and NERs	Update Package with Business Functions and NERs but no Specs
Applications			
Imbedded event rules	X	X	
Vocabulary overrides (FDA text)	X	X	
Data structure	X	X	
Processing options (report)	X	X	
Business Functions			
C Language source/include /object (if a compiler exists)	X	X	X
Consolidated business function DLLs	X	X	X
Data structure	X	X	
Table event rules	X	X	X
Named event rules	X	X	X
Batch Applications			
Report	X	X	
Imbedded event rules in a report	X	X	
Report data structure	X	X	
Report vocabulary overrides	X	X	
Report processing options	X	X	
Versions and processing option values (depends on processing options)	X	X	
Imbedded event rules in versions	X	X	
Processing option templates	X	X	
Business Views			
Added or changed fields	X	X	

Modification	Full Package	Update Package with Specs, Business Functions, and NERs	Update Package with Business Functions and NERs but no Specs
Tables			
Structure (specifications)	X	X	
Indexes	X	X	
Joins	X	X	
Miscellaneous			
Helps stored on the server			
Helps stored on the workstation	X	X	
Generic text data structure	X	X	
Data dictionary items			
Foundation code (required for full packages, optional for update packages)	X	X	X
Foreign languages	X	X	
Non-PeopleSoft objects (custom items must be defined in the EnterpriseOne Central Objects database, and can be deployed through any package type)	X	X	
Replicated local data (required for full packages, optional for update packages)	X	X	X
New icons	X	X	

For an update package with JITI types, these changes are installed:

- Applications
- Imbedded ER
- Vocabulary overrides (FDA text)
- Data structure
- Processing options (report)
- Batch Applications

- Report
- Imbedded ER in a report
- Report data structure
- Report vocabulary overrides
- Report processing options
- Versions and processing option values (depends on processing options)
- Imbedded ER in versions
- Business Views
- Added or changed fields
- Miscellaneous
- Foundation code (required for full packages, optional for update packages)
- Foreign languages
- Non-PeopleSoft objects (custom items can be deployed through any package type)
- Replicated local data (required for full packages, optional for update packages)

Just-in-Time Installation

Just-in-Time Installation (JITI) occurs when the system retrieves an application at runtime and loads it on the workstation the first time that you select that application from the menu. Loading happens only once; the next time that you select the same application, it is still loaded on the workstation. JITI applies only to applications. Business functions cannot be installed through JITI.

JITI works when your workstation receives an update package that does not contain specifications. Update packages that contain specifications do not require the JITI process.

When you receive an update package that contains a changed application without the new specifications, the system first determines whether specifications for the application reside on your workstation. If so, it deletes from your workstation old versions of that application. Then, the next time that you select that application from the menu, the system loads the new version of that application.

JITI can be used in remote locations that are using multitiered deployment to install packages. However, you might find that performance time for the JITI is unacceptable. In this case, you can initially install full packages and use update packages to deploy software changes.

When a package includes applications without specifications, at the time of execution only these related objects are deployed:

- Interactive or batch application specifications
- Embedded event rules for the application
- Processing option templates, data structures, and related business views

The listed objects are not deployed, and, therefore, must be included in the package if they have been modified:

- Business functions and their data structures
- Generic text data structures
- Table event rules, which are included with tables
- Named event rules

- New icons

Recommendations for Sites Using Full Packages with JITI

For sites that use full packages with JITI, it is recommended that you adopt a cumulative update package strategy. Each week that you need to deploy a change, add that object to the existing update package, and then rebuild and schedule the package.

The advantage of this strategy is that you do not need to rebuild your full package each week. By using this strategy to deploy packages to a new workstation (or to completely refresh any workstation), you must install the full package and then install one cumulative update package.

The disadvantage of this strategy is that the update package might become so large that the deployment time increases. You must determine when to rebuild the full package for new workstations and enable existing users to install the new update package.

Disabling Just-In-Time Installation

Select one of these methods to disable just-in-time installation (JITI):

- Use the Work with Environments program (P0094) to disable JITI for the environment. When JITI is disabled, users who sign on to that environment can access only those applications for which they have specifications. It is recommended that you use this method during the cumulative installation process when you update your production central objects with new changes.
- Use application security to disable JITI for a particular application, for an end user, or for a role profile. When a user accesses an application for which the specifications do not reside on the user's workstation, the system first reviews P0094 to determine whether JITI is disabled for the entire environment. If JITI is on for the environment, the system determines whether application security prevents the user from using JITI. Application security has a field called "not allowed to install."

Although you can disable JITI for an environment, the system still uses JITI to copy data dictionary items to the global tables and data dictionary tables. The structure of the data dictionary prevents you from disabling JITI for it.

Implementing EnterpriseOne Packages

The following is an overview of the steps for creating and deploying a package:

1. Assemble the package.

   During this step, you specify the type of package that you are building and provide a name, path code, and package description. Next, you assemble your package by specifying the objects, helps, data, foundation, features, and so on, that you want to include in the package. If you are building an update package, you can specify individual objects to include.

   To simplify the process of assembling a package, the Package Assembly program (P9601) includes the Package Assembly Director, which displays a series of forms that guide you through the steps of naming your package and adding the objects that you want to include in the package.

2. Define the package build.

   After you assemble the package, you must define the build before you can deploy the package to your workstations and servers. In this step, you specify:

   - Build options

- Build specification options
- Business functions build options
- Compression options
- Build features options

  You also need to specify whether the package is for a workstation, a server, or both. If the package is for servers, you must specify the servers for which the package should be built.

  Again, to simplify the build process, the Package Build Director program (P9621) includes the Package Build Definition Director, which displays a series of forms that guide you through the steps of specifying where to build the package, whether to include specifications, whether to compress or build business functions, and so on.

3. Build the package.

   During the actual build process, the system takes the information that you provided when you assembled and defined the package and copies and converts central objects to the package. It also globally builds the business functions that are included in the package and then compresses the package.

4. Schedule the package for deployment

   After you have defined and built your package, it is ready for distribution. Depending on the package type, you can deploy packages through the Client Workstation Installation program or the Deployment Director program (P9631).

   Deployment Director enables you to specify the workstations and servers that receive the package, as well as when the package is made available. Packages can be deployed to all computers within the enterprise, a select group of computers, or individual computers. You can schedule a package to be *pushed* from the deployment server to workstations. Push installation requires no interaction with the workstation users.

   When you schedule the package, you can indicate whether package installation is mandatory or optional. At this same time, you can specify whether you want the package to be deployed using push installation, which requires no interaction with the package recipient.

5. Deploy the package to deployment and enterprise servers.

   Use the Deployment Director program (P9631) to move any changed objects to the enterprise server.

   If you specify a server during the package build definition process, the system automatically creates a corresponding server package in the correct format. If you do not specify a server and define only a workstation package, you should create a corresponding server package. The process is nearly identical to creating a workstation package.

CHAPTER 2

Working with Objects

This chapter provides overviews of objects and modification rules and describes how to copy records between data sources.

Understanding Objects

By industry standards, an *object* is a self-sufficient entity that contains data as well as the structures and functions used to manipulate the data. An object is a reusable entity that is based on software *specifications*. A specification is a complete description of a system object. Each object has one or more of its own specifications, which are stored on both the server and the workstation.

The system stores objects in two places:

- A central-storage server, where you keep your central objects. When a developer checks out a central object, it is replicated on the workstation and then converted back into a central object when the developer checks it in.
- Workstations and servers, which store run-time objects. Run-time objects are replicated objects that actually run the system.

The following conceptual information describes how objects are moved between the central objects location and the object destinations.

Object Movement

When you perform any of the following tasks, objects move between the central objects location and the object destination:

- Check objects in or out
- Add existing objects to a project
- Perform a get object from the Object Management Workbench program (P98220)
- Run the Workstation Installation program

  During the workstation installation, all objects and the Supported Local Database, which contains replicated data, are copied from the package to the workstation. The system copies objects in only those packages for which you included specifications. If the package does not include specifications, objects are not replicated; instead, they are installed on the workstation through just-in-time installation.

Unless otherwise noted, object movement is the same when you check objects in or out, add objects to a project, or perform a get object. The following table describes the objects and specifications that move when you check in, check out, add, or get each type of object:

Table (object type TBLE)	These objects move: • Table specs • Table event rule specifications (If the tables have event rules) • Source files (*.c) • Table header files (*.h) • Object files (*.obj) (If the objects have event rules) • Table event rule include files (*.hxx) (If the tables have event rules) The table header is not the same as the actual table that resides in a database. The table itself is created through Table Design Aid when you generate it. The Workstation Installation program copies this table to the workstation if the table is stored in the Supported Local Database.
Business view (object type BSVW)	• Specifications move. • .h files (if generated).
C business function (object type BSFN, source language C)	These objects move: • Specifications • Source files (*.c) • Header files (*.h) • Object files (*.obj)
NER business functions	Business function event rules (object type BSFN) can be checked out. When business function event rules are checked out, the .h file moves to the include directory, the .c file moves to the source directory, the .obj moves to the obj directory, and the local specifications are updated. These objects move: • Specifications • Source (*.c) • Header (*.h) • Object (*.obj)
Business function data structure (object type DSTR)	Specifications move.
Embedded event rules	You cannot check out embedded event rules. Embedded event rules move when you check out the object in which the event rule is embedded. For example, if embedded event rules are attached to a table, interactive application, or batch application, when you move the table or application, the specifications for the embedded event rules move with it.

Media object data structure (object type GT)	Data structure specifications move.
Interactive application (object type APPL)	These specifications move: • Application • Form • Form data structure • Embedded event rules
Batch application (object type UBE)	• Report and event rule specifications move. You must check in and check out the version separately from the report. • .h file (if generated by the developer).

See Also

EnterpriseOne Tools 8.94 PeopleBook: Object Management Workbench, "Getting Started with PeopleSoft Tools Object Management Workbench," PeopleSoft Tools OMW Overview

Performing Backups and Restoring Objects

You can back up development objects on workstations and servers as frequently as necessary.

It is recommended that you perform the following backups:

- Developer workstation backups: Developers at a certain company do not back up their directory to the server because of server-space concerns. Instead, they check in the objects they are working on every eight hours, or however often needed to avoid rework.

  Unless you have unlimited disk space on a file server to enable developers to back up their entire path code directory, you must use the check-in process as your backup method. If you follow the recommended development process, developers will know that they are enabled to check in unfinished or malfunctioning applications to the DEV path code.
- End-user workstation backups: End users should not have unreplicated data on their machines except store-and-forward transactions that have not yet been uploaded to the transaction tables. You should establish a policy that all store-and-forward users must upload their transactions before they leave work for the day. If a user is unable to perform this upload, he or she should back up the local database to a subdirectory on a file server. The name of the subdirectory should be the user's name.
- Development Server backups: At a certain company, the IT department backs up both the development file server (normally the deployment server) and necessary databases (central objects, Object Librarian, and data dictionary). When a developer needs to restore a particular object from backups, a database administrator restores the export to a path code called Restore. The developer checks out the object from Restore, ensures that the object functions as expected, and checks the object into the normal development path code.
- Deployment Server backups: In most cases, you do not need to back up the entire server nightly. However, under certain conditions, you might need to back up the following directories nightly:
- The DEV path code, if you are modifying objects, building new packages, or updating the database that is delivered during a workstation installation.
- MEDIA OBJ, if your media objects reside on the deployment server. Data sources in Oracle or SQL Server should be backed up nightly if your system data or any other important data is stored on the deployment server.

- Enterprise Server backups:
- Back up the DBMS nightly. It is recommended that you use the backup tool that your database vendor provides.
- Back up objects by backing up the entire directory. Also back up the PROD and DEV path codes and the JDE.INI file. Path codes are updated when the Version Control administrator deploys an object that was modified by developers who are authorized to access the Server Package application, and by end users who create new batch versions that will be executed on the server.

See Also

EnterpriseOne Tools 8.94 PeopleBook: Server & Workstation Administration, "Backing Up EnterpriseOne Tables," Backing Up an Enterprise Server

Replicating Data Dictionary Items

The data dictionary resides in relational tables on a server. This set of tables is the publisher data dictionary where you make all data dictionary changes that you want to replicate to servers and workstations. The system stores the publisher data dictionary in these tables:

- Data Item Master (F9200)
- Data Field Display Text (F9202)
- Data Item Alpha Descriptions (F9203)
- Data Dictionary - Error Message Information (F9207)
- Data Field Specifications (F9210)
- Data Dictionary - Smart Fields (F9211)
- Media Object storage (F00165)

Correlating Replicated and Central Objects

The following table describes the correlation between replicated objects that are stored in specification tables and central objects that are stored in a relational database that has a binary large object (BLOB).

Replicated Object	Central Object and Description
DDTABL	Table Header table (F98710). Contains one record for each table.
DDCLMN	Table Columns table (F98711). Contains one record for each column in a table.
DDPKEYH	Primary Index Header table (F98712). Contains one record for each table index.
DDPKEYD	Primary Index Detail table (F98713). Contains one record for each column in an index.
BOBSPEC	Business View Specifications table (F98720). Contains one record for each business view.

Replicated Object	Central Object and Description
GBRLINK	Event Rules - Link Table (F98740). Contains one record for each event that has event rules for applications, reports, or tables (Named Event Rule links are stored in the Business Function F9862).
GBRSPEC	Event Rules - Specifications Table (F98741). Contains one record for each line of event rules.
DSTMPL	Data Structure Templates table (F98743). Contains one record for each business function, processing option, form interconnection, report interconnection data structures, and power form.
FDATEXT	Forms Design Aid Text Information table (F98750). Contains override text for applications.
FDASPEC	Forms Design Aid Specification Information (F98751). One record for every column, grid line, button, hyperitem, control, and so on, in the application.
ASVRHDR	Forms Design Aid/Software Versions Repository Header Info. (F98752). One record for each application (if the application has processing options, that information is also stored in the record).
ASVRDTL	Forms Design Aid/Software Versions Repository Detail Info. (F98753). One record for each form (includes references to the data structures).
RDATEXT	Report Design Aid Text Information (F98760). Override text for batch reports.
RDASPEC	Report Design Aid Specification Info (F98761). One record for each section, column, sort, constant, and so on, in Batch Reports and Versions.
JDEBLC	JDEBLC - Behavior Information (F98762). One record for each function in BSFN.
CGTYPE	Stored in specification format only. Code Generator Form Types.
DDDICT: One record for each data dictionary item that has been just-in-time installed DDINDEX DDTEXT	One record for each data dictionary item that has been just-in-time installed obsolete in B73.2 - indexes for dictionary data dictionary text
GLBLTBL	Cache Information From Data Dictionary and Table Specifications. Contains runtime table and override information. Built dynamically the first time a table is used.
SMRTTMPL	Data structure required field information.

Replicated Object	Central Object and Description
F9200	Data Item Master
F9202	Data Field Display Text
F9203	Data Item Alpha Description
F9207	Data Dictionary - Error Message Information
F9210	Data Field Specifications
NEXTID	Next ID Master table (F98701). Local record of next IDs assigned to each workstation.

Modification Rules

This section discusses:

- Types of Modifications
- What an Upgrade Preserves and Replaces

Types of Modifications

Because the Development Tools are comprehensive and flexible, you can customize certain aspects of business solutions and applications without making custom modifications. This concept is referred to as *modless modifications*, which are modifications that you can perform easily without the help of a developer. You can perform modless modifications on:

- User overrides
- User defined codes
- Menu revisions
- All text
- Processing options values
- Data dictionary attributes
- Workflow processes

This flexibility improves efficiency and provides distinct advantages, such as the ability to:

- Export grid records to other applications, such as a Microsoft Excel spreadsheet
- Resequence a grid on a different column
- Change grid fonts and colors
- Control major system functions using processing options

Developers may need to modify the PeopleSoft EnterpriseOne software more extensively. To ensure that the modifications perform like modless modifications and to provide a seamless and predictable upgrade to the next release level, verify that any software modifications that you make comply with the recommended rules and standards.

To ensure a smooth upgrade, you should prepare for the upgrade before you make any custom modifications. If you plan modifications properly, you can minimize the tasks that you need to perform following an upgrade. Planning usually reduces the time required to upgrade your software, which reduces disruption to your business and the overhead cost of the upgrade.

The system tracks all custom modifications as you check them into the server. Before you perform an upgrade, you can run the Object Librarian Modifications report (R9840D) to see a list of the changed objects.

The system consists of control tables, such as menus, user defined codes, versions, and the data dictionary; and transaction tables, such as Address Book Master (F0101). The system provides control tables with data that you can modify, while transaction tables contain your business data.

During an upgrade, both sets of tables go through an automatic merge process. The system merges control tables with new data and converts transaction tables to the new specifications without change to your existing data. For the object specification merges (such as business views, tables, data structures, processing options, event rules, and applications), the system merges the specifications or replaces them, depending on the rules that are defined in the software.

What an Upgrade Preserves and Replaces

This section discusses the modification rules for these types of objects:

- Interactive Applications
- Reports
- Application Text Changes
- Table Specifications
- Control Tables
- Business Views
- Event Rules
- Data Structures
- Business Functions
- Versions

General Rules for Modification

If your business needs require custom modifications to the software, use these general definitions to ensure a smooth and predictable upgrade. These definitions describe which modifications the upgrade process preserves and which modifications it replaces.

- Preserves

  During an upgrade, the system automatically merges your modifications with the new applications that are shipped with the upgrade, and you do not lose your modifications. If a direct conflict exists between your specifications and system specifications, the upgrade process uses your specifications. When no direct conflict exists, then the upgrade process merges the two specifications.

- Replaces

  The upgrade does not merge your modifications with new applications and, therefore, the new software replaces your modifications. You must recreate your modifications after the upgrade completes.

Run the Object Librarian Modifications Report (R9840D) before the upgrade process to identify the objects that you modified.

These general modification rules apply to all objects:

- When adding new objects, use system codes 55 - 59.

  The system uses reserved system codes that enable it to categorize different applications and vertical groups. When you use system codes 55 through 59 for your custom modifications, the system does not overlay your modifications with new applications.

- Do not create custom or new version names that begin with ZJDE or XJDE.

  These prefixes are reserved for standard version templates that are included with the software, and these prefixes do not preserve your custom versions in case of a naming conflict. You can copy the pristine versions to create new templates or versions.

- For upgrades, build a package from the last modified central objects set and perform backups of your development server, central objects, and Object Librarian data sources so that you can access those specifications for comparison or for troubleshooting purposes.

Interactive Applications

Do not delete controls, grid columns, or hyper items on existing applications. Instead, hide or disable them. The system might use these items for calculations or as variables, and deleting them might disable major system functions.

The following table describes the interactive application elements that are preserved or replaced during an upgrade.

Object	Preserved	Replaced	Comments
New applications	X		You can either create an application from scratch, or copy an existing application using the Copy feature in Application Design Aid. This feature enables you to copy all of the application specifications, including event rules. If you use the Copy feature to copy an existing application for some modifications, during an upgrade your new application does not receive any changes that the system might have made to the original application.
New hyper items added to existing forms		X	
New controls added to existing forms		X	
New grid columns added to existing forms		X	

Object	Preserved	Replaced	Comments
Style changes		X	Style changes include fonts and colors. New controls have the standard base definitions. If you adjust the style, you need to also adjust the styles for any new controls that you added to an application.
Code-generator overrides		X	
Data dictionary overrides		X	
Location and size changes		X	In a subsequent release of the software, a new control might be placed in the same location that you have placed a custom control. In this case, the new control appears on top of your custom control. This situation does not affect the event rules or the functions of the application. After the upgrade, you can use Application Design Aid to rearrange the controls.
Sequence changes for tabs or columns		X	The upgrade process adds new controls to the end of your custom tab sequence. You can review the tab sequence after an upgrade.
Custom forms on existing applications		X	Instead of adding custom forms to existing applications, create a custom application using system codes 55 through 59, and then place the custom form on that custom application. You can then add to existing applications Form exits and Row exits that call your custom forms within your custom applications. System performance is not adversely affected when you call an external application from a row exit instead of from a form within the application.

Note. Starting with 8.94, none of the custom modifications to the PeopleSoft applications are preserved during the batch spec merge process. Instead, administrators must manually retrofit the modifications from an EnterpriseOne workstation with the help of Visual ER Compare and FDA Compare tools when the upgrade is complete.

Reports

The following rule applies to Report Design Aid specifications:

Do not delete objects on existing reports. Instead, hide them. The system might use these objects for calculations or as variables, and deleting them might disable major system functions.

The following table describes the report elements that are preserved or replaced during an upgrade.

Object	Preserved	Replaced	Comments
New reports	X		You can either create a report from scratch or copy an existing report using the Copy feature in Report Design Aid. This feature enables you to copy all the report specifications, including event rules. If you use the Copy feature to copy an existing report for some modifications, during an upgrade your new report does not receive any changes that updates might have made to the original report.
New constants added to existing reports	X		
New alphabetical variables added to existing reports	X		
New numeric variables added to existing reports	X		
New data variables added to existing reports	X		
New runtime variables added to existing reports	X		
New database variables added to existing reports	X		
New data dictionary variables added to existing reports	X		

Object	Preserved	Replaced	Comments
Style changes	X		Style changes include fonts and colors. New controls have the standard base definitions. If you have adjusted the default style, you need to also adjust the styles for any new controls that you added to a report.
Location and size changes for objects	X		In a subsequent release of the software, a new object, such as a control, might be placed in the same location as you placed a custom object. In this case, the objects display next to each other. This situation does not affect the event rules or the functions of the report in any way. After the upgrade, you can use Report Design Aid to rearrange the objects.
Data dictionary overrides	X		
Custom sections on existing reports		X	Instead of adding custom sections to existing reports, use Report Interconnect and connect to a new custom report that uses system codes 55 through 59. System performance is not adversely affected when you call a report through report interconnections.

Application Text Changes

This table describes the application text that is preserved or replaced during an upgrade.

Object	Preserved	Replaced	Comments
Overrides performed in Application Design Aid		X	Use the Visual Compare tools.
Overrides performed in Report Design Aid	X		
Overrides performed in Interactive Vocabulary Override		X	Use the Visual Compare tools.
Overrides performed in Batch Vocabulary Override	X		

Table Specifications

An upgrade merges your table specifications from one release level to the next.

The following table describes the table specification elements that are preserved or replaced during an upgrade.

Object	Preserved	Replaced	Comments
New Tables	X		
Custom Indices to tables	X		
Columns added to or removed from existing tables		X	This object includes changing field length, field type, and decimal position. Instead of adding a new column to an existing table, use a tag table with system codes 55 through 59.

For custom tag files, be aware of data item changes in the data dictionary. In subsequent releases, PeopleSoft might change certain attributes of a data item, such as its size, that might affect data integrity and how data is stored in the database. For this reason, you might need to use the Table Conversion tool to convert the tag file data to the new release level. For base files, the upgrade process automatically applies the data dictionary to the new release level. An upgrade preserves custom indices for the custom tag files.

Control Tables

Control tables contain user defined codes (UDCs), menus, and data dictionary items. An upgrade merges your control tables from one release level to the next using the change table process, which uses your control tables, not system tables, as the basis for the data merge.

The following table describes the control table elements that are preserved or replaced during an upgrade.

Object	Preserved	Replaced	Comments
Data dictionary custom changes	X		This object includes changes to row, column, and glossary text. The upgrade process uses your data dictionary as the base, and in case of a conflict with system data items, your changes override. Create new data items using system codes 55 through 59.
UDCs	X		The upgrade process merges any new, hard-coded values. (PeopleSoft-owned values are stored in systems 90 and above, and H90 and above.) The process also reports any hard-coded values that conflict with your custom values.

Object	Preserved	Replaced	Comments
Menus	X		In case of a conflict with base menus, your custom changes override.
Columns added or removed from existing control tables		X	

Business Views

Do not remove columns from existing business views. Changing business views that applications use can cause unpredictable results when you run the application. If you need to hide columns, do so within the application design using either Application Design Aid or Report Design Aid. Deleting a few columns from a business view does not significantly improve system performance.

The following table describes the business view elements that are preserved or replaced during an upgrade.

Object	Preserved	Replaced	Comments
New custom business views	X		
New columns, joins, or indexes added to existing business views	X		
Columns removed from business views		X	

Event Rules

During the upgrade process, the system checks for custom event rules that conflict with new event rules that the software installs. If a conflict exists, the system disables the custom event rules and appends them to the end of the new event rules.

The following table describes the event rule elements that are preserved or replaced during an upgrade.

Object	Preserved	Replaced	Comments
Custom event rules for custom applications, reports, and tables	X		
Custom event rules for custom business functions	X		
Custom event rules on a new custom control		X	Use the Visual Compare tools.
Events for system applications, reports, and tables that do not have any system event rules attached to the same event	X		

Object	Preserved	Replaced	Comments
Events for system business functions that do not have any system event rules attached to the same event	X		
Events for system applications, reports, and tables that have existing event rules attached to the same event		X	Use the Visual Compare tools.
Events for system business functions that have event rules attached to the same event		X	Use the Visual Compare tools.

To restore your custom event rules to system objects, highlight and drag the event rules back to the proper place in the event and enable them. Prior to an upgrade, perform these tasks:

- Run the Object Librarian Modifications report to identify modified applications
- Print the event rules for the modified application so that you can see the logic for the event when you restore custom event rules

Data Structures

The following table describes the data structure elements that are preserved or replaced during an upgrade.

Object	Preserved	Replaced
Custom forms data structures	X	
Custom processing options data structures	X	
Custom reports data structures	X	
Custom business functions data structures	X	
Custom generic text data structures	X	
Modifications to existing system forms data structures		X
Modifications to existing system processing options data structures		X
Modifications to existing system reports data structures		X

Object	Preserved	Replaced
Modifications to existing system business functions data structures		X
Modifications to existing system generic text data structures		X

To bring forward to the next release level the custom modifications that you made to system data structures, run the Object Librarian Modifications report (R9840D) to list all of the modified data structures, and use this report as a guide when you manually enter data structure changes.

Business Functions

For any new custom business functions (BSFNs), create a new (custom) parent DLL to store your custom modifications.

To bring your custom changes forward to the next release level, run the Object Librarian Modifications report (R9840D) to list all of the modified business functions, and use this report as a guide when you manually enter the business function changes.

Always use the standard API (jdeCallObject) to call other business functions from within a business function. Not following this and all other standards will cause problems.

To determine whether the source code of existing base business functions has been modified, use a third-party source-compare tool, such as Microsoft WinDiff. To determine modifications to APIs within business functions, see the online help feature for the most current information about APIs.

The following table describes the business function elements that are preserved or replaced during an upgrade.

Object	Preserved	Replaced	Comments
New custom business function objects	X		
Modifications made to existing system business function objects		X	NER BSFNs can be modified

Versions

For new custom versions, create a new version with a name that does not begin with XJDE or ZJDE.

The following table describes the versions elements that are preserved or replaced during an upgrade.

Object	Preserved	Replaced
Non-PeopleSoft versions	X	
Version specifications	X	
Processing option data	X	

Object	Preserved	Replaced
All ZJDE and XJDE version specifications		X
All processing option data for XJDE versions		X

In addition, processing option data is copied but not converted for non-PeopleSoft versions that use processing option templates. A warning is issued at runtime, and some data might be lost.

Also, event rule modifications for custom versions of PeopleSoft templates are not reconciled with the parent template.

Copying Records Between Data Sources

This section provides an overview of the process for copying control table records and discusses how to copy these records between data sources.

Understanding the Record Copying Process

You use the Object Management Workbench program (P98220) to copy control table records from one data source to another. However, to copy Solution Explorer records, you must use the Solution Explorer Record Copy program (P9864A). For example, you might need a separate set of UDCs, roles, or data dictionaries for your production and CRP environments. In this case, you can use the Record Copy program to copy each change that you make in the CRP environment to the production environment.

You can copy control table records in Solution Explorer from one data source to another data source. You can either copy all control table records, or you can copy the following records individually:

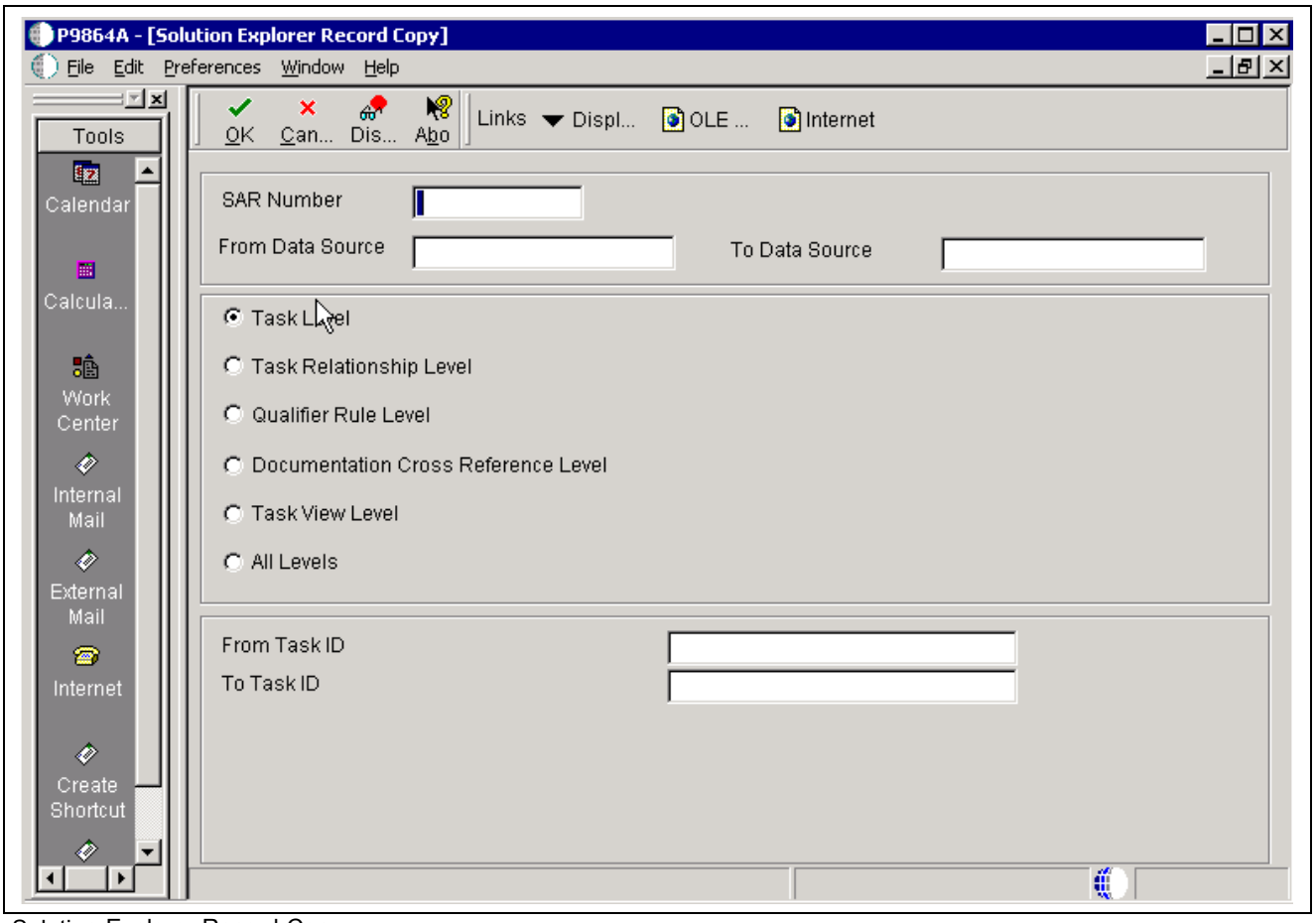
- Tasks
- Task relationships
- Qualifier rules
- Documentation indices
- Task views

Forms Used to Copy Control Table Records

Form Name	Form ID	Navigation	Usage
Solution Explorer Record Copy	W9864AA	From the Content Management task view, select Solution Explorer Record Copy (P9864A).	Copy Solution Explorer records between data sources.

Copying Control Table Records

Access the Solution Explorer Record Copy form.



Solution Explorer Record Copy

To copy records between data sources :

1. On Solution Explorer Record Copy, complete these fields:

SAR Number	Type the SAR number for the group of tasks that you want to move. If the tasks are not associated with a SAR, type an arbitrary number in this field.
From Data Source	Type the data source from which you are copying the records.
To Data Source	Type the data source to which you are copying the records.

2. Select the following options and then complete the relevant fields that appear at the bottom of the form.

   - Task Level

     Transfers a single task or a range of tasks. The system transfers all tasks in the range between the From Task ID and To Task ID. If you want to transfer a single task, enter the task ID in both the From and To fields.

     When you copy Solution Explorer tasks from the task level, the records are copied into all task relationships and task views in which they appear in the source data source.

- Task Relationship Level

  Transfers tasks that are associated with a parent task (task node). Complete these fields to transfer tasks at this level:

Parent Task	Type the parent task of the node or task that you want to transfer.
Child Task	Type a child task. If the child task is a node, the system transfers the node and all of its children.
Task View	Type the task view for which you want to copy this task relationship. The system copies the task relationship only to the task view that you specify, even if this task relationship exists in other task views.

- Qualifier Rule Level

  Transfers one or more qualifier rules. The system transfers all qualifier rules within the range that you specify.

- Documentation Cross Reference Level

  Transfers documentation index records that are associated with task IDs. Type a range of task IDs to transfer a range of documentation indexes. The system transfers all indexes within the range that you specify.

- Task View Level

  Transfers all tasks, task relationships, qualifier rules, and documentation indexes that are associated with a task view. When you transfer these records within a given task view, the system displays only those tasks within the view that you specify.

- All Levels

  Transfers all Solution Explorer tasks, task relationships, qualifier rules, documentation indexes, and variants. Check Clear To Data Source if you want to delete the contents in the target data source before you copy the new records from the source data source.

CHAPTER 3

Understanding the Package Build Process

This chapter discusses:

- The creation and deployment process
- Server packages
- Workstation packages
- Files created by the build process
- Features

The Creation and Deployment Process

This section discusses:

- How the system builds a full package
- How the system builds a full mobile package
- How the System Builds an update package
- How the System Builds a mobile update package

How the System Builds a Full Client Package

The following is an overview of how the system builds a full package:

1. Creates the package build directories.
2. Creates the INF file.
3. Copies these directories and files from the check-in location to the package name directory:
   - res
   - source (.c files)
   - include (.h files)
   - work
   - make
   - bin32
   - lib32
   - object (.obj files)

Note. If you select to build business functions with the package build, the system does not copy bin32, lib32, and object (.obj) files because the BusBuild program creates them.

4. Builds the table access management (TAM) specification files from the information in the relational database.
5. Runs the BusBuild program to compile and link the business functions that create the DLLs in the bin32 directory, the objects in the obj directory, and the libraries in the lib32 directory.
6. Compresses the directories.

How the System Builds a Full and Update Server Package

The following is an overview of how the system builds a full server package:

1. Creates the package build directories.
2. Creates the INF file.
3. Builds the TAM specification files using information in the pack files.
4. Copies the source (.c) and header (.h and .hxx) files to the server, then builds the business functions.
5. Compresses the package on the enterprise server.
6. Copies the compressed files from the enterprise server to the deployment server.

How the System Builds a Full Mobile Package

When a full package is marked as a mobile package, the package build process automatically creates both a full package, as described previously, and a full mobile package in the same directory structure. The mobile package has the same first eight digits in the name as the full package, appended with "_M." This naming scheme reduces the uniqueness of the package name from 10 to 8 characters. In addition to the folders listed previously for a full package, the package build process also creates two new folders for a full mobile package:

- mobilespec

  This directory contains the mobile spec files, which include only those applications and UBEs marked as "mobile."

- mobiledb

  This directory contains the mobile repository, which includes a complete set of EnterpriseOne tables based on the WAN environment.

How the System Builds a Client Update Package

The following is an overview of how the system builds an update package.

1. Creates the package build directories.
2. Creates the INF file.
3. For each object in the Package Build History table (F96225), retrieves the information from the relational database and adds it to the TAM specification files.
4. Copies the associated .c, .h, and .hxx files for the selected objects from the check-in location to the package build area.

5. Runs the BusBuild program to update the DLLs in the bin32 directory, the objects in the obj directory, and the libraries in the lib32 directory.

How the System Builds a Mobile Update Package

The process for creating a mobile update package is the same as that for creating a regular update package, and happens automatically if the select parent package was built as a mobile package.

Server Packages

This section discusses:

- Server packages
- The server package build process
- JDE.INI settings for server package builds
- Compressing Server Packages
- Source Code for Sun Servers

A description of server packages

All application development takes place on workstations. Object-related files are stored on a single deployment server, and specs are stored in the central objects database on the database server. Application development is managed by the Object Management Workbench. This configuration enables you to partition business applications to an enterprise server. To ensure that modifications and enhancements that are developed on the workstation are reflected on the server, you must build a server package that contains those modifications and enhancements.

Using the same Package Assembly (P9601) and Package Build Director (P9621) programs that enable you to create workstation packages, you can assemble, define, and build server packages, then push objects to enterprise servers. A server package is a group of specification records, source files, header files, and compiled objects that are created on the enterprise servers. After defining and building a server package, you can install it to enterprise, logic, or application servers by using the Deployment Director program (P9631).

In a development environment, developers can create server packages whenever they create or modify an object and need to transfer that object to the enterprise server. In a production environment, a system administrator should create server packages.

A server package is essentially the same as a client workstation package, with these exceptions:

- Server packages do not include help information or a Supported Local Database.
- Foundation code is not deployed as part of a server package.
- Some sets of TAM specifications, such as those used in form design (that is, interactive engine specifications), are not used on servers, and therefore are not included in a server package.
- Some business functions (such as client only business functions) are not built on the server, and therefore are not included in a server package.

Under the following circumstances, you should create a separate server package that corresponds to the workstation package:

- When business functions are not compatible across platforms. For example, a business function that is compiled on a Windows workstation will not run on a UNIX server. Also, not all business functions are compiled on the server.
- When data alignment is different across platforms. TAM files that are built on the workstation will run on the server only if you convert them. Also, integer representation (the way numbers are recognized and identified) is different on servers than on the workstation.
- When ASCII to EBCDIC conversion must take place for the iSeries.

In addition, server platforms are not always compatible. For example, each enterprise server platform has its own internal storage for specifications, which are not compatible across platforms. Also, each enterprise server platform uses different compilers and has different object representation, so business functions are not compatible across platforms.

The Server Package Build Process

Although creating a server package is identical to creating a client workstation package, you create them for different purposes. The main purpose for building a server package is to enable administrators to transfer TAM specifications and business functions to the enterprise servers.

The Package Build Director program (P9621) provides these benefits for building server package builds:

- Provides complete integration with client workstation packages
- Enables administrators to build a package on multiple servers simultaneously
- Enables administrators to build individual package components simultaneously on the server
- Enables you to build a package on one enterprise server and deploy to another server of the same type
- Creates history records that enable monitoring from the client workstation, so that the administrator can determine which packages have been built
- Creates compressed files and loads them onto the deployment server for easier mastering to a CD or for deploying to another server of the same type
- Supports both full and update packages
- Provides restart capabilities for packages that do not build successfully

You can install these objects on an enterprise server:

- Business functions
- Business views
- Data structures
- Tables (installation does not create the table in a database; it only pushes the specifications and table header files to the server)
- Batch application specifications (both templates and versions)
- Application specification records

Because server packages are assembled and defined in the same way as client workstation packages, you can assemble a server package, using the Package Assembly program (P9601), and build the server package, using the Package Build Director program (P9621), at the same time that you assemble and build client workstation packages.

After you assemble the server package and define the package build, these events occur:

1. After the build process starts, the system creates packed TAM files. These are specification files in a platform independent format.
2. On the build machine, the system starts a batch application for the server package.
3. This batch application calls a business function, which, in turn, calls the server package build engine.
4. The build engine uses the records that the Package Assembly and Package Build Director programs create to:
   - Initialize directories on each server.
   - Begin transferring packed TAM files. As each packed file is transferred, an unpack process starts on the server. This process converts the specification files to a platform dependent format.
   - Transfer all business function source and header files to the server. At this time, the build engine reads the Object Librarian Master Table (F9860) to determine the DLL to which each module belongs. For the JDBTRIG library, a special function is called to direct the trigger library to which the module belongs. In this case, the system does not use the Object Librarian Master Table.
   - Start a build master process on the server when the source files for all business functions are transferred. This build master starts one or several individual build processes simultaneously. Each DLL has its own build process. The JDE.INI file indicates the number of processes that can run simultaneously.
   - Move the process to another server, if one was specified during the package build process. The process transfers and builds all components on that server.
   - Check the status of each build piece on each server after the build process has begun on all servers. History records are updated as the statuses change.
   - Compress the package components and transfer the compressed files back to the deployment server when the building is complete. This happens only if you specify that the system compress the files when it builds the package. This process is repeated for all servers.

JDE.INI Settings for Server Package Builds

If your server package includes business functions, the BSFN BUILD section of the JDE.INI file applies to the package.

Setting	Value	Purpose
Build Area=	/usr/PeopleSoft/E811/packages	Indicates the location on the server where the package will be built.
Optimization Flags=	+02 (default for HP 9000) -02 (default for RS/6000 and Sun)	Varies, depending on the platform. The system uses these compile flags to build business functions in Release mode. You should not change these flags.
DebugFlags=	-g -y -D_DEBUG _DJDEDEBUG (default for HP 9000) -g -qfulpath -qdbextra -D_DEBUG -DJDEDEBUG (default for RS/6000) -g -D_DEBUG -DJDEDEBUG (default for Sun)	Varies, depending on the platform. The system uses these compile flags to build business functions in Debug mode. You should not change these flags.

Setting	Value	Purpose
InliningFlags=	blank (default)	Indicates whether the iSeries uses inlining. Enter Yes to select inlining on the iSeries. Enter No to turn it off. This flag is blank or ignored for non-iSeries servers.
DefineFlags=	-DKERNEL -DPRODUCTION_ VERSION -DNATURAL_ ALIGNMENT -D_HPUX-SOURCE (default for HP 9000) -DKERNEL -DPRODUCTION_ VERSION -DNATURAL_ ALIGNMENT (default for RS/6000) -DKERNEL -DPRODUCTION_ VERSION -DNATURAL_ ALIGNMENT -D_SUN-SOURCE (default for Sun)	Indicates the Kernel Production Version of the source for HP, RS, and SUN.
CompilerFlags=	-Aa +w1 +z -c (default for HP 9000) -qalign=natural -qflag=I:I -c (default for RS/6000) -qspill=1024 -misalign -KPIC (default for Sun)	Varies, depending on the platform. The spill flag sets the stack space when business functions are compiled. Typically, 1024 is adequate space to compile the delivered business functions.
OSReleaseLevel=	+DAportable -q32 (for AIX)	Indicates the release level for which you are compiling the package. You should not change these flags.
LinkFlags=	-b -z (default for HP 9000) -bl:/<your system directory>/bin32 /functlist.imp -bM:SRE -bexpall -brtl -lc -bnentry -L. L/usr/<your system directory>/lib -ljdelib -ljdekrnl -ljdenet -bloadmap:loadmap (default for RS/6000) -G -L$(ORACLE_HOME)/lib (default for Sun)	Varies, depending on the platform. The system uses these flags to link business functions. You should not change these flags.
LinkLibraries=	blank (default)	Indicates the libraries to which business functions are linked. (Applies to Windows and iSeries servers only.)

Setting	Value	Purpose
SimultaneousBuilds=	0 (unlimited) (default) any integer (number of simultaneous builds)	Indicates the number of DLLs that can be built at a time. Zero means that all will be built simultaneously.
Qname=	<queue name>	Applies to AS400 only. Specify a queue name if you want the jobs for building dlls to go to a queue other than the default queue.

Compressing Server Packages

To compress packages that you build on the server, add the [BSFN BUILD] section to the jde.ini file on the workstation and create the following entry:

```
DoCompression=1
```

This setting compresses packages that you built on the server for deployment later to other servers of the same type, such as all AIX Unix servers or NT servers. If you plan to deploy a package to an enterprise server of a certain type, the package must be built on the same type of server.

When the server builds a compressed package, it stores the compressed files in subdirectories, such as \bin32, \specs, that are subordinate to the following path on the deployment server: \package*package name**server type*\, where *package name* is the name of the package, and*server type* is the type of server for which the package is compressed. The compression process creates a new file called compressed.inf in the *server type* directory. This file includes the information that the system needs to deploy the compressed files. The following table describes the type of compressed files that the process creates for each type of server:

NT	UNIX	AS/400
.cab	.z	SRVPG, SPECS, MODULE, USRSPC

When you deploy the package to another enterprise server, the system reads the compressed.inf file and uses this information to copy the compressed files from the package directory on the deployment server to the enterprise server.

Source Code for Sun Servers

The Sun Solaris compiler expects a newline character at the end of every source code file that it compiles. If the compiler does not find this newline character, it rejects the line and displays a warning message. In some cases, this line rejection and message might cause the package build to fail.

If you develop custom modifications on Sun servers that use the Solaris operating system, you must ensure that this newline character is present in the compiled source code before you assemble, define, and build packages that contain your modifications. This step helps ensure that the package build process completes successfully.

In some cases, the system automatically adds the newline character, and you do not need to add it manually. If you edit source code in the UNIX environment using an editor such as VI or emacs, these editors automatically add the newline character. Also, all of the source code files for business functions include the newline character.

However, editors that are included with PC workstations typically do not add the newline character. Therefore, if you edit source code on a PC workstation and then transfer the file to the server for compiling, verify that the newline character exists in the source code.

See Also

EnterpriseOne Tools 8.94 PeopleBook: System Administration, "Understanding the Jde.ini File Settings," How to Use This Section

Workstation Packages

This section discusses

- Workstation installation
- Building specifications and business functions
- Package INF files

Workstation Installation

A typical full workstation installation takes more than 1.4 GB of disk space and can take 10 to 30 minutes to install, depending on network traffic. A workstation configuration contains the full suite of applications, including those that the user rarely or never uses, but all applications are available immediately.

Building Specifications and Business Functions

If you build a full client package that include both business functions and specifications, add the following setting to the [INSTALL] section of the workstation jde.ini file on the computer that you use to build the packages:

```
WaitForBusbuild=Y/N
```

Y - business functions are built after all the specs are complete. The system builds the specifications and business functions sequentially instead of simultaneously.

N - specs and business functions are built simultaneously, which can speed up the build process.

Package INF Files

The Package INF file is essentially the interface between the package build and the Workstation Installation program. The INF file defines the components included in the package, the source and destination paths for those components, and the attributes needed to install the package.

The INF file is created during the package build process and is stored in its own package_inf directory, based on the release master directory. Workstation Installation reads the INF file for the package it is installing to determine which components are loaded to a workstation, as well as their locations.

Following is a typical INF file for package PD811FA, which is full package A for the PD811 path code. This INF file includes these sections:

- [Attributes]
- [SrcDirs]
- [DestDirs]
- [Filesets]
- [Components]

- [Typical]
- [Compact]
- [MSDE]
- [Features]

[Attributes]

This section contains information about the current release, global tables, specification and help files, and the JDE.INI file.

Item	Purpose
PackageName=STEST	Indicates the name of the package.
PathCode=	Indicates the path code for which the package is being built.
Built=	Indicates the package status. A status of 50 or 70 means that the package is ready for installation.
PackageType=Full	Indicates the package type: full or update.
Release=	Indicates the current release, which determines which setup.inf file to use in building the jde.ini for the workstation. The release is also used to determine paths for system and helps.
SystemBuildType	Indicates the type of build: DEBUG or RELEASE. This option is retrieved from owver.dll.
MFCVersion	Indicates the version of the MFC compiler.
SpecFilesAvailable=Y	Indicates that specification files are available to merge or copy. This option is always set to Y for full packages. For update packages, this option is set to Y only if objects are included in the package.
DelBlblTbl=Y	Indicates whether to delete global tables when installing. This option is set to Y for full or packages. For update packages, this option is set to Y only if the objects include a table object.
ReplaceIni=Y	Indicates whether to delete the existing jde.ini file when installing, and then create a new one. This option is set to Y for full packages. For update packages, the user must specify during package build definition whether to replace the jde.ini file.
AppBuildDate=Mon Jul 20 11:22:22 2005	Indicates the date and time when the package was built. Full packages always have this date. This option is blank when no objects are included in the package.

Item	Purpose
FoundationBuildDate=Wed Jun 03 15:08:34 2005	Indicates the date and time when the foundation was built. For 8.10 and later releases, this date is retrieved from owver.dll. Full packages always have this date. This option is blank when no foundation location is specified in the package.
DataBuildDate=Wed Jun 03 15:08:34 2005	Indicates the date and time when the database file was built. Full packages always have this date. This option is blank when no database location is specified in the package.
DeploymentServerName	Indicates the computer name of the deployment server.
Location	Indicates the location of the computer (the deployment server).
DeploymentStatus=	Indicates the package deployment status.
PackageDescription=	Indicates the package description.
Icon Description	Describes the desktop icon.
Default Environment	Indicates the default environment.
Spec	Indicates the name of the spec file that is specified in the file set.

[SrcDirs]

The Workstation Installation program uses these settings to determine the source path from which information is copied. A package build compresses these directories. Workstation Installation copies the compressed directories to workstations.

Item	Purpose
SPathcode=\\MachineName\ PeopleSoft\E811\PACKAGE \ PD811FA	Indicates the location of the package that the package builds for Workstation Installation. The default value for this path is the path code directory over which you built the package. You can change this setting if you want to use a different package.
SSYS=\\MachineName\ PeopleSoft\E811\SYSTEM	Indicates the location of the system directory that the package builds for Workstation Installation. The default value for this path is the system directory that is associated with the path code over which you built the package. Normally, this directory is subordinate to the release share name (811). This item appears only when included in the package.
SPathcodeDATA=\\MachineName\ PeopleSoft\E811\ PD811\PACKAGE\DATA	Indicates the location of the Supported Local Database that the package builds for Workstation Installation. The default value for this path is the data directory that is subordinate to the path code over which you built the package. This item appears only when included in the package.

[DestDirs]

The Workstation Installation program uses these settings to determine the destination paths on the workstation. The process replaces %INSTALL with the user's computer configuration that is set up in the User Display Preferences table (F00921) and the User Defined Codes Language Status table (F00051).

Item	Purpose
DPathcode=%INSTALL\path code	Indicates the destination directory for the package.
DSYS=%INSTALL\system	Indicates the destination directory for system files. This item appears only when included in the package.
DPathcodeDATA=%INSTALL\path code\data	Indicates the destination directory for the database. This item appears only when included in the package.

[Filesets]

These settings list the various source and destination directories that are subordinate to the path code for the package. Y=compressed, and N=not compressed. The source and destination directory names are preceded by an S and D, respectively.

Item	Purpose
Pathcode1=Y, $Spathcode\bin32, $Dpathcode\bin32	Indicates the bin32 directory that is subordinate to the path code.
Pathcode2=Y, $Spathcode\res, $Dpathcode\res	Indicates the res directory that is subordinate to the path code.
Pathcode3=Y, $Spathcode\spec, $Dpathcode\spec	Indicates the spec directory that is subordinate to the path code.
Pathcode4=Y, $Spathcode\include, $Dpathcode\include	Indicates the include directory that is subordinate to the path code.
Pathcode5=Y, $Spathcode\lib, $Dpathcode\lib	Indicates the lib directory that is subordinate to the path code.
Pathcode6=Y, $Spathcode\obj, $Dpathcode\obj	Indicates the obj directory that is subordinate to the path code.
Pathcode7=Y, $Spathcode\source, $Dpathcode\source	Indicates the source directory that is subordinate to the path code.
Pathcode8=Y, $Spathcode\work, $Dpathcode\work	Indicates the work directory that is subordinate to the path code.
Pathcode9=Y, $Spathcode\make, $Dpathcode\make	Indicates the make directory that is subordinate to the path code.
PathcodeDATA=Y, $SpathcodeDATA\data.CAB, $DpathcodeDATA	Indicates the compressed database that the package build generates.

[Components]

These settings indicate the location of the foundation, production, and development objects, as well as the database files.

Item	Purpose
ProdObj=APPL_PKG1, APPL_PKG2, APPL_PKG3	Indicates the location of production objects.
DevObj=APPL_PKG4, APPL_PKG5, APPL_PKG6, APPL_PKG7, APPL_PKG8, APPL_PKG9	Indicates the location of development objects.
Foundation=SYS	Indicates the foundation location.
Data=<pathcode> DATA	Indicates the database location.

[Typical]

This section describes the setting for a typical development user.

Item	Purpose
Name=Development	Indicates that the package is for a development user.
Description=Install the development objects	Indicates the package description.
Components=ProdObj, DevObj, Foundation, Data	Indicates that the package should contain both production and development objects, as well as the database and foundation.

[Compact]

This section describes settings for a typical production user.

Item	Purpose
Name=Production	Indicates that the package is for a production user.
Description=Install the production objects	Indicates the package description.
Components=ProdObj, Foundation, Data	Indicates that the package should contain only production objects, as well as the database and foundation.

[MSDE]

This section contains information about the local data source. The name of this section corresponds to the local data source name in the ODBC data sources section.

Item	Purpose
Driver=jdboledb.dll	Indicates the name of the driver.
DefaultDir=C:\ACCESS	Indicates the default directory.

Item	Purpose
UID=sa	Indicates a required attribute setting for Supported Local Database.
Driver32=jdboledb.dll	Indicates the ODBC driver for the Supported Local Database.
DBQ=$DAPPL_PGFDATA\ JDELocal.mdf	Indicates the path code for the Supported Local Database.
MaxBufferSize=2048	Indicates the maximum size of the buffer.
Threads=1028	Specifies the number of threads.

[Features]

This section describes information for any features that are included in the package. A feature is a set of files or configuration options that is included in a package for deployment to a workstation or server.

Item	Purpose
FeatureName=SFEAT	Indicates the name of the feature in the package.
Feature.inf location=	Indicates the location of the feature.inf file for the feature.

Files Created By the Build Process

This section describes the files that the system creates during the build process for the workstation and the various server platforms.

Workstation Package Build

Business function dynamic linked libraries (DLLs) on workstations are grouped by related business functions. This grouping limits the size and number of procedures that are contained in each DLL. Grouping prevents memory allocation errors and avoids the platform limitations that can occur when you export too many procedures from the same DLL.

The production environment PD811/bin32 directory contains the DLLs that are created on the workstation. All of the business function source files are in the PD811/source directory.

Files Created by a Business Function Build

When you build a single business function through the Object Librarian, the Business Function Builder program uses the make (*.mak) file that is generated at runtime and creates or copies the following files and builds the business functions into their respective DLLs:

- Source file (*.c)
- Header file (*.h)
- Object file (*.obj)

You must use the jdecallobject API to call a business function from a business function.

These files are created for named event rules business functions:

- OBJNAME.c
- OBJNAME.h
- OBJNAME.obj

These files are created for table event rules:

- OBJNAME.c
- OBJNAME.hxx
- OBJNAME.obj

Server Package Builds

Server package builds are used to move path code objects from the deployment server to enterprise server platforms. Server package builds are initiated by creating either full or update packages during package assembly. After you have assembled the package, you must select the server option during package definition, and select the relevant servers from the list of available servers in the screen that follows. Once package definition is complete and the package has been activated, highlight the package and select Submit Build from the Row menu to start the server package build. When the server package build has completed successfully, you can deploy the server package to activate path code objects.

To assemble a server package, use the foundation, database, and object information in package assembly to generate build information, specification files, and business function source for .c, .h, and .hxx files to place into a staging area assigned during package assembly. After the server package build has generated these objects and placed them in the staging area, the system transfers the objects to each of the servers specified in the package definition. Server package build then directs the servers receiving the server packages to compile the business function source code and generate the corresponding business function DLLs.

UNIX Server Build

This topic describes the files that the system creates when it builds business functions on a Unix server.

Files Created by a Business Function Build

When building business functions, the following groups of source files are actually compiled:

- Named Event Rules (NER) business functions
- Table event rules
- C business function event rules

When building business functions, the following file types are supplied to the build process:

- Source files (.c)
- Header files (.h, .hxx)

When building business functions, the build process creates these file types:

- Object files (.o)
- Make files (.mak)
- Shared libraries (.sl, .so)

Shared libraries for business functions, which are equivalent to a DLL for a Windows workstation, are consolidated. Therefore, one shared library is created for each parent DLL in the Object Librarian - Status Detail table (F9861). If you are creating custom business functions, use a custom parent DLL instead of one of the parent DLLs that PeopleSoft EnterpriseOne provides.

Where Business Functions Are Stored

On UNIX platforms, related business functions are grouped into shared libraries. This grouping limits the size and number of procedures that are contained in each shared library. Grouping prevents memory allocation errors and avoids platform-specific limitations in the number of procedures that you can export per shared library. Subordinate to the environment PD811 directory is a source directory. This source directory contains subdirectories for each shared library that is created on the enterprise server. The directory structure looks like the following:

PD811

—source

——CAEC

——CALLBSFN

——CCORE

——CDESIGN

——CDIST

——CFIN

——CHRM

——CMFG

——JDBTRIG

—bin32

Each subdirectory contains the business function source files that belong to the shared library. All shared libraries are installed in the PD811/bin32 directory. The naming convention for the shared libraries is lib, followed by the name of the shared library subdirectory, followed by .sl (for HPUX) or .so (for AIX). For example, libccore.sl.

Specification Files

Specification files must be consolidated into individual pack files before you transfer them to a server for translation. A specification file is a collection of two files, grouped by the same name (RDATEXT, for example). The two files have different extensions (.ddb for the data file, and .xdb for the index file), and together these files comprise a complete specification file.

Windows Server Build

This topic describes the files that the system creates when it builds business functions on a Windows server.

Files Created by a Business Function Build

When building business functions, the following groups of source files are actually compiled:

- Business function event rules

- Table event rules
- C business function event rules

When building business functions, the following file types are supplied to the build process:

- Source files (.c)
- Header files (.h, .hxx)

When building business functions, the build process creates these file types:

- Object files (.o)
- Make files (.mak)
- DLLs (.dll)

Business function DLLs are consolidated, just as they are on the UNIX platform or workstation. Therefore, one shared library is created for each parent DLL in the Object Librarian - Status Detail table (F9861). If you are creating custom business functions, use a custom parent DLL instead of one of the parent DLLs that PeopleSoft EnterpriseOne provides.

Where Business Functions Are Stored

On Windows platforms, business function are grouped into parent DLLs for related business functions. This grouping limits the size and number of procedures that are contained in each DLL. Grouping also prevents memory allocation errors and avoids platform-specific limits of exported procedures per DLL.

Subordinate to the environment PD811 directory is a source directory. This source directory contains subdirectories for each DLL that is created on the enterprise server.

The directory structure looks like the following example:

PD811

source

CAEC

CALLBSFN

CCORE

CDESIGN

CDIST

CFIN

CHRM

CMFG

JDBTRIG

Each subdirectory contains all of the business function source files that belong to the DLL. All DLLs are installed in the PD811\bin32 directory. They have the same name as the DLL subdirectories, except that they have the .dll suffix.

Specification Files

Specification files must be consolidated into individual files before you transfer them to a server for translation. A specification file is a collection of two files, grouped by the same file name (RDATEXT, for example). The two files have different extensions (.ddb for the data file, and .xdb for the index file), and together these files comprise a complete specification file.

iSeries Server Build

This topic describes the files that the system creates when it builds business functions on an iSeries server.

Files Created by a Business Function Build

When building business functions, the server package build creates the following file types in a library with the package name in the QSYS file system:

- *MODULES - Object files
- *USRSPC - User spaces hold information about which .c files each business function DLL contains
- *SRVPGM - Server programs are the DLLs on the iSeries
- *FILE - Contains only logs about compiled business functions

Where Business Function Source Members Are Stored

The business function source and header locations are standardized across all server platforms when performing server package builds. iSeries business function source and headers are now transferred to the Integrated File System (IFS). Server package build transfers objects to the following subdirectories under the server package directory in the IFS for the iSeries.

PD811

include

pack

source

CAEC

CALLBSFN

CCORE

CDESIGN

CDIST

.

spec

text

`PD811/include` - This is the location where .h and .hxx source files are located. These objects are taken from the server and built on.

`PD811/pack` - The pack directory contains pack files used to move specifications from the deployment server to the enterprise server.

`PD811/source` - Contains subdirectories that include the business function DLL names. Each subdirectory contains .c source for the business functions that are compiled and linked into the DLL.

`PD811/spec` - Contains specification files for the path code. A specification file is a collection of two files grouped by the same name (RDATEXT for example). The extensions .xdb and .ddb describe a specification file.

`PD811/text` - Contains build text and status files (.txt and .sts) for business function DLLs and specification files. The text files contain information needed for the server package build. The text files also contain build directives for creating business function DLLs. The status files for specification files indicate if a server package build was successful in converting pack files into spec files. The status files for business function DLLs indicate which .c source files were successfully compiled and linked.

Note. After an upgrade, existing iSeries server path codes must be rebuilt with the server package build to avoid problems building server package updates and manually re-linking business functions using the LINKBSFN program.

Specification Files

Specification files must be consolidated into individual files before you transfer them to a server for translation. A specification file is a collection of two files, grouped by the same file name (RDATEXT, for example). The two files have different extensions (.ddb for the data file, and .xdb for the index file), and together these files comprise a complete specification file.

Features

In addition to objects, you can also add a *feature* to a package. A feature is a set of files or configuration options that must be copied to a workstation or server to support an application or another function. Like objects, features are included in a package and deployed to the workstations and servers that require the feature components.

For example, you might need to add to a package ActiveX controls, a Supported Local Database for the Sales Force Automation feature, ODBC data sources for use with Open Data Access, or Microsoft Windows registry settings.

You define a feature by using the Package Assembly program (P9601). You can then add the feature to a package by using the Package Assembly program (P9601) and the Package Build Director program (P9621).

Feature INF Files

When a package contains features, a section called [Features] in the Package INF file includes both the feature name and a pointer to the Feature INF file that is created for each feature in the package. These Feature INF files provide specifications that tell the installation program the actions to perform during the installation.

The Feature INF file can include these sections:

- [Header]
- [Registry]
- [INI]
- [FileSets]
- [Shortcut]
- [ThirdPartyApps]
- [ODBCDataSources]

The following is a typical Feature INF file for which the sections contain specifications for each feature component.

[Header]

The header section contains general information about the feature and specifies the installation options for the feature.

Item	Purpose
Feature=	Name of the feature.
FeatureType=	Type of feature.
Description=	Text description of the feature.
Required=	A setting that indicates whether installation of the feature is required.
InitialChoice=	A setting that specifies the default selections for features that the user has the option to install.

The Required and InitialChoice entries are set using the three Feature Installation option settings (Required, Selected, Deselected) on the Feature Information form. When you select one of these three options, the system writes the following values into the Required and InitialChoice entries in the feature inf file.

Feature Installation Option	Required	InitialChoice
Required	Y	Both
Selected	N	Both
Deselected	N	Custom

[Registry]

This section contains information about how the feature affects the Windows registry.

Registry[no.]=Root[value],Key,[prefix]Name,[prefix]Value

The following table indicates the specifications for the Windows registry settings for the features:

Item	Purpose
Root	Describes the root in the registry with these values: • 0 means root • 1 means current user • 2 means local machine location • 3 means users
Key	Indicates the key for the registry value.
Name	The registry value name. Name prefixes are: • + means that the name is created (if it does not already exist) when the feature is installed • - means that the name is deleted with all subkeys when the feature is uninstalled • * means that the name is created (if it does not already exist) when the feature is installed, and it is removed with all subkeys when the feature is uninstalled
Value	The name of the registry value. Value prefixes are: • #x means that the value is stored as a hexadecimal value • #% means that the value is stored as an expandable string • # means that the value is stored as an integer • #$ means that the value is stored as a string

[INI]

This section contains information about how the feature affects the JDE.INI file.

Ini[no.]=FileName,Directory,Section,Key,Value,Action[value]

Item	Purpose
FileName	The name of the destination INI file.
Directory	The location of the destination INI file.
Section	The name of the section in the destination file.
Key	The name of the key within the section of the destination file.

Item	Purpose
Value	The value to be written to the key of the destination file.
Action	The action to take regarding the INI entry: • 0 means create the INI entry. • 1 means create the INI entry only if it does not already exist. • 3 means create the INI entry or append to the existing entry.

[FileSets]

This section contains information about additional files that must be installed for the feature to function correctly.

Fileset[no.]=Compression, SourceDirectory, FileName, TargetDirectory

Item	Purpose
Compression	An option that indicates whether the fileset is compressed.
Source Directory	The source location of the fileset.
FileName	The name of the CAB file for the fileset.
Target Directory	The target location into which the fileset will be placed.

[Shortcut]

This section contains information about shortcuts that appear on the Windows desktop as part of the feature installation.

Shortcut[no.]=Directory,Name,Target,Arguments,Description,HotKey,Icon,IconIndex,ShowCmd [value],WkDir

Item	Purpose
Directory	The directory where the shortcut is created.
Name	The name of the link file for the shortcut.
Target	The name of the executable file for the shortcut.
Arguments	Any command line arguments for the shortcut.
Description	A description of the shortcut.
HotKey	A hot key that launches the shortcut.
Icon	The shortcut icon and location.
IconIndex	An index of the icon if the icon is inside an image list.

Item	Purpose
ShowCmd	A command for the application window, with these value options: • 0 means show the window normal-sized. • 3 means show the window maximized. • 7 means show the window minimized; not active.
WkDir	The working directory for the shortcut.

[ThirdPartyApps]

This section contains information about third-party products that are installed with the feature.

ThirdPartyApp[no.]=Source Directory, Description, Synchronous/Asynchronous, FileName

Item	Purpose
Source Directory	Source location of the executable for running the third-party application.
Description	Description of the third-party application.
Synchronous/Asynchronous	An option that indicates whether the third-party application can be installed in parallel (synchronous) or must be installed serially (asynchronous).
FileName	The name of the file that launches the third-party application.

[ODBCDataSources]

This section contains information about ODBC data sources that are installed with the feature.

ODBC data sources have 2 sections in the feature INF. One section contains header information and the other contains the detail information. The feature.inf contains one header section listing all data source components that are included in the feature. For each data source listed in the header, a corresponding detail section exists. Only the header section is described in the following table. For information about the detail section, see the documentation for the selected ODBC Driver.

DataSourceName=DataSourceDriver

Item	Purpose
DataSource Name	The name of the ODBC data source.
DataSource Driver	The driver used for the data source.

CHAPTER 4

Assembling Packages

This chapter provides an overview of the Package Assembly process and discusses how to:

- Verify a Path Code for Package Assembly
- Assemble Packages
- Revise an Existing Package
- Activate an Assembled Package

Understanding the Package Assembly Process

The first step in building and deploying a package is to assemble the package, which entails selecting the type of package that you want to build, entering a package name and detailed description, and assembling the objects, data, foundation, and features that you want to include in the package. The package name and description appear during workstation installation when the user selects a package to install.

The Package Assembly Director, which you access from the Package Assembly program (P9601), steps you through the process. During package assembly, the build status is always either In Definition or Definition Complete. After you assemble the package, you can then define its build process.

Understanding the Package Assembly Director

The Package Assembly Director guides you through the process of specifying or confirming the location where package components can be found, as well as indicating the objects to include in the package. The director always gives you the option to either continue to the next form or go back to the previous form. Also, you can always cancel the assembly process.

You can enter default information on each of the main forms of the Package Assembly Director, or from each of the main forms you can access sub forms to customize the information. The steps are the same whether you are adding components for the first time or revising a previously-assembled package.

You can also access any previously assembled packages and view information about these packages by clicking the plus (+) symbol of the package on the Work with Packages form. For any previously assembled packages, underneath the package name you can view the package properties (including package type and current status), as well as the selections for foundation, database, help, and language.

When you finish adding the selected components to a package, those components appear on the specific form for that component, the Package Component Revision form, and the Work with Packages form of the Package Assembly Director.

The following table summarizes the function of each form in the Package Assembly Director:

Package Assembly Directory form	Use this form to review introductory information about the Package Assembly Director.
Package Information form	Use this form to enter the package name, description, and corresponding path code.
Package Type Selection form	Use this form to indicate whether you are creating a full or update package. When you create an update package, you must also indicate the parent package on which the update package is based. For example, if you are creating a package to update your original package called APPL_B, you would enter APPL_B as the parent package for your update package. You can also include object specifications in an update package. If you do not include specifications, the application is installed through just-in-time installation the first time that a user selects that application. That is, the system automatically retrieves the application specifications from the central objects data source the first time that a user selects the application after receiving the package.
Foundation Component form	Use this form to enter the location of the foundation. A foundation is the code required to run all applications. It is required for all full packages. If you do not specify a foundation path for full packages, the system uses the default foundation path. Update packages use the foundation for the parent package unless you specify another foundation.
Help Component form	This form is now obsolete and it's functionality has been disabled.
Database Component form	Use this form to specify the location of the database to be included in the package. For full packages, if you do not specify a database location, the system uses the default database path. Update packages do not require a database.
Default Object Component form (for full packages only)	Use this form to verify the deployment data source. When you build a full package, the system retrieves the objects that are included in the package from the deployment data source that is associated with the path code that you specified for the package.

Object Component form (update packages only)	Use this form to specify the individual objects that you want to include in the package. You can add any of these objects: • Interactive or batch applications • Business functions • Business views • Data structures • Media object data structures • Table definitions
Features Component form	Use this form to include features in your package. A feature is a set of files or configuration options, such as registry settings, that must be copied to a workstation or server to support an application or another feature.
Language Component form	Use this form to include in your package language specifications for a language other than English.
Package Component Revisions form	Use this form to review the information that you entered on the previous forms. You can modify any or all of your selections on this form.
Mobile Client Database Revisions form	Use this form to build a mobile client package and to select the mobile client databases.

Accepting Default Values

Many of the forms in the Package Assembly Director have default values and, after verifying that you want to use the default value, you can advance to the next form without entering anything.

Forms determine the default values based on these criteria:

- Foundation:

  The default foundation location is the server share path under the path code for the package.
- Database:

  The default database location is the server share path under the path code for the package.
- Objects:

  The default location for full packages is the deployment data source.
- Language:

  The default language is English.

On forms that have a default value, even if you change or clear the field you can always restore the original default value by clicking the Default button. The Form menu of the Package Component Revisions form also has a Set Default option that restores default values.

If you are building a full package and do not need to specify the objects in that package, the fastest way to define the package is to accept the default locations for the foundation, database, help, and language. This method applies only to full packages. For an update package, if you accept the defaults but do not include any objects, the system creates an empty package.

As you view the forms in the Package Assembly Director, you can accept the default selections by clicking Next. If necessary, you can always make changes at the final Package Component Revisions form.

Verifying a Path Code for Package Assembly

Before you assemble a package, you can verify that the path code from which the package is built is configured correctly.

This section provides an overview and discusses how to verify a path code.

Understanding the Process to Verify a Path Code

The verification process tests the environment, machines, and tables before a package is submitted. By verifying your environment, you eliminate the chance that your package build will fail due to configuration issues. This verification can save many hours in building a package.

During the verification process, the program verifies these items:

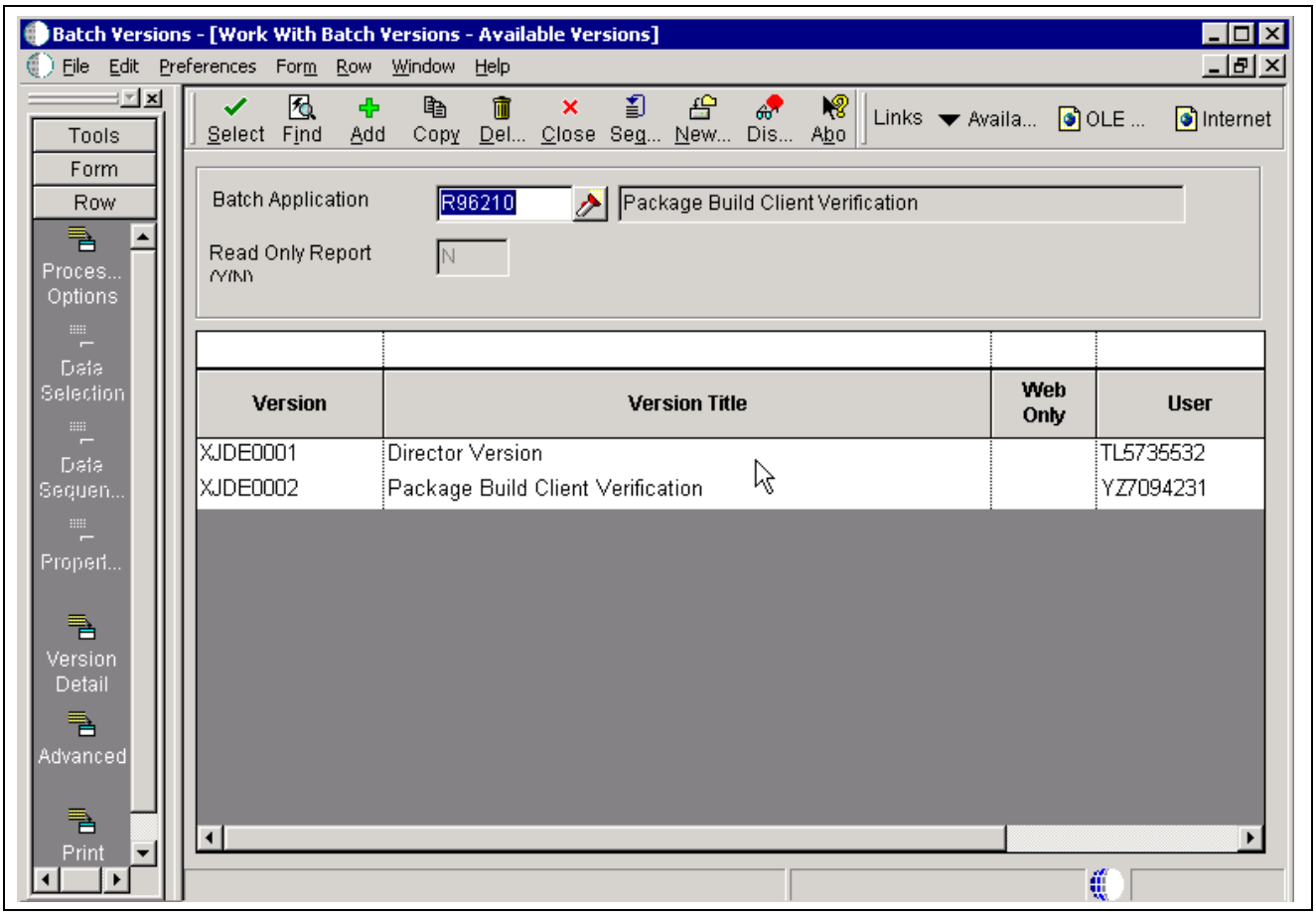
- Disk space is adequate
- Central objects and package build tables are accessible
- User has permissions to create directories on the deployment server and enterprise server
- Required service pack is installed
- Required MDAC is installed
- Machine tables are set up
- Required compiler version is installed
- Enterprise Server port is accessible
- Debug levels of the jde.ini files are adequate for the client and enterprise server

Form Used to Verify a Path Code for Package Assembly

Form Name	Form ID	Navigation	Usage
Work with Batch Versions - Available Versions	W98305A	From the Package and Deployment Tools menu (GH9083), select Package Assembly. Select Build Verification from the Form menu.	Verify the path code for package assembly.

Verifying a Path Code for package assembly

Access the Work with Batch Versions - Available Versions form.



Work with Batch Versions

To verify a path code for package build:

1. On Work with Batch Versions - Available Versions, select a version and click Select.
2. On Version Prompting, click Submit.

   Complete the Processing Options fields, and click OK.
3. On Printer Selection, select the desired printer, and click OK.

Assembling a Package

This section discusses how to:

- Assemble a new package
- Add a foundation location
- Add a help file location
- Add a database location
- Add batch versions
- Add features

- Review the package assembly selections

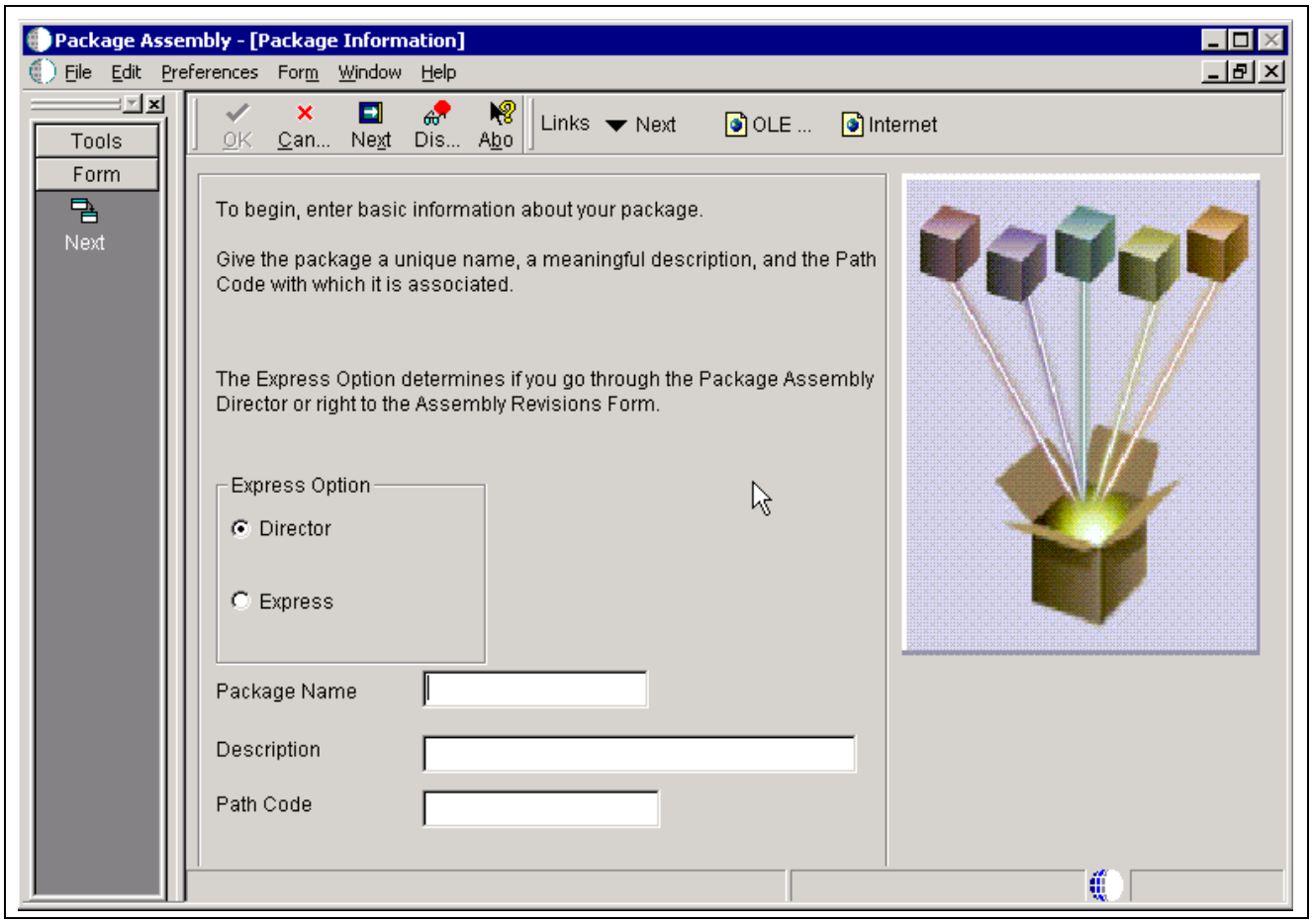
Forms Used to Assemble a Package

Form Name	Form ID	Navigation	Usage
Package Information	W9601F	From the Package and Deployment Tools menu (GH9083), select Package Assembly. Click Add, then click Next.	Assemble a new package
Mobile Client Database Revisions	W9601AD	From Package Information, click Next until the Mobile Client Database Revisions form appears.	Select to build a mobile package. If you select this option, you can also select the MSDE databases to include in the mobile package.
Foundation Component Selection	W9601H	From Package Information, click Next until the Foundation Component form appears. Click Browse. On Package Component Revision, click the Foundation button.	Add an existing foundation location to your package. Locate the existing foundation location that you want to use in your package and click Select.
Foundation Item Revisions	W9883D	From Package Information, click Next until the Foundation Component form appears. Click Browse. Click Add. On Package Component Revision, click the Foundation button. Click Add.	Add a new foundation location to your package.
Help Component Selection	W9601M	From Package Information, click Next until the Help Component form appears. Click Browse.	This is an obsolete form and has been disabled.
Help Item Revisions	W9883N	From Package Information, click Next until the Help component form appears. Click Browse. Click Add. On Package Component Revision, click the Helps button. Click Add.	This is an obsolete form and has been disabled.
Database Component Selection	W9601N	From Package Information, click Next until the database component form appears. Click Browse.	Add an existing database location to your package. Locate the existing database location that you want to use in your package and click Select.

Page Name	Object Name	Navigation	Usage
Database Item Revisions	W9883K	From Package Information, click Next until the database Component form appears. Click Browse. Click Add. On Package Component Revision, click the Database button. Click Add.	Add a new database location to your package.
Object Component Selection	W9601D	From Package Information, click Next until the Object Component form appears. Click Browse.	Add an object to your package.
Feature Component Selection	W9601AB	From Package Information, click Next until the Feature Component form appears. Click Browse. On Package Component Revision, click the Features button. Click Browse.	Add a feature to your package.
Package Component Revisions	W9601B	From Package Information, click Next until the Package Component Revisions form appears.	Review and revise the components in your package.

Assembling a new Package

Access the Package Information form.



Package Information

To run the Package Assembly Director:

1. On Package Information, complete the Package Name, Description, and Path Code fields:

   Note. You can build a single foundation package to deploy to all path codes by typing *ALL in the Path Code field. This option enables you to create an update package for service packs that can be installed to any pathcode. If you enter *ALL in the Path Code field, the application does not enable you to select, build, or deploy any objects (such as specs, help, business functions, and so on) in the package-only a foundation. The package is built to a directory called all_packages located under the release path (for example, B9/all_packages). This package can be deployed to any path code.

   Before you can use this option, you must add an entry to the Path Code Master called *ALL. See Path Code Setup in the Configurable Network Computing Implementation Guide for more information about how to set up this path code.

   ---

2. Select one of these options:

   - Director

     To configure the package assembly, click this option and then complete the remaining steps for this task.

   - Express

     To accept default values for the package assembly, click this option and then continue with the task Reviewing the Package Assembly Selections.

3. Click Next.

   On Package Type Selection, complete these fields:

Full or Update	If you are building a foundation package to the *ALL pathcode, the application automatically selects an Update package.
Parent Package	(for update packages only)
Include Object Specifications	(for client packages only)

   Note. If you select to build an Update package, the program disables the option to build a Mobile Client package. However, the program will automatically build a Mobile Client package for the update if the parent package includes a Mobile Client.

4. Click Next.
5. If you are assembling a full package and have at least one environment marked as “mobile, ” the Mobile Client Database Revisions form appears.
6. Click Build Mobile Client Package if you want to build a mobile package.

   Click Next.
7. On Foundation Component, perform one of these actions:
   - For full packages, accept the default location by clicking Next, or click Browse to specify another foundation location.

     Note. For more information about entering a foundation, see Entering a Foundation Location in the Package Management Guide.

   - For an update, click Clear unless the package includes a Foundation, and then click Next.
8. On Help Component, perform one of these actions:
   - accept the default location by clicking Next.

     Note. The Help forms are obsolete and have been disabled.

   - For an update, click Clear unless the package includes a Foundation, and then click Next.
9. On Database Component, perform one of these actions:
   - For full packages, accept the default location, or click Browse to specify another database location.

     Note. For more information about entering a database location, see Entering a Database Location in the Package Management Guide.

   - For an update, click Clear unless the package includes a Foundation.
10. Complete one of these actions:
    - If you are assembling a full package, click Next.

      For a full package, the system builds your package from the deployment data source that is associated with the default object path. Verify that the correct location appears on the form and proceed to the next step.

- If you are assembling an update package, click Next on Database Component, and then proceed to the next step.

  When you assemble an update package, the Object Component form appears. This form enables you to specify the individual objects that you want to include in your package. When you revise a previously assembled package, objects that you added earlier also appear.

11. On Object Component, to add an object, click Browse.
12. On Object Component Selection, locate and select the objects that you want to include in your package and then click Close to return to the Object Component form.

Note. For more information about adding objects, see Adding Objects to a Package in the Package Management Guide.

---

13. On Object Component, click Next.

    The Features Component form appears, on which you can specify the features that you want to include in your package. When you revise a previously assembled package, the system displays the features that you added earlier.
14. To add a feature, click Browse.
15. On the Feature Component Selection form, click Find to display a list of features, select one or more features, and then click Select to add the features that you want to include in your package.

    To select multiple features, press the Ctrl or Shift key while clicking features, and then click Select.
16. Click Close to return to Features Component.

Note. For more information about adding features, see Adding Features to a Package in the Package Management Guide.

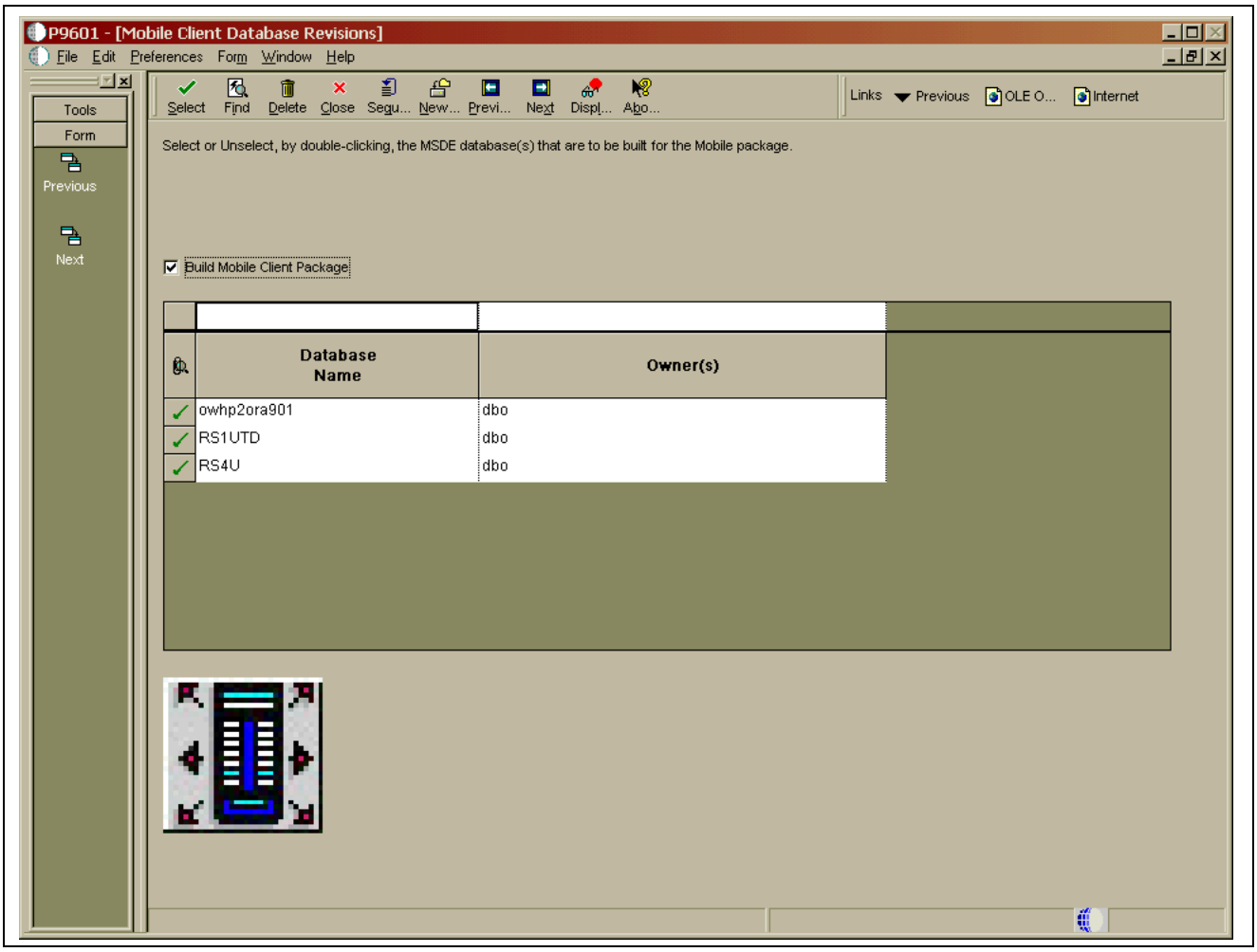
---

17. On Feature Component, click Next.
18. On Language Component, click Next if English is the only language that you want to configure.
19. To add a language to the language specifications for your package, double-click its row header in the detail area.

    If you add a language to your package, only that language will be included. For example, if you add French, English will not be included even though it is the default language. To include two languages (such as French and English), you must select the detail records for both languages.
20. Click Next.
21. Continue with Reviewing the Package Assembly Selections.

Selecting Mobile Packages

Access the Mobile client Database Revisions form.



Mobile Client Database Revisions

1. Select Build Mobile Client Package if you want to build a mobile package.

   If you select this option, the package build program will create a mobile package name by appending "_M" to the name of the existing package. The program also lists the database names and owners of the Mobile Client databases.

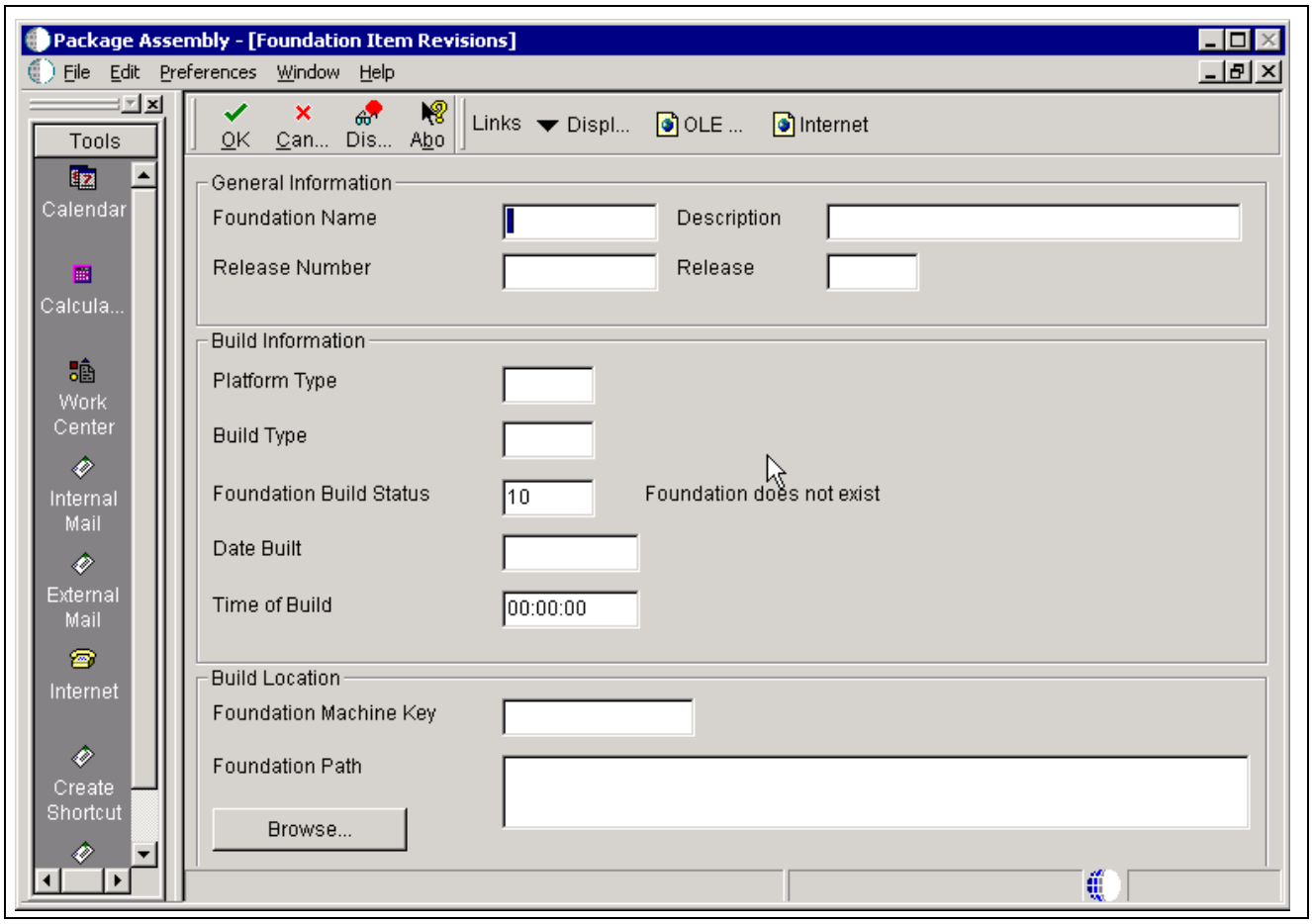
2. Select the MSDE databases you want to include in the mobile package.

   Note. You cannot change the owners of the mobile databases.

3. Click Select.
4. Click Next.

Adding a New Foundation Location

Access the Foundation Item Revisions form.



Foundation Item Revisions

The default foundation location is determined by the release that is associated with the path code for the package. This location is normally the system directory that is at the same directory level as your path code. Enter a foundation location to change the foundation location from the default location that appears on the Foundation Component form.

Foundation Name A foundation is the code required to run all applications. A foundation ID is required for all full packages. For full packages, if you do not select a foundation, the default foundation is used. The default foundation is determined through the release associated with the path code for the package. This is normally the system directory at the same directory level as your path code. The foundation must be compressed when built.

Release Number A service pack is an update to the foundation code that is delivered between major releases and cumulative releases of the software.

Release The release number as defined in the Release Master.

--- FORM SPECIFIC ---

On this form, Release Number is the EnterpriseOne major release with which this foundation is associated.

Platform Type Host Machine Type.

Build Type Indicate the compiler configuration to use for the software build.

Foundation Build Status	Describes the current status of the build process for foundation.
Date Built	The date the software build completed.
Time of Build	The time at which the software build completed.
Foundation Machine Key	The Location or Machine Key indicates the name of the machine on the network (server or workstation). --- FORM SPECIFIC --- On this form, the Machine Key is the name of the deployment server where your custom foundation resides.
Foundation Path	The shared directory for this path code. The objects that are stored on a file server will be found in this path. --- FORM SPECIFIC --- On this form, enter the exact path from which this item should be copied. All files in the last directory specified will be included in the package. Source Machine Key and Source are used together to define the item's location.

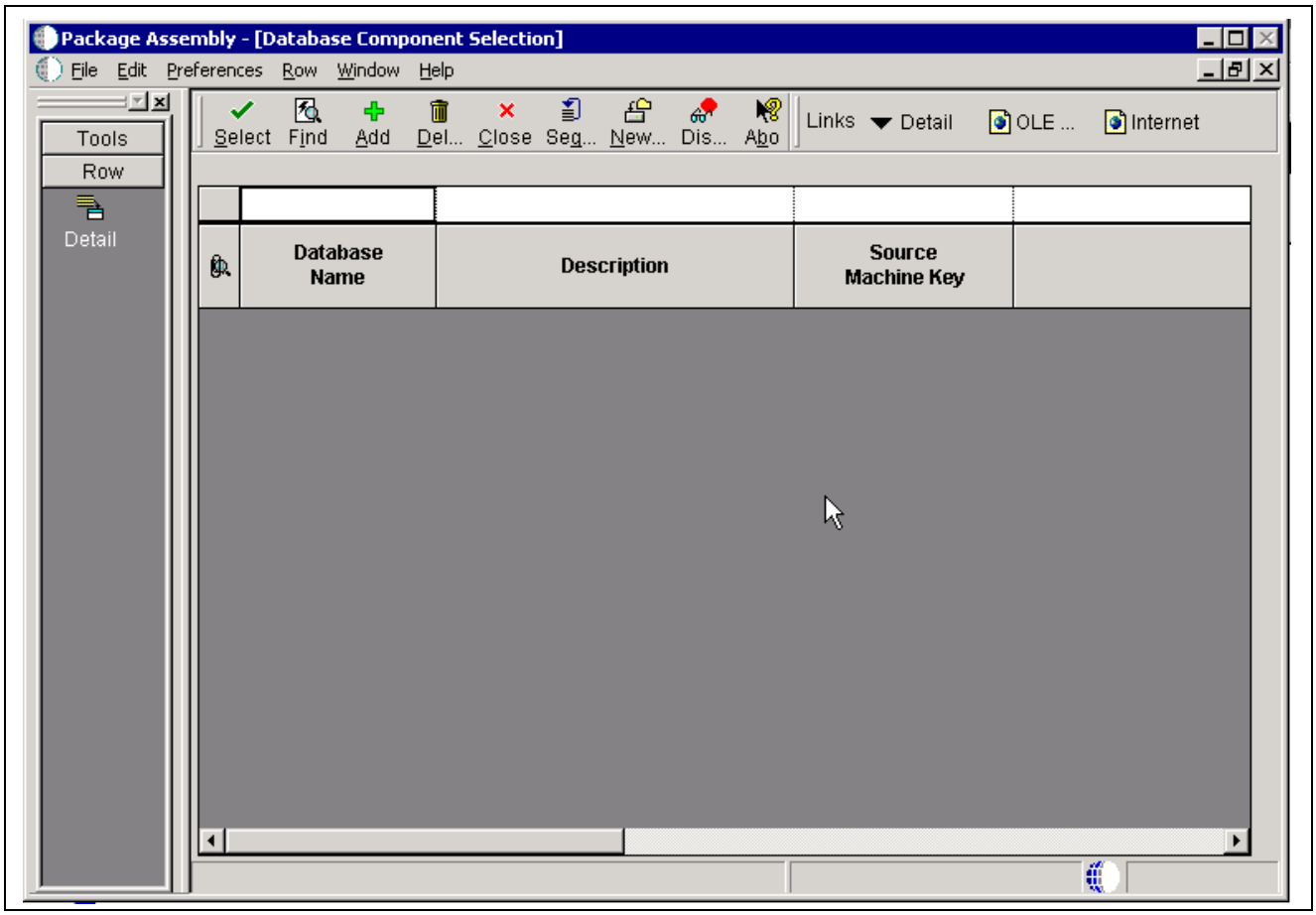
Adding a Help File Location

Access the Help Item Revisions form.

Note. This form is obsolete and has been disabled.

Adding a Database Location

Access the Database Item Revisions form.



Database Item Revisions

For full packages, if you do not specify a database location, the system uses the default database path (*pathcode*\Packages\Data). Update packages do not require a database.

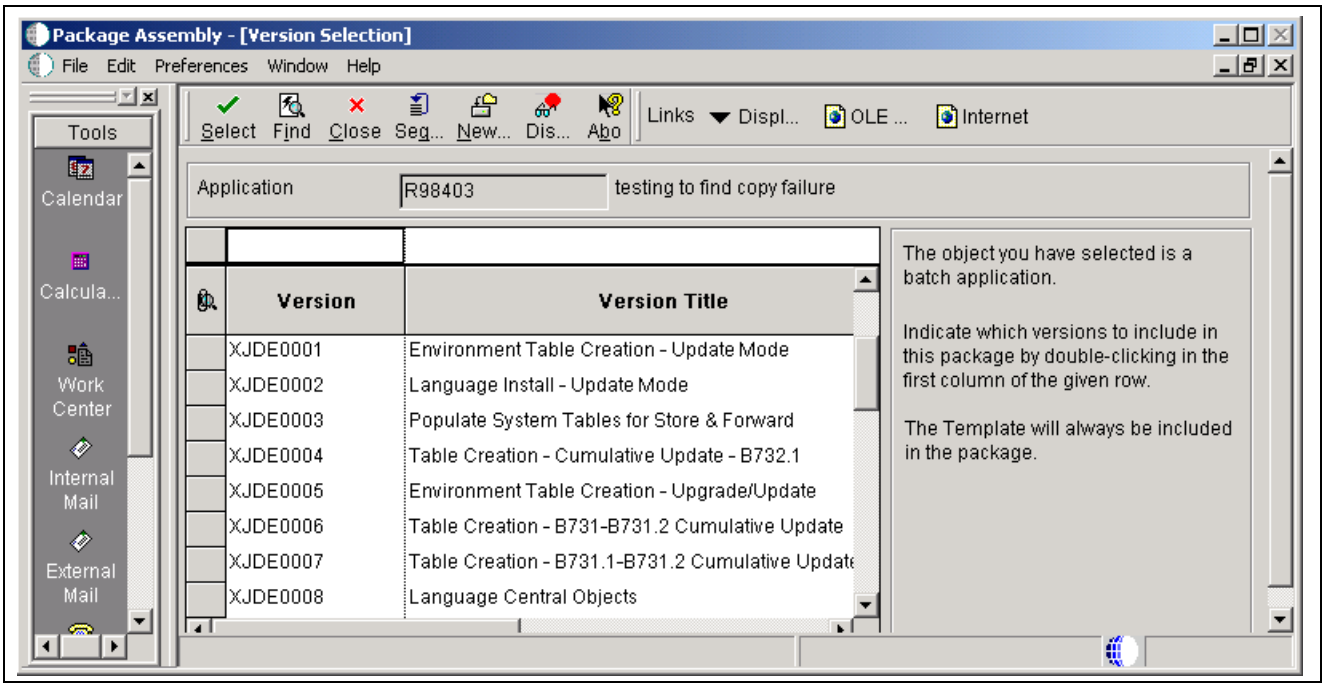
Database Name Name of the database component.

Source Machine Key The Location or Machine Key indicates the name of the machine on the network (server or workstation).

Server Share Path The shared directory for this path code. The objects that are stored on a file server will be found in this path.

Adding Batch Versions to a Package

Access the Version Selection form.



Version Selection

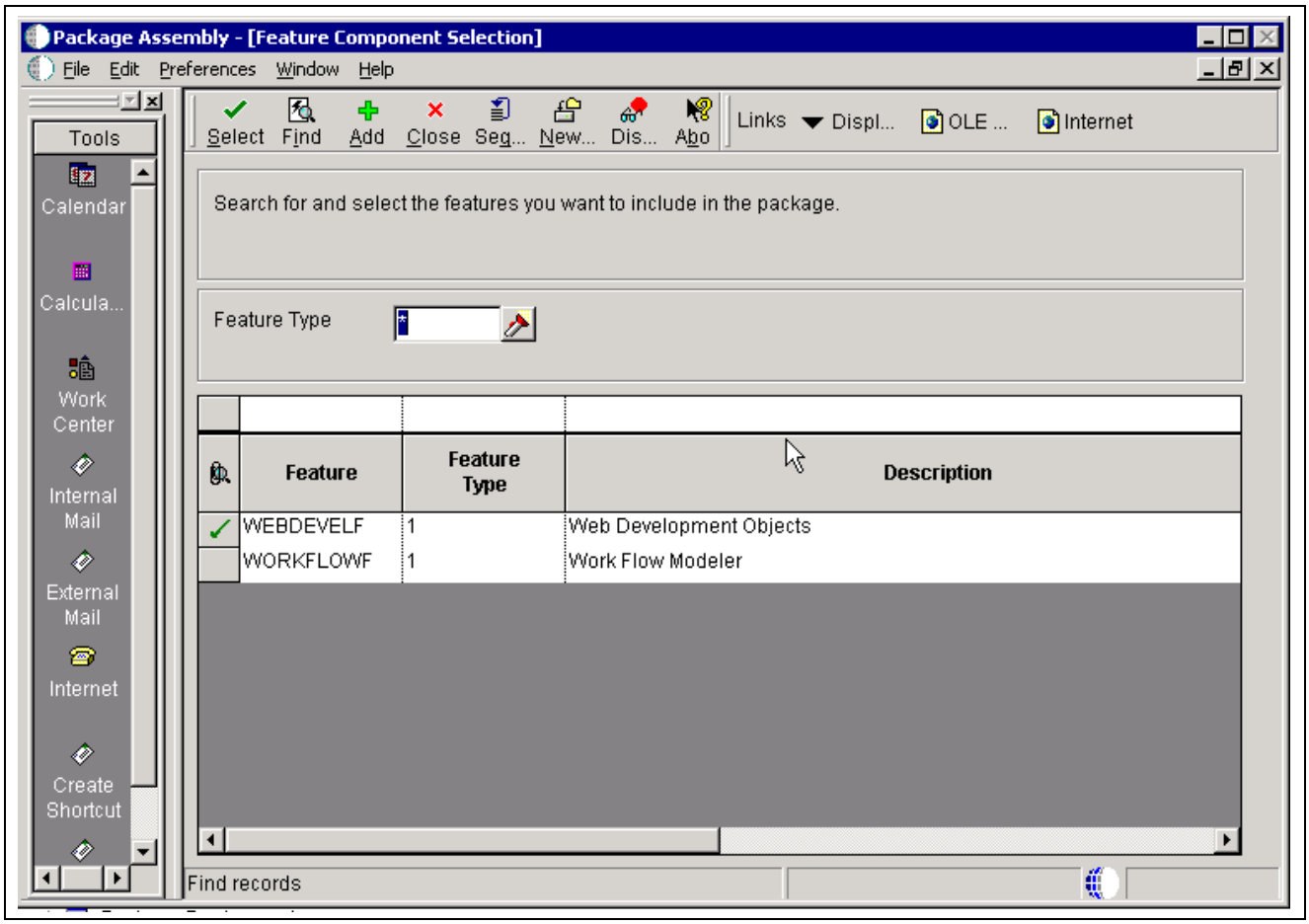
Version A user-defined set of specifications that control how applications and reports run. You use versions to group and save a set of user-defined processing option values and data selection and sequencing options. Interactive versions are associated with applications (usually as a menu selection). Batch versions are associated with batch jobs or reports. To run a batch process, you must select a version.

Version Title A description of the version that appears next to the version number. The version title is different from the report title.

This field should describe the use of a version. For example, an application for generating pick slips might have a version called Pick Slips - Accounting and another version called Pick Slips - Inventory Management.

Adding Features to a Package

Access the Feature Component Selection form.



Feature Component Selection

A feature is a set of files or configuration options, such as registry settings, that is copied to a workstation or server to support an application or other function. Like objects, features are built into a package and deployed to the workstations and servers that require the feature components.

To add features to a package:

1. Find and select the existing features that you want to include in the package, and then click Select.

   To select multiple features, press the Control or Shift key.

2. If the feature that you want to include has not been defined, you can create the feature definition by clicking Add.

   The Feature Based Deployment Director launches. You can use this feature to create the new feature.

3. Repeat steps 2 and 3 until you have finished adding features to your package.

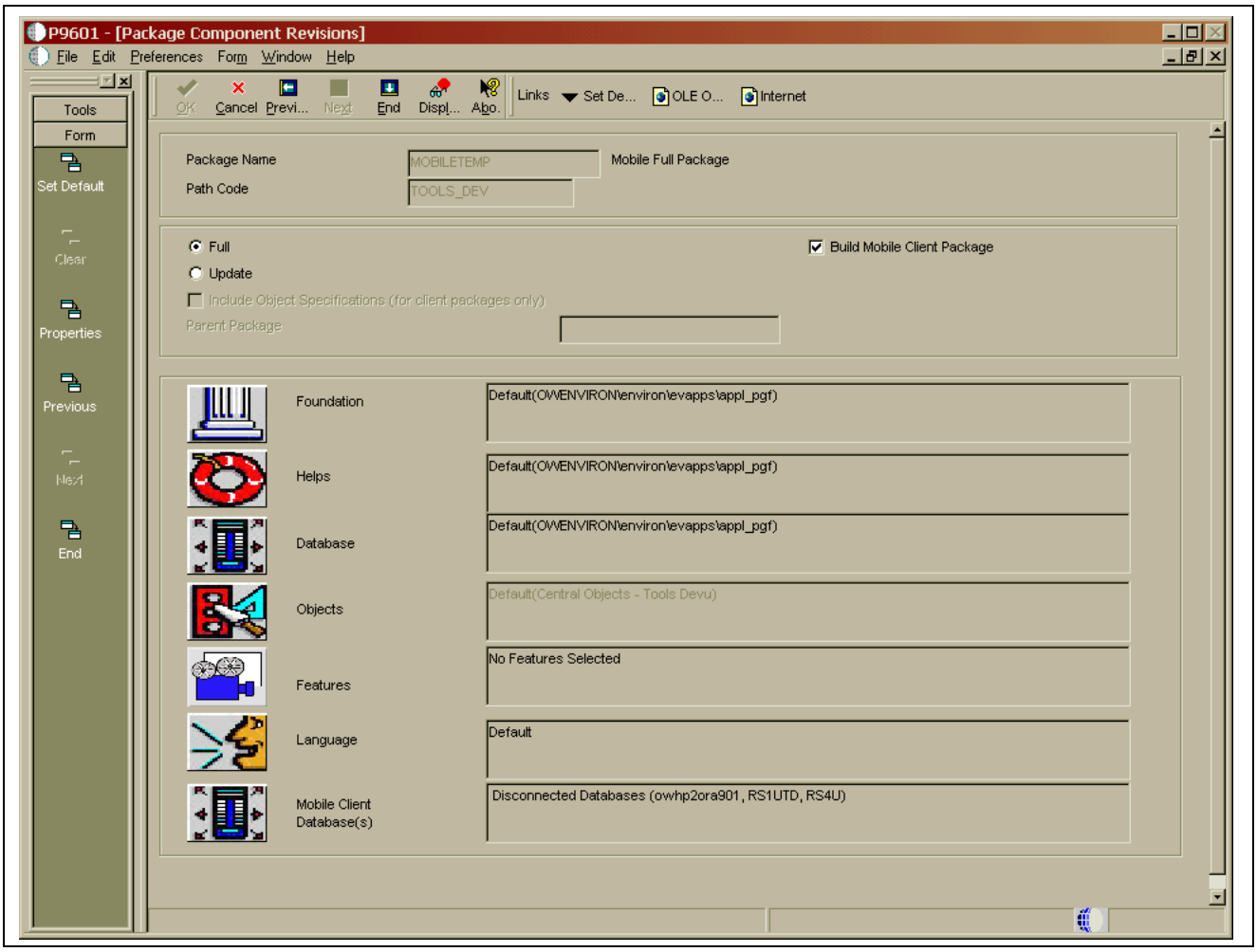
4. When you are finished, click Close to return to the form from which you accessed the Feature Component Selection form.

See Also

Chapter 4, “Assembling Packages,” Adding Features to a Package, page 73

Reviewing the Package Assembly Selections

Access the Package Component Revisions form.



Package Component Revisions

The Package Component Revisions form enables you to see, at a glance, the current foundation, help, and database locations, and mobile databases, as well as the objects and features in the package and the language selection, if one exists.

To review the package assembly selections:

1. On Package Component Revisions, to change any of the package components, click the button for the component that you want to change.

   The form for that package component appears.

   Note. From the Package Component Revisions screen, you can select or deselect the option to Build Mobile Client Packages. This option is automatically disabled if the name of the package currently being revised violates the eight character uniqueness rule or if a mobile package exists with the same name.

2. When you are finished assembling the package, click End to exit from the Package Assembly Director.
3. Continue with the task Activating an Assembled Package.

Field Descriptions

Path Code The path code is a pointer to a set of objects, and is used to keep track of sets of objects and their locations.

Include Object Specifications

When you include individual objects in an update package, you have the option of including a corresponding set of specifications for those objects. When you include specifications, a snapshot of the specifications will be included in the package after it is built. When the package is deployed, the package recipient receives those object specifications.

If you do not include object specifications in the package, the old specifications for the objects in the package will be deleted from the workstation when the package is deployed. Then, the next time the package recipient attempts to use the object, a new set of specifications for it will be transferred to the workstation using just-in-time installation.

Parent Package

Since an update package includes only a subset of objects, you must indicate the parent package on which the update package is based or related to. This information is used by the system to determine how to build business functions.

Selected Languages Only

When this option is turned on, only languages that have been selected for inclusion in the package will appear in the detail area.

Language

A list of valid codes for a specific user defined code list.

Revising an Existing Package

This section provides an overview of the package revision process and discusses how to revise a package.

Understanding the Package Revision Process

After you have assembled a package, you can use the Package Component Revision form to revise any of the components in the package. You do not need to complete all of the forms in the Package Assembly Director to revise a package.

Prerequisite

Verify that the status of the package is In Definition. If you try to revise a package that has a definition of Assembly-Definition Complete, the system displays an error message. To change the status of a package, select Active/Inactive from the Row menu on the Work with Packages form.

Form Used to Revise a Package

Form Name	Form ID	Navigation	Usage
Package Component Revisions	w9601B	From the Package and Deployment Tools menu (GH9083), select Package Assembly. Select the package that you want to revise, and then select Package Revisions from the Row menu.	Verify the path code for package assembly.

Revising an existing package

Access the Package Component Revisions form.

To revise an existing package:

1. On Package Component Revisions, make any necessary changes.
2. When you are finished revising the package definition, click OK to return to the Work with Packages form.

   If any build information exists for the package, the system warns you that your changes will delete the existing build information.
3. Select one of these buttons:
   - OK, to accept your revisions and delete the existing build information. If you accept the revisions, you should update the build information so that it reflects the changes that you made.
   - Cancel, to delete your revisions and save the existing build information.

Activating an Assembled Package

This section provides an overview of the activation process and discusses how to activate a package.

Understanding the Activation Process

After you have assembled a package, the package status remains at Assembly. While you define the package, it is inactive. You must activate the package to change the package status to Assembly-Definition Complete. An assembled package cannot be built until the status has been changed to Assembly-Definition Complete. The Assembly-Definition Complete status indicates that you are finished assembling the package and are ready to begin the build definition process.

You can change the package status at any time until you start the build definition process. That is, even after you have changed a package status to Assembly-Definition Complete, you can change the status back to In Definition if you need to revise the assembled package. When you are ready to define the build for the package, follow the steps described in Defining Package Builds.

Form Used to Activate a Package

Form Name	Form ID	Navigation	Usage
Work with Packages	W9601L	From the Package and Deployment Tools menu (GH9083), select Package Assembly. Select the package that you want to activate, and click Active/Inactive in the Row menu.	Activate the package. You can use this same process to change the status of a complete package back to In Definition.

CHAPTER 5

Building Packages

This chapter provides an overview of the Package Build process and discusses how to:

- Build a package
- Copy a built package
- Incorporate features into packages
- View package build records and resubmit builds

Understanding the Build Process

After you assemble a package, you must define the package build before you can build and deploy it to your workstations. The build process reads the central objects data source for the path code that you defined in the package. This information is then converted from a relational database format to replicated objects, which are put in the package itself.

The 811 (release name) directory structure looks similar to the example. Directories with an asterisk represent locations to which developers check in development objects.

PD811 (path code name)

- PACKAGE
  - PackageA (package name)
  - —bin32
  - —include
  - —lib32
  - —make
  - —obj
  - —res
  - —source
  - —spec
  - —work
  - —MobileSpec
  - —MobileDB
- *bin32
- *include

- *lib32
- make
- *obj
- *res
- *source
- work

When you build a package, the directories under the package name are populated. Files for the source and include directories are copied from the location under the path code on the deployment server from which developers check in development objects. Information for all other directories comes from the central objects data source. The bin32, lib32, and obj directories are populated with the output of the business function build process.

Note. Normally the packages are built in subdirectories under the package name on the deployment server. However, for mobile client packages, the mobile databases are instead created on the build machine, because MSDE databases cannot be created across the network.

The process that you perform to build a package might take several hours. For this reason, it is recommended that you initiate the actual package build at the end of the working day, if possible. This is a checklist of the tasks that you need to complete when you build a package.

- Transfer objects.

  Ensure that all of the objects that you want to include in the build have been transferred to the appropriate path code.
- Ensure that the database for the package has the most current replicated data.
- Build a package.

  Build a package using the path code to which objects were transferred.
- Perform a cross-reference build (optional).

  Perform a cross-reference build to verify that the cross-reference information reflects the changed objects. This process takes up to 15 hours to complete. You can deploy your package before the cross-reference build has finished.
- Deploy the software to these machines:
  - Workstations and servers
  - Tiered deployment locations

See Also

Chapter 4, “Assembling Packages,” Adding Features to a Package, page 73

Understanding the Package Build Definition Director

Like the Package Assembly Director, the Package Build Definition Director simplifies and expedites the build definition process by displaying a series of forms that guide you through the process. As with the Package Assembly Director, you can always either click Next to continue to the next form or click Previous to go back to the previous form. Also, you can always cancel the build definition process by clicking Cancel.

The following table summarizes the function of each form in the Package Build Definition Director:

Package Build Definition Director form	Use this form to review introductory information about the Package Build Definition Director.
Package Selection form	Use this form to select the defined package that you want to build. The status of the package must be Assembly-Definition Complete.
Package Build Location form	Use this form to specify whether you want to build the package for the client workstation, one or more servers, or both clients and servers.
Server Selection form	Use this form to specify the server location. (The server location is required when you build a package for a server.)
Build Specification Options form	Use this form to specify whether you want to include all specification tables in the package or only selected tables. The option to build individual specifications is useful if a build fails and your package error log indicates that an individual specification file needs to be rebuilt.
Individual Specification Selection form	Use this form to include only selected specs in the package.
Business Function Options form	Use this form to build business functions. You can also specify the build mode, the severity level at which to interrupt the build process, whether to build business function documentation, and whether to clear the output destination before building.
Compression Options form	Use this form to specify package compression and to specify whether to compress directories (all or individual), data, and foundation.
Individual Directory Selection form	Use this form to select the individual directories that you want to compress.
Build Features form	Use this form to enter file set and compression information for features that you added to the package. A feature is a set of files or configuration options, such as registry settings, that must be copied to a workstation or server to support an application or other function.
Package Build Revisions form	Use this form to review or change any of the options that you have specified for your package.

Building Business Functions During Package Build

As part of the build definition process, you can specify whether you want to build business functions. If so, the system globally builds business functions during the package build process. When you build business functions as part of the package build, the system performs the same process as if you had manually run the BusBuild program (selected Build from the Global Build menu) after you built the package.

The system retrieves source and header information from the package (from the source and include directories), compiles it, and stores it in the bin32, obj, and lib32 directories. The system builds business functions in the package, not on the workstation. If you select the Compress Package option, the system compresses the business functions after it builds them.

These guidelines apply to the path code, foundation, and destination for the business function build:

- When building business functions, use the path code that you defined in the package.
- The foundation is either the same as the foundation that is included in the package or, for an update package, it is the foundation for the parent package.
- Build output is directed to the bin32, obj, and lib32 directories of the package itself.
- When building a full package, or when building an update package that includes a business function, always build business functions. Otherwise, the consolidated DLLs included in the package will not be current.

  For update packages, the system builds each business function individually. After it builds an individual business function, the system performs a global link for that object and all other objects that are in the same consolidated DLL. The global link affects all objects in the check-in location for the path code of that package.

Note. The build process does not update check-in location directories with the output from the build.

See Also

EnterpriseOne Tools 8.94 PeopleBook: Development Tools: Application Development, "Getting Started," PeopleSoft Tools Application Development Overview

Building a Package

This section provides overviews of the Package Build Definition Director and package compression, and discusses how to:

- Set Processing Options for the Package Build Definition Director
- Define a Package
- Review Package Build Selections
- Build a Package

Understanding the Package Build Definition Director

You can access the Package Build Definition Director either from the Package Assembly menu selection, or from the Package Build menu selection. The advantage of accessing the director from the Package Assembly menu selection is that the system automatically enters the package name and other information. If you access the director from the Work With Package Build Definition form, you must manually specify the name of the package that you want to build.

Before you launch the Pack Build Definition Director, you can use the Work with Package Build Definition form to review information about any previously-designed packages. For example, you can review the properties, build options, business function options, and compression options for the package. As on any other parent/child form, you can click the plus (+) symbol to view more information about the package or click the minus (-) symbol to view less information about the package.

Viewing Package Build History and Resubmitting Builds

After you submit the package for building, you can track the build status using the Package Build History program (P9622). This application also enables you to view logs that are associated with the build process to determine if any errors occurred during the build process.

If the build did not complete successfully, you can resubmit the package and resume building from the point where the build stopped. Alternatively, you can reset the status of the specifications and objects and then build the package again.

See Also

Chapter 5, "Building Packages," Viewing the Package Build History, page 119

Understanding Package Compression

This section describes how to:

- Compress a parent package (after building an Update)
- Compress a server package

Compressing the Parent Package (After Building the Update)

The system backs up the parent package cab files when you build an update package. During the package build process, you must select options on the Compress Options form to recompress the parent package before you deploy the parent package.

Compressing Server Packages

To compress packages that you build on the server, add the [BSFN BUILD] section to the jde.ini file on the *client/deployment* workstation and create this entry:

DoCompression=1

This setting compresses packages that you built on the server and deploys them to other servers of the same type, such as all AIX Unix servers or NT servers. If you plan to deploy a package to an enterprise server, you must build the package on the same type of server with compression selected.

When the server builds a compressed package, it stores the compressed files in subdirectories, such as \bin32, \specs, that are subordinate to this path on the deployment server: *\package\package name\server type*, where*package name*is the name of the package, and*server type*is the type of server for which the package is compressed. The compression process creates a new file called compressed.inf in the*server type*directory. This file includes the information that the system needs to deploy the compressed files. This table describes the type of compressed files that the process creates for each type of server:

NT	UNIX	iSeries
.cab	.z	SRVPG, SPECS, MODULE, USRSPC

When you deploy the package to another enterprise server, the system reads the compress.inf file and uses this information to copy the compressed files from the package directory on the deployment server to the enterprise server.

Compressing Server Update Packages

To compress update packages for a server and deploy it to other servers of the same type, enter a *1* for processing option number 2 on the Package Build Director application (P9621).

This option displays a field to compress an update package when defining that package. When you select this option and build the package, the program creates a compressed file in the *package name**server type*\bin32 directory. The .pak files are stored in the *package name**server type*\spec directory and do not get compressed. The program also creates .cab files in the*package name*\ directory, although these files are not used when you deploy to a client workstation.

Prerequisites

Before you complete the tasks in this section:

- Assemble your package and verify that the status of the assembled package is Assembly-Definition Complete.
- Verify that Object Configuration Manager (OCM) mappings are correctly set for the Package Build (R9621) and Server Package Build (R9622) programs, which the system generates as part of the package build process. For example, if you want the programs to run locally, ensure that the OCM mappings point to Local for the environment in which the package build is running.

See Also

EnterpriseOne Tools 8.94 PeopleBook: Configurable Network Computing Implementation, "Object Configuration Manager"

Forms Used to Build a Package

Form Name	Form ID	Navigation	Usage
Package Selection	W9621C	From the Package and Deployment Tools menu (GH9083), select Package Build. Click Add, then click Next.	Select a package to define a build.
Package Build Location	W9621F	From the Package Selection, select Director and click Next.	Select the type of package to build (Client or Server).
Server Selection	W9621A	From the Package Build Location, select Server and click Next.	Select the servers to add to the package.
Build Specification Options	W9621G	From Server Selection, click Next.	Select to include all specs or individual specs in the package build.
Individual Specification Selection	W9621P	From Build Specification Options, select Individual Specification Tables and click Next.	Select individual specs to include in the package build.

Page Name	Object Name	Navigation	Usage
Business Function Options	W9621N	From Package Selection, select a full or update package that includes business functions or tables with table event rules, complete the forms, then click Next.	Select options for building business functions.
Compress Options	W9621M	From Package Selection, select a full package and complete the forms, then click Next.	Select options for compressing a package.
Individual Directory Selection	W9621H	From Compress Options, select Individual Directories, then click Next.	Select individual directories to compress.
Package Build Revisions	W9621B	From Package Selection, complete the forms, as required, then click Next. From the Package and Deployment Tools menu (GH9083), select Package Build. Select a package, and deactivate it by selecting Active/Inactive from the Row menu. Select Build Revisions from the Row menu.	Review and change package build options.
Work With Package Build Definition	W9621L	From the Package and Deployment Tools menu (GH9083), select Package Build.	Add a package build definition. Build a defined package.
Build Verification	NA	From the Package and Deployment Tools menu (GH9083), select Package Assembly or Package Build. From the Row menu, select Build Verification.	Verify the configuration of the package and the environment.

Setting Processing Options for the Package Build Definition Director (P9621)

Processing options enable you to specify the default processing for programs and reports.

For programs, you can specify options such as the default values for specific transactions, whether fields appear on a form, and the version of the program that you want to run.

For reports, processing options enable you to specify the information that appears on reports. For example, you set a processing option to include the fiscal year or the number of aging days on a report.

Do not modify EnterpriseOne demo versions, which are identified by ZJDE or XJDE prefixes. Copy these versions or create new versions to change any values, including the version number, version title, prompting options, security, and processing options.

Processing

1. Enter a value to determine how changes will occur.

Blank = Changes will only be allowed at the package level and will apply to all servers selected.

1 = Changes will be allowed to the build definitions by individual server.

2. Mark this processing option with a 1 if this process is for Mastering purposes. If the process is for all users, mark this processing option with a blank.

<Blank> = All Users

1 = Mastering Only

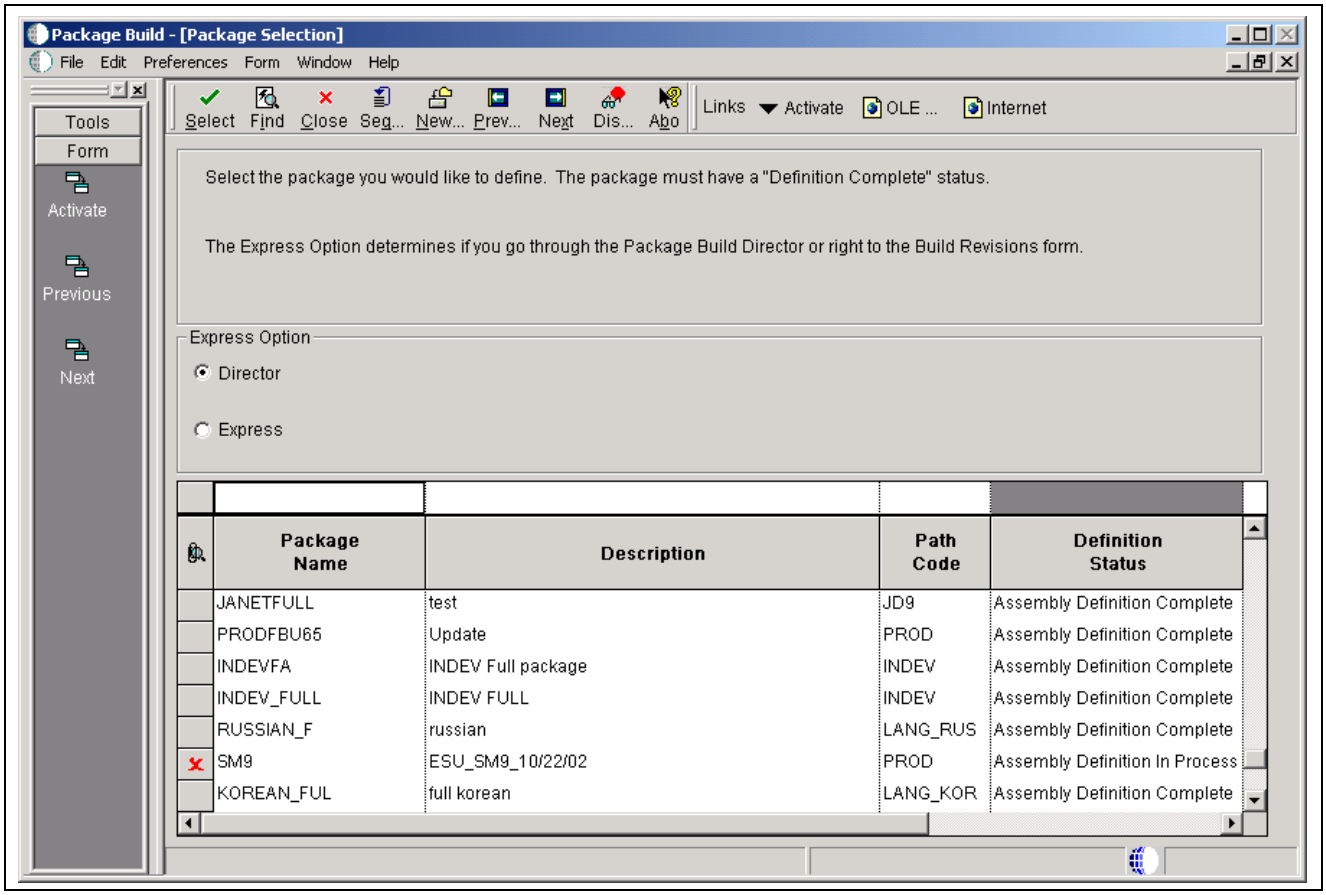
3. Mark this processing option with a 1 if the Build Verification UBE is to be run prior to building all packages. If the build verification fails, the package build UBE will not be run.

Blank - Do not run Build Verification

1 - Run Build Verification

Defining a Package Build

Access the Package Selection form.



Package Selection

To define a package build:

1. On Package Selection, find and select the defined package that you want to build.

   If the package definition has a status of In Definition, you must change the status to Assembly-Definition Complete before you build the package. To change the status, select the package and select Activate from the Row menu.

2. On the Express Option pane, select one of these options:

   - Director

     Select this option if you want to configure your package build. Director enables you to navigate the Package Build Definition forms.

   - Express

     Select this option if you want to accept the default build parameters. Express enables you to accept the default options for the package build and skip the package build definition forms.

3. Perform one of these actions:

   - If you chose Express, click Next and continue with the task To review the package build selections.
   - If you chose Director, click Next and complete the remaining steps in this task.

4. On Package Build Location, select one or both of these options to turn them on, and then click Next:

   - Client

     Note. If the package includes a build for a mobile client, the Client option is automatically selected.

   - Server(s)

     If you are building a package for workstations only, proceed to step 10.

     If you clicked the Server(s) option on the Package Build Location form, the Server Selection form appears. This form enables you to select the servers on which you want to build the package.

5. To select a server, click Find, and then double-click the row header for the server.

   A check mark indicates your selection. You can select multiple servers.

6. Click Next.

7. On Build Specification Options, select Build Options to take the package definition and copy and convert objects from the central data source to the replicated format used by workstations.

8. Select one of these options:

   - All Specification Tables

     Indicate whether you want to build all specification tables into the package or whether you would rather select individual tables to include in the package.

   - Individual Specification Tables

     If you select All Specification Tables, all of the tables listed on the Individual Specification Selection form will be included in the package. If you are building all specification tables, click Next and complete the Business Function Options form.

9. Complete these fields:

   Stop-Build Option The Stop-Build Option field enables you to indicate the point at which the system should stop the build. You can continue building on all errors, stop building on specification errors, stop building on business function errors, or avoid compressing when errors exist.

Replace JDE.INI This field applies to update packages only, and indicates if you want a new JDE.INI file delivered with the package. You should leave this unchecked unless your JDE.INI file has changed. For example, your JDE.INI may change when you perform upgrades or when you reconfigure in release master.

10. Click Next.

    If you chose to build individual specification tables, the Individual Specification Selection form appears.

11. To indicate that you do not want to build a specification table, click its option to clear it.

    You can clear multiple options.

12. Click Next.

    For a full package or for an update package that includes business functions or tables with table event rules, the Business Function Options form appears.

13. Complete these fields:

Generate NER When resubmitting a client or server build, you can indicate whether NER should be rebuilt. If you already have a successful build of the NER, you can save substantial time by not rebuilding it.

Build Functions Options Select options for the build functions.

Build Mode You can select from three build modes: debug, optimize, and performance. Debug and performance are for PeopleSoft developers only. Users should select the optimize mode.

Stop-Build Option This option enables you to specify what action the system should take if an error occurs during the business function building process.

Build BSFN Documentation Indicate whether you want to build business function documentation during the package build process. For a full package, the documentation will be built for all business functions. For an update package, documentation will be built for only those functions included in the package. Verify that this option is cleared.

Clear Output Destination This option applies to a full package. When building business functions, you should normally clear the output directories (bin32, lib32 and obj) so that old business functions are not included when the package is created. If you do not clear the output and a function fails to process, the global link process will use the old libraries and objects to create the consolidated DLL file. This can result in an inaccurate global package.

14. Click Next.

    If you are building a full package, the Compression Options form appears. If you are building an update package, this form does not appear.

15. On Compression Options, click this option to select it:

    - Compress Options

      Select this option to compress the applications included in the package, and to specify options for the compression process.

      If the package that you are building will be deployed to a server, you should select Compress Options only under these circumstances:

- You are building the same package for both the workstation and server, and you want to create compressed files for the workstation package
- You plan to build the package on one enterprise server and deploy it to another enterprise server.

16. Select from these options:
    - All Directories
    - Individual Directories
    - Compress Data

      Indicate whether to compress the data in a package after the package is created. Compress Data compresses the Supported Local Database that is associated with this package.
    - Compress Helps

      This functionality is obsolete and has been disabled.
    - Compress Foundation

      Indicate whether to compress the foundation files in the package after the package is created. Compress Foundation compresses the foundation that is associated with your package.

      If you select the All Directories options, all of the directories listed on the Individual Directory Selection form are compressed. If you are compressing all directories, skip this next step.

      Note. Verify that the DoCompression setting is set to 1 to enable compression. If this setting is not set to 1, the system does not create compressed files on the server. For more information, see JDE.INI Settings for Server Package Builds in the Package Management Guide.

    For a Server Package

    If you need to compress a server package, click Compress Options and select All Directories. This selection compresses the directories on the enterprise server and copies them to your package on the deployment server in this directory (the directory on which your server.ini file is located):

    <pathcode>/<package>/<package name>/<operating system type>

    The directories are compressed into file types that are compatible with the type of enterprise server that you are using. For NT servers, the file type is .cab, for UNIX the file type is .z, and for AS400, the file has no extension.

    For a Client Package

    When you compress directories, the application objects that are included in the package are automatically compressed. The system also creates an entry in the package INF file that indicates whether the foundation, data, and application objects are compressed.
17. Click Next.

    If you chose to compress individual directories, the Individual Directory Selection form appears.
18. On Individual Directory Selection, to indicate that you want to compress a directory, click its option to select it.

    You can select multiple options.
19. Click Next.

    If the package does not include features, skip to the next task.

20. On Build Features, if you want to build a feature.inf file with the package, click the Build Feature INFs option to select it:

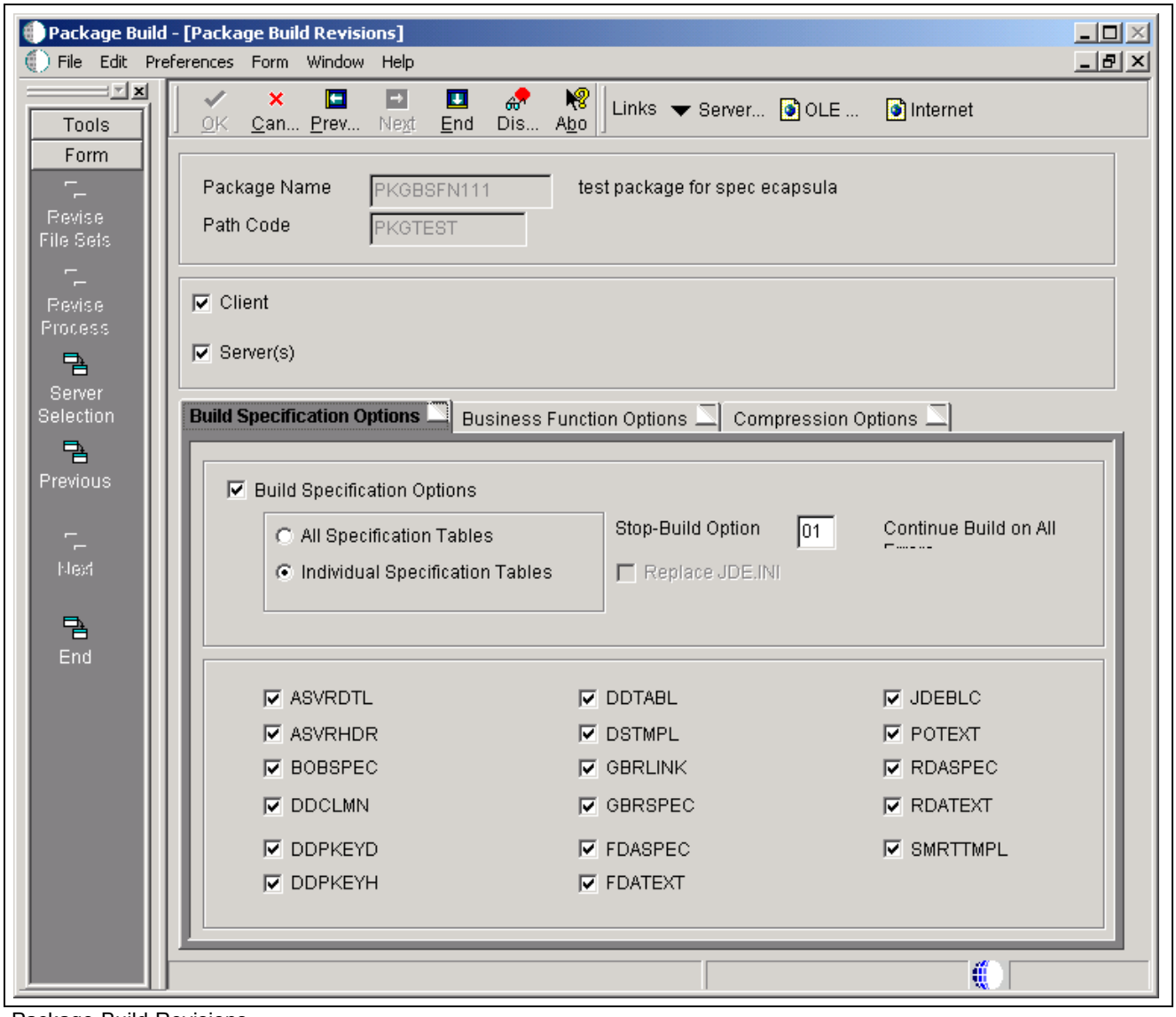
Note. When you select this option, the Compress and Build options become available. For more information about the Compress and Build options, see Configuring Features During Package Build Definition in the Package Management Guide.

21. Click Next.

22. Continue with the next task, Reviewing Package Build Selections.

Reviewing Package Build Selections

Access the Package Build Revisions form.



Package Build Revisions

To review the package build selections:

This form enables you to see at a glance the current build options, business function options, compression options, and feature options that you specified for the package.

1. To change any of these options, click the tab for the type of option that you want to change.

   Only tabs for options that you selected appear on this form.

2. On Package Build Revisions, when you are finished reviewing or changing your build options, click End to exit the Package Build Definition Director, or click OK to accept changes to an existing package.
3. On Work With Package Build Definition, activate the package by choosing Active/Inactive from the Row menu.

   After you enter the build options for a package, you can easily revise any of those options using the Package Build Revision form. You do not need to go through all of the forms in the Package Build Definition Director to revise build options.

Building a Package

Access the Work With Package Build Definition form.

To build a package:

1. On Work With Package Build Definition, select Active/Inactive from the Row menu to activate the package.
2. Select Submit Build from the Row menu when you are ready to initiate the package build.
3. Select one of these options and click OK:
   - On Screen
   - To Printer

   The form closes and the system begins building the package. Build time varies, depending on the number and size of the items in your package. A build could take five minutes for a small package, or several hours for a full package that contains all applications. When the build finishes, the report either appears on the screen or prints, depending on the destination that you specified.
4. Review the report to make sure that all components in your package were built successfully. If the report indicates any errors, review the error logs for more detail.

   If the package build finishes successfully, you can schedule the package for deployment.

Verifying a Package

After you assemble a package or define a package build, you can verify whether the package can be built successfully. You can use this verification to test the package before you submit the build, or troubleshoot problems with the build process if the package build fails.

During the verification process, the program verifies these items:

- Disk space is adequate
- Central objects and package build tables are accessible
- User has permissions to create directories on the deployment server and enterprise server
- Required service pack is installed
- Required MDAC is installed
- Machine tables are set up
- Required compiler version is installed

- Server port is accessible
- Debug levels of the jde.ini files are adequate for the client and enterprise server

Copying a Built Package

This section includes and overview of the process to copy a package and discusses how to:

- Copy a package build from a specific server
- Copy a general package build definition

Understanding the process to copy a package build

After you build a package, you can copy the package definition, which includes the package record, header, and detail files. The copy function is useful in cases where you want to create a package that is similar to an existing package. In this situation, you can copy the package definition and then modify the package record, header, or detail files as needed.

You can select from these two ways to copy a package:

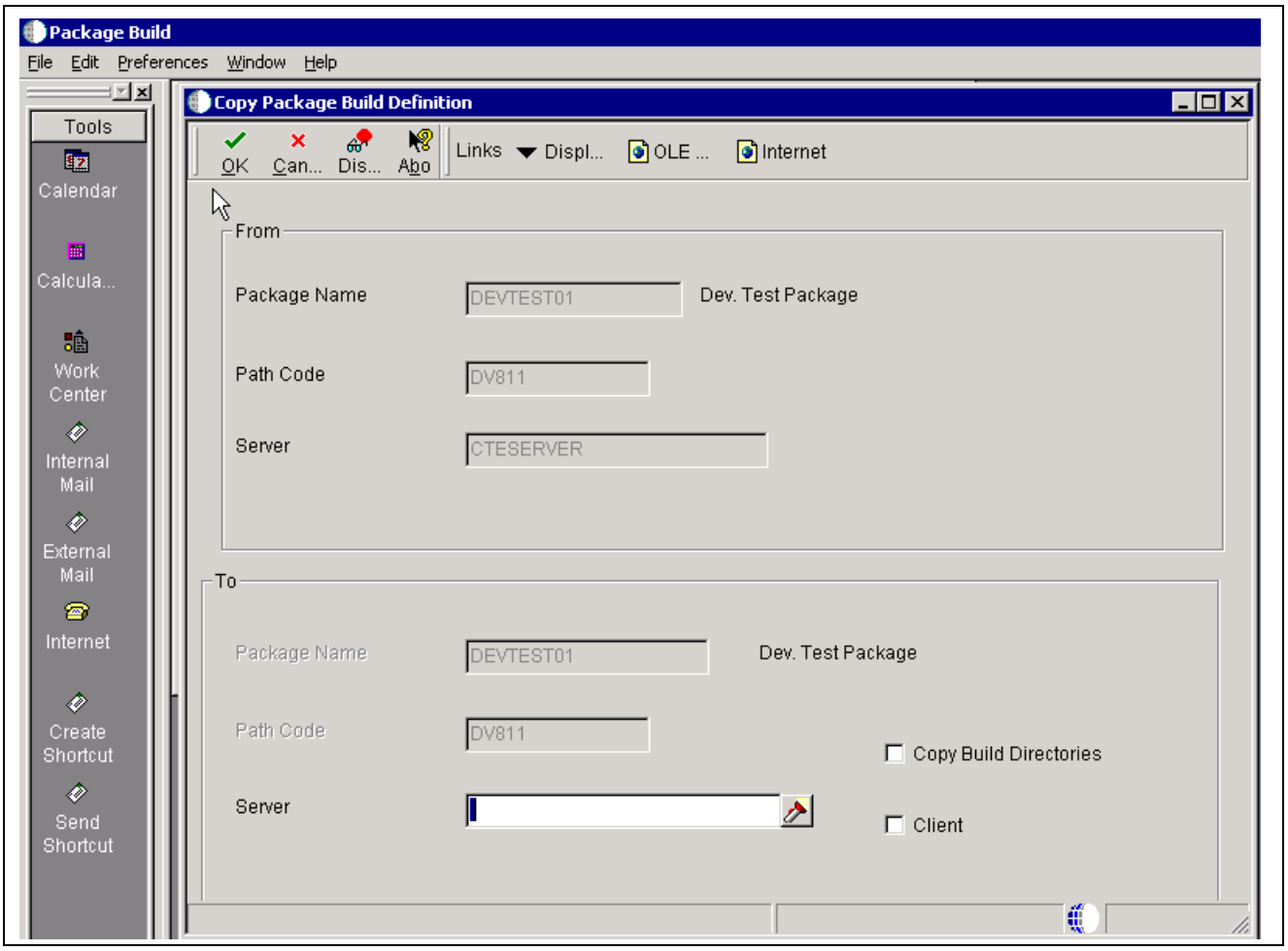
- Copy the build definition for a specific server. You might use this method when you want to copy the same package from one server to another, or to a workstation.
- Copy the build definition for the entire package. You might use this method when you want to create a new package based on an existing package.

Forms Used to Copy a Built Package

Form Name	Form ID	Navigation	Usage
Copy Package Build Definition	WP9621I	From the Package and Deployment Tools menu (GH9083), select Package Build. Select a package, and select the server for which you want to copy the definition. Click Copy. From the Package and Deployment Tools menu (GH9083), select Package Build. Select a package and click Copy.	Copy a previously built package

Copying a package build definition from a specific server

Access the Copy Package Build Definition form. (Be sure to select the server from which to copy the definition.)



Copy Package Build Definition

To copy a package build definition for a server:

1. On Copy Package Build Definition, enter the name of the server to which you want to copy the package.

   The system displays the package name, path code, and server name from which you are copying, as well as the destination package name and path code, which are the same.

2. Complete these optional fields:

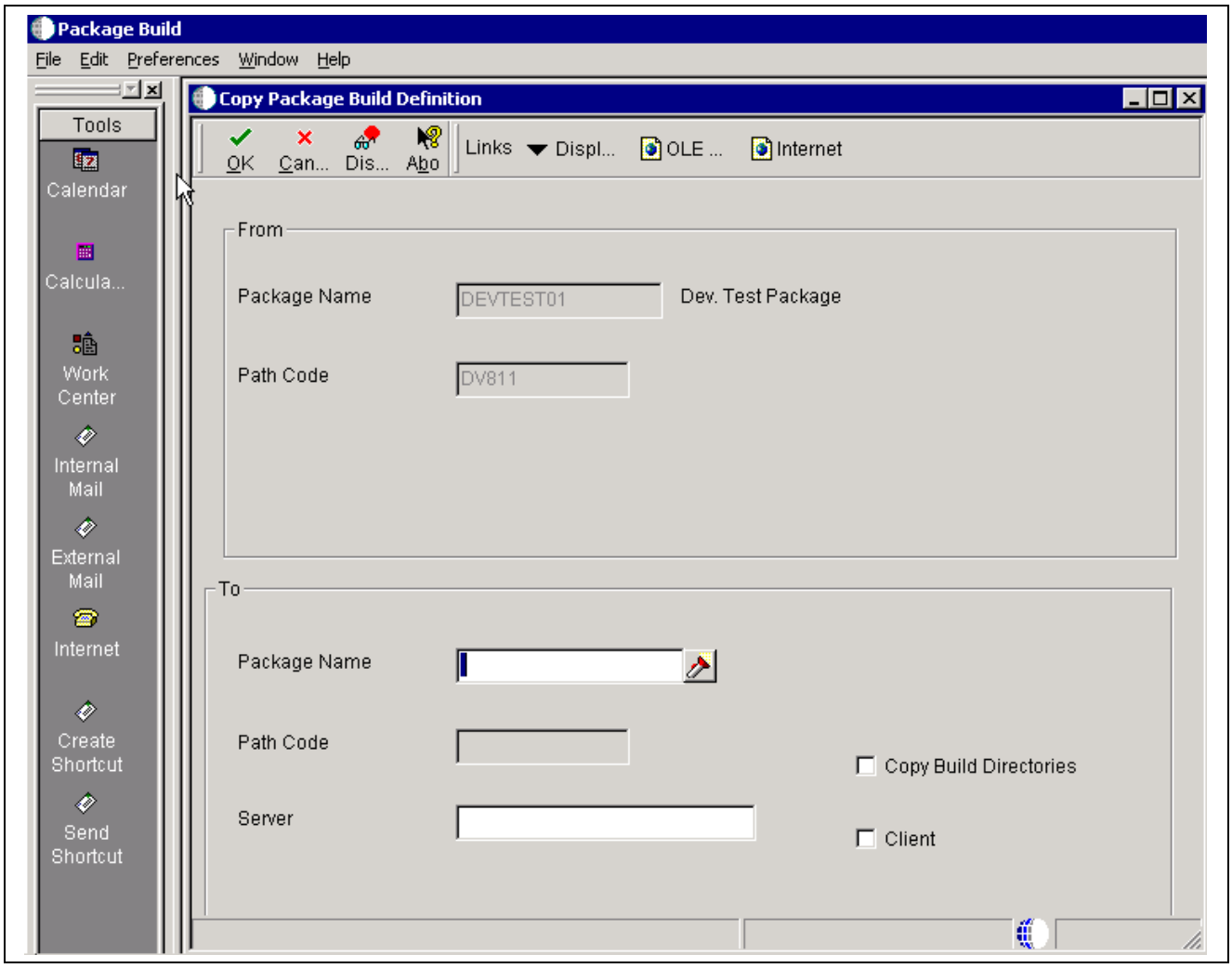
   Copy Build Directories Select this option to copy the package build directories.

   Client Select this option to copy to the client workstation instead of to the server.

3. Click OK.

Copying a general package build definition

Access the Copy Package Build Definition form. (Do not select a server from which to copy the definition.)



Copy Package Build Definition

To copy a general package build definition:

1. On Copy Package Build Definition, enter the new package name and server name.

   The system displays the package name, path code, and server name from which you are copying.
2. Complete these optional fields:

   Copy Build Directories Select this option to copy the package build directories.

   Client Select this option to copy to the client workstation instead of to the server.
3. Click OK.

Incorporating Features into Packages

This section includes an overview of the feature build process and the Feature Based Deployment Director and discusses how to:

- Create a feature

- Define a file set
- Define a registry
- Define a shortcut
- Define additional package build processes
- Define additional install processes
- Define an intialization file
- Define a new ODBC data source
- Import an existing ODBC data source
- Review feature components
- Copy features
- Add a feature to a package
- Configure features during the package build definition
- Configure features for an existing package build definition

Understanding the Feature Build and Deployment Process

A feature is a set of files or configuration options, such as registry settings, that is copied to a workstation or server to support an application or other functions. Like objects, features are built into a package and deployed to the workstations and servers that require the feature components.

Note. EnterpriseOne Web Development clients require a specific feature component in order to develop web-based objects. Refer to the *Web Development Client Installation and Configuration PeopleBook* for instructions on adding the web development feature to a package. This feature is also required for mobile packages, because mobile clients require a Web Development client in order to operate.

Here are some examples of features that you might want to include when you build a package:

- ActiveX controls. The Application Design Aid tool enables you to include ActiveX controls in applications. If ActiveX controls are delivered with the software, you need a way to copy these controls to the workstation.
- Open Data Access (ODA) data sources. ODA requires that additional ODBC data sources be created on any workstation or server that uses ODA.
- Sales Force Automation databases. The Sales Force Automation feature requires that you install a separate Supported Local Database on the workstation so that it can be disconnected from the network during offline operation. You must also write a registry setting that indicates that the machine is used offline.
- BMC Patrol, GenCorba, GenCom, and other third-party interfaces or products. Each of these products and interfaces requires additional components on the workstation and server in order to function. As functionality expands to support additional third-party products and interfaces, these products will each have their own set of supporting files.

For software releases prior to EnterpriseOne 8.10, custom programming was required to add feature components to the workstation and server. You can now use familiar tools such as the Package Assembly Director and the Package Build Definition Director to create a package that contains your feature, and then you can deploy it using the Package Deployment Director or multitier deployment.

Because feature components are not objects, the process for incorporating feature components into a package is slightly different from the normal package build process. Specifically, you must first define the feature before you can add it to a package.

Define the feature

Before adding the feature to a package, you must first define it using the Feature Based Deployment Director. During feature definition, you specify the feature name and type, enter a brief description, and specify installation parameters.

The forms in the Feature Based Deployment Director enable you to:

- Create a file set
- Define registry settings
- Define a Windows shortcut
- Enter initialization file information
- Add ODBC data sources
- Specify the feature build sequence
- Enter information for third-party products

Select the feature during package assembly

After you have defined the feature, it is ready to be included in a package. Use the Package Assembly Director to assemble the package as you would any other package. When you assemble the package, feature-specific forms enable you to specify the features that you want to include.

Configure the feature during package build definition

After you have assembled the package that contains the features, you can use the Package Build Definition Director to define the build for the package. Forms in this director enable you to select the file sets that will be compressed within the package, and to specify the processes that will be run before and after the feature is built.

Deploy the package

After you have built the package, you are ready to schedule it for deployment by using the Package Deployment Director. The procedure is the same as the procedure that you use to schedule packages that do not include features.

Run Workstation Installation and Deployment Server Installation

After you have deployed the package to workstations and deployment servers, use the Workstation Installation and Deployment Server Installation applications to install the package.

Understanding Feature Entries in the Package.INF File

When a package contains a feature, the Package.INF file [Features] section provides the feature name and the location of the feature.INF file that the system creates for each feature. The feature.INF file contains information pertaining to the feature, such as shortcut information, registry settings, initialization file settings, and environment information.

Installing Packages Containing Features

You install packages containing features on workstations and servers in the same way in which you install any other package: through the Workstation Installation and Deployment Server Installation applications.

When you launch either of these installations, you can select the Custom option to select the features that you want to install.

See Also

See these topics in the Package Management Guide:

Chapter 3, "Understanding the Package Build Process," Features, page 54

Chapter 4, "Assembling Packages," Adding Features to a Package, page 73

Chapter 4, "Assembling Packages," Adding Features to a Package, page 73

Chapter 3, "Understanding the Package Build Process," Workstation Packages, page 44

EnterpriseOne Installation Guide

Understanding the Feature Based Deployment Director

The Feature Based Deployment Director enables you to define your feature so that it can be included in a package and then deployed to workstations and servers. The forms in the director enable you to specify the name and type of the feature, as well as the different feature components. The feature Information form enables you to select the types of components to include in the feature, and determines the subsequent forms that display in the Feature Based Deployment Director.

For this release, the Platform value must always be 80 for CLIENT. Future releases will enable you to select alternative platforms.

Throughout the feature definition process, you can always proceed to the next or previous form by clicking Next or Previous. Also, regardless of where you are in the process, you can always cancel the feature definition by clicking Cancel.

The following table summarizes each of the forms in the Feature Based Deployment Director and the function of each form:

Copying a feature definition

The Feature Based Deployment Director includes a copy function that enables you to copy an existing feature and rename it as a new feature. This feature is especially useful if you want to create a feature definition that closely matches an existing feature definition.

Forms Used to Incorporate Features into Packages

Form Name	Form ID	Navigation	Usage
Feature Information	W9326C	From the Package and Deployment Tools menu (GH9083), select Package Assembly. Select Features from the Form menu. Click Add. Click Next.	Define a feature and add one or more components to the feature.

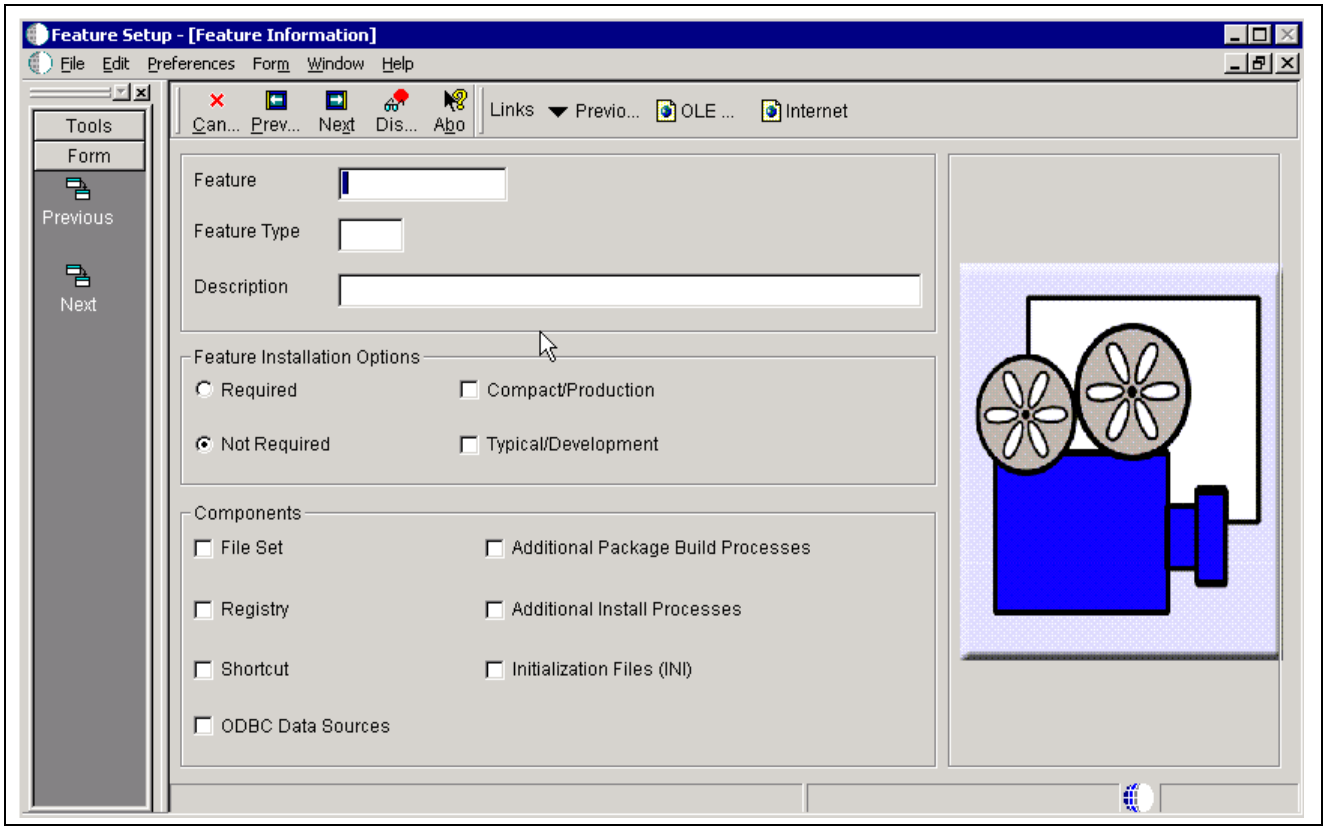
Page Name	Object Name	Navigation	Usage
File Set Definition	W9326J	From the Feature Information form, select File Set and click Next.	Enter information about any file sets that must be installed on the workstation or server in order for the feature to function properly. A file set is a collection of files that must be installed on the workstation or deployment server in order for the feature to function correctly.
Registry Definition	W9326D	From the Feature Information form, select Registry and click Next until the Registry Definition form appears.	Enter information that should be added to the Windows registry as part of the feature installation. Registry information that you enter on this form will be delivered in the package that contains the feature.
Shortcut Definition	W9326G	From the Feature Information form, select Shortcut and click Next until the Shortcut Definition form appears.	Use this form to add a shortcut for your feature to the Windows desktop. The system creates a shortcut on the desktop after the feature is installed.
Shortcut Advanced Options	W9326P	From the Shortcut Definition form, select Advanced from the Form menu.	Enter advanced shortcut options.
Additional Package Build Processes	W9326H	From the Feature Information form, select Additional Package Build Processes and click Next until the Additional Package Build Processes form appears.	Specify a batch application or executable program to run either before or after the package that contains the feature is installed.
Additional Install Processes	W9326K	From the Feature Information form, select Additional Install Processes and click Next until the Additional Install Processes form appears.	Enter information about third-party applications that should be run when the package is installed.
Initialization File (INI) Definition	W9326I	From the Feature Information form, select Initialization Files (INI) and click Next until the Initialization File (INI) Definition form appears.	Enter information that should be written to an initialization file (such as JDE.ini) as part of the feature installation. The INI file is automatically updated when the package is installed.

Page Name	Object Name	Navigation	Usage
ODBC Data Source Definition	W9326N	From the Feature Information form, select ODBC Data Sources and click Next until the ODBC Data Source Definition form appears.	Enter information for any ODBC data sources that must be added to support the feature.
Local Data Sources	W9326O	On ODBC Data Source Definition, select Import from the Form menu.	Select previously created data sources that reside locally on your machine.
Features Summary	W9326L	From the Feature Information form, click Next and add the each feature component. After you add all the components, the wizard displays the Features Summary form.	Review and modify information that you entered on any of the Feature Based Deployments forms.
Feature Copy	W9326M	From the Package and Deployment Tools menu (GH9083), select Package Assembly. Select Features from the Form menu. Select the feature from which to copy the definition, and click Copy.	Copy an existing feature and rename it as a new feature. This function is useful if you want to create a feature definition that closely matches an existing feature definition.
Feature Component Selection	W9601AB	From the Package and Deployment Tools menu (GH9083), select Package Assembly. Click Add to create a new package. Enter the forms in the Package Assembly Directory until the Features Component form appears. Click Browse. From the Package and Deployment Tools menu (GH9083), select Package Assembly. Select a package and then select Package Revisions from the Row menu. On Package Component Revisions, click the Features button. To add a feature, click Browse.	Add defined features to a new package. Add defined features to an existing package that is open for revision.

Page Name	Object Name	Navigation	Usage
Build Features	W9621B	During a new package build definition: From the Package and Deployment Tools menu (GH9083), select Package Build. Click Add to launch the Package Build Definition Director. Click Next and complete the screens until you come to the Build Features form. From an existing package: From the Package and Deployment Tools menu (GH9083), select Package Build. Find and select the package that contains features. Select Build Revisions from the Row menu. Click the Build Features tab.	Enables you to specify whether the system builds feature INF files for the features in the package. If you defined a fileset component in your feature, you can select to compress it. If any additional package build processes are included in the feature, you must click Build Processes and select them before they will run during package build.

Creating a Feature

Access the Feature Information form.



Feature Information

To create a feature:

1. On Feature Information, complete these fields:
   - Feature
   - Feature Type
   - Description
2. Select one:
   - Required

     The installation of this feature is mandatory for both Compact/Production and Typical/Development installs. Inclusion of this feature cannot be overridden when the package is installed.
   - Not Required

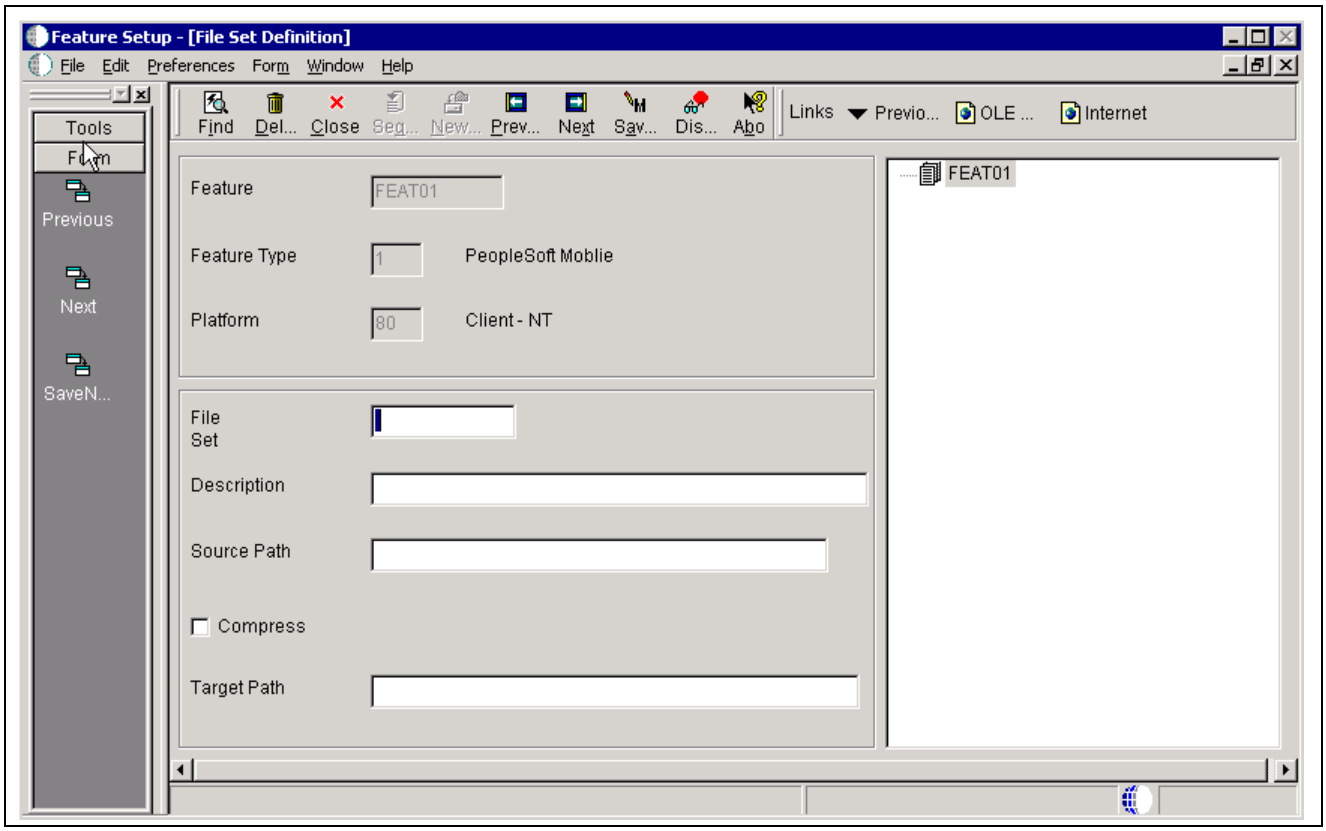
     The installation of this feature is optional. Whether the feature is installed depends on the options that you select (Compact/Production and Typical/Development). Inclusion of the feature can be overridden when the package is installed.
3. Select one or both of the options that follow. (If you chose Required, both of these options are automatically selected.)
   - Compact/Production

     When turned on, this feature is included in a Compact/Production install by default. This option can be overridden when the package is installed if Not Required is also selected.
   - Typical/Development

     When turned on, this feature is included in a Typical/Development install by default. This option can be overridden when the package is installed if Not Required is also selected.
4. Select any of these options and feature components that apply to your package. The forms that subsequently appear depend on the feature components that you select.
   - File Set
   - Registry
   - Shortcut
   - ODBC Data Sources
   - Additional Package Build Processes
   - Additional Install Processes
   - Initialization Files (INI)
5. Click Next.
6. Continue with the next task.

Defining a File Set Component

If you selected the File Set component, the File Set Definition form appears.



File Set Definition

To define a file set component:

1. On File Set Definition, complete these fields:

   File Set A free-form text field for comments or memoranda.

   File Set Description A description of a group of files.

   Source Path A path that identifies the source location of a file set.

   Compress An option to compress the file.

   Target Path A path that identifies the target location of a file set.

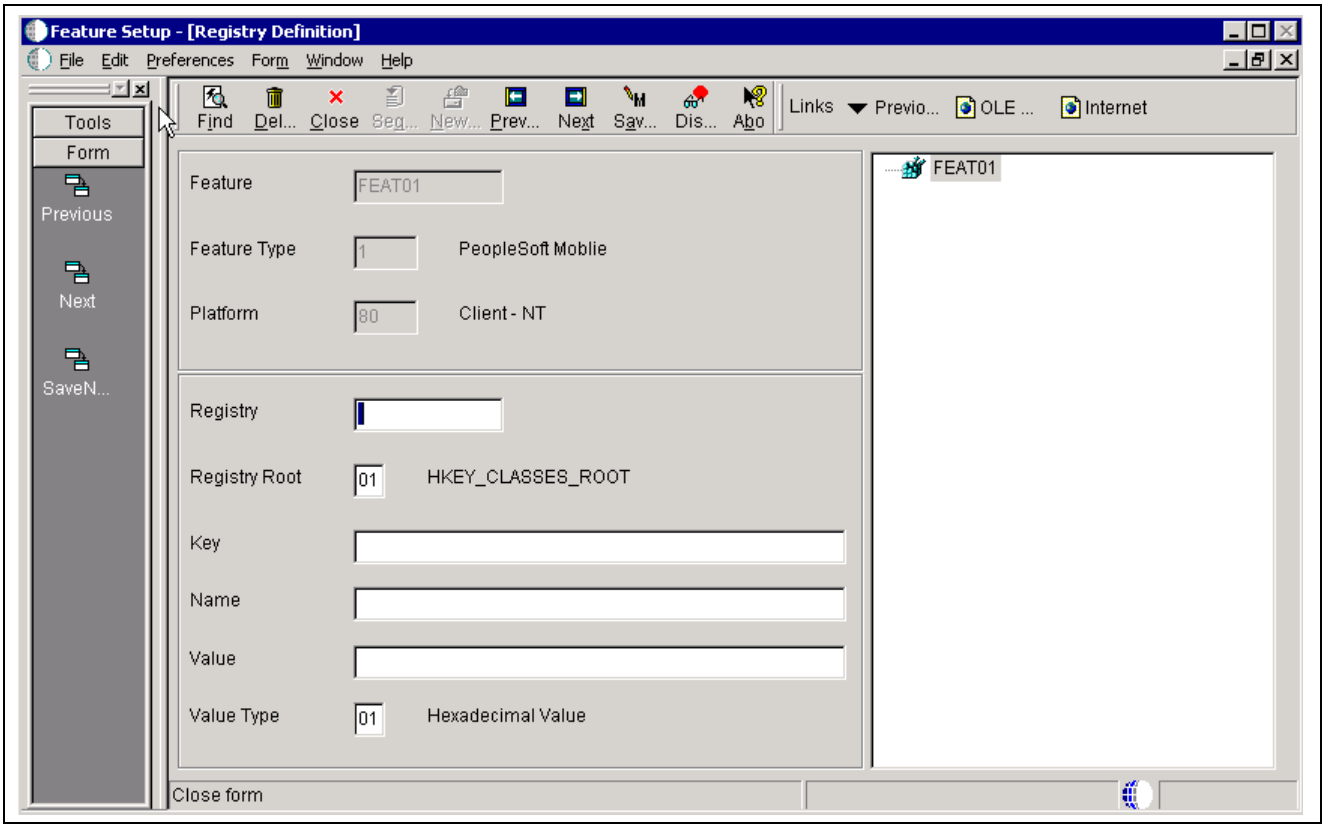
   The source path tells the system where to find the file set to be copied into the package, and the target path indicates the location to which the file set should be copied when the package is installed. Although a feature can have an unlimited number of file sets, each file set can have only one target path.

   You can also use this form to modify or delete any previously defined file sets. Existing file sets appear in the tree structure on the right side of the form. To modify a file set, select the file set on the tree structure and modify any of the fields for the file set. To delete a file set, select the file set and click Delete.

2. When you are finished adding file set information, select Save Node from the Form menu.
3. Click Next.

Defining a Registry Component

If you selected the Registry component, the Registry Definition form appears.



Registry Definition

To define a registry component:

1. On Registry Definition, complete these fields:

   Registry The identifier of a registry modification.

   Registry Root The root key in the registry.

   Key The key for a registry value.

   Name The registry value name.

   Value The registry value.

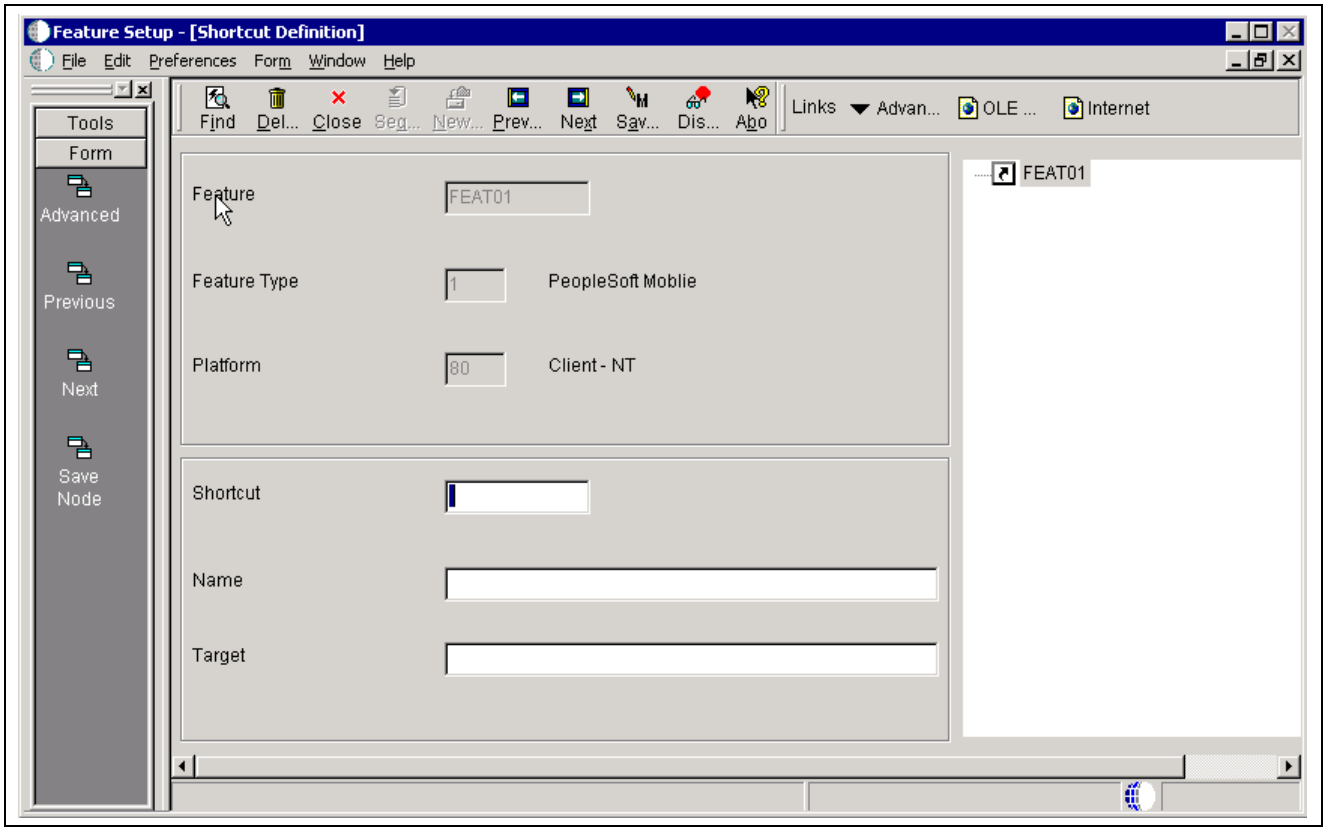
   Value Type In the registry, the data type in which the value is stored.

   You can also use this form to modify or delete any previous registry definitions. Existing registry definitions appear in the tree structure on the right side of the form. To modify a registry definition, select the item on the tree structure and modify any of the fields for the registry definition. To delete a registry definition, select the item and click Delete.

2. When you are finished adding registry information, select Save Node from the Form menu.
3. Click Next.

Defining a Shortcut Component

If you selected the Shortcut component, the Shortcut Definition form appears.



Shortcut Definition

To define a shortcut component:

1. On Shortcut Definition, complete these fields:

   Shortcut A name that identifies a unique shortcut to a user's computer.

   Name The name of the shortcut.

   Target The path and file name of a target file.

2. To enter advanced shortcut options, select Advanced from the Form menu.
3. On Shortcut Advanced Options, complete any of these fields:

   Arguments Parameters that are entered at the command line for the shortcut.

   Description The description of the shortcut.

   Hot Key A key sequence that, when pressed, automatically launches the shortcut.

   Icon The path and name of an icon file, based on a relative target path.

   Icon Index The icon index for a shortcut.

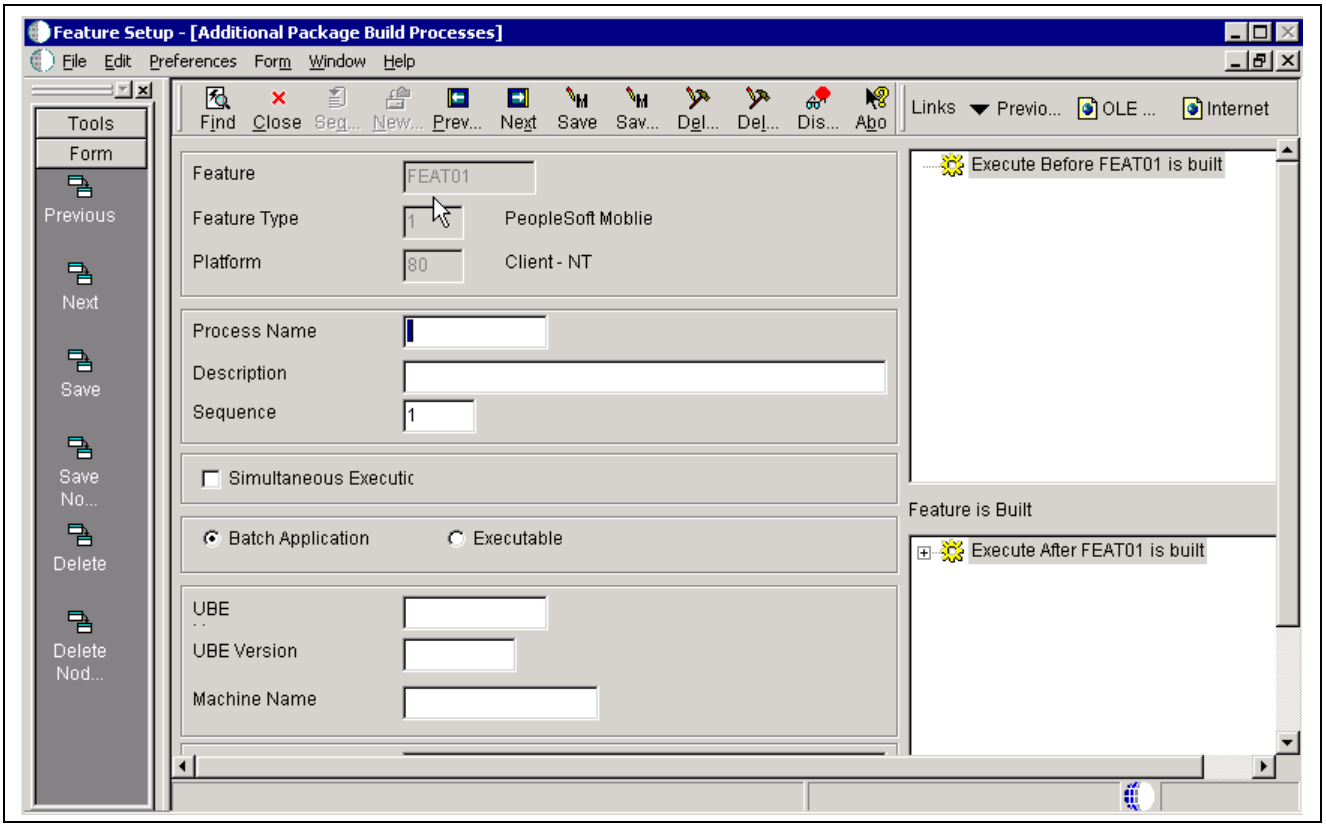
   Show Command The size of the window after the shortcut is launched. For example, the window might be minimized or maximized.

   Work Directory The identifier of the directory path or the working directory of a shortcut.

4. Click OK to save your entries and return to the Shortcut Definition form.
5. Complete the shortcut information and select Save Node from the Form menu.
6. Add all of your shortcuts.
7. Click Next.

Defining Additional Package Build Processes

If you selected additional package build processes, the Additional Package Build Processes form appears.



Additional Package Build Processes

To define additional package build processes:

1. On Additional Package Build Processes, complete these fields:

Process Name	Name of the build process.
Description	Description of the build process.
Sequence	A number that identifies the order in which the process will be run relative to the other processes that run during the package build.
Synchronous Execution	An option that indicates whether the package build job waits for the process to finish before it continues.
Batch Application or Executable	Specify whether the process is an application or an executable.

2. If you selected Batch Application, complete these fields:

- UBE Name
- UBE Version
- Machine Name

  The name of the server or workstation on which the UBE will run.

3. If you selected an executable program, complete these fields:

   Executable Name The name of the executable program that the system launches to install the third-party software.

   Target Path The path and file name of a target file.

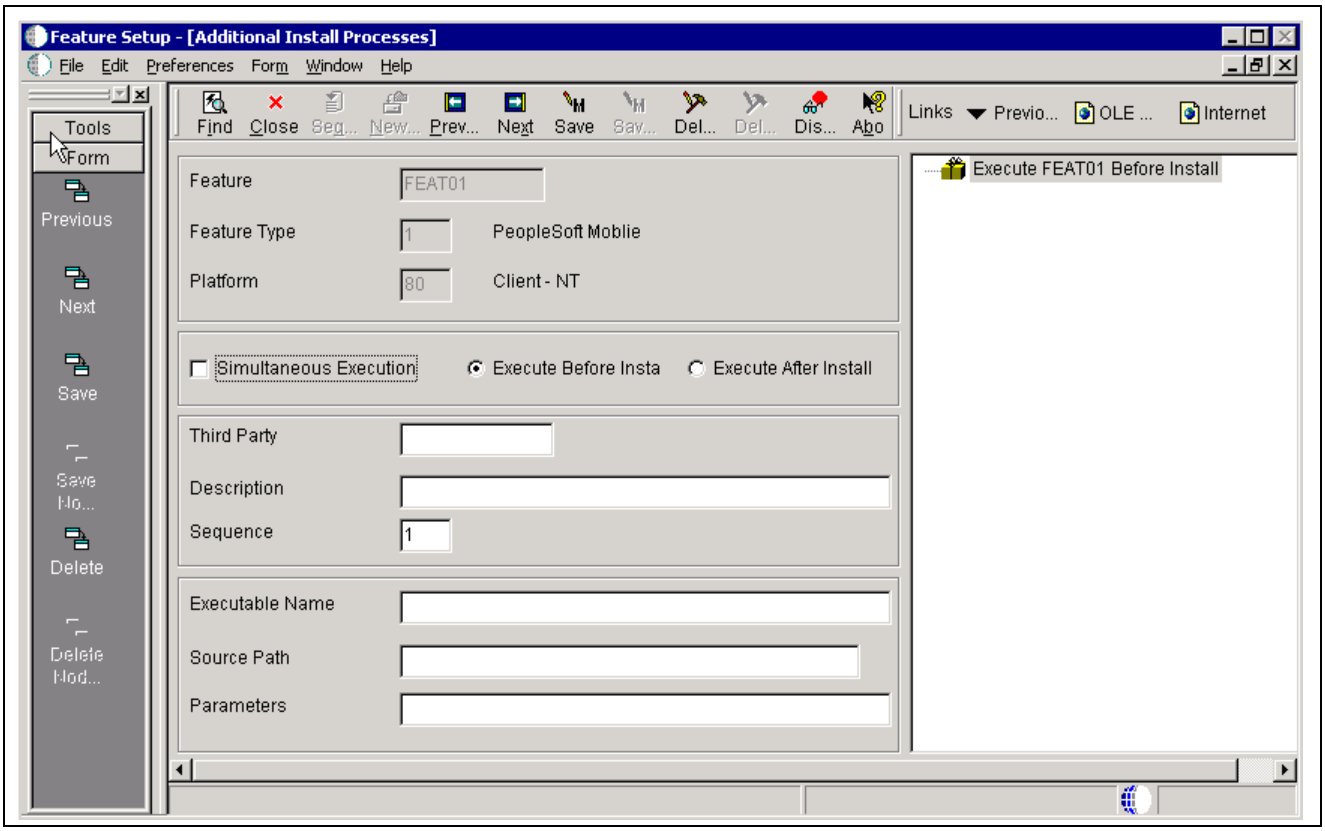
   Parameters The executable parameters that the setup program uses to install the third-party software.

   You can also use this form to modify or delete any previously defined processes. Existing processes appear in the tree structure on the right side of the form. To modify a process definition, select the item on the tree structure and modify any of the fields for the definition. To delete a process definition, select the item and then select Delete or Delete Node After from the Form menu, depending on whether you want to delete a process that is executed before or after the feature is installed. You can run the process either before or after the feature is built.

4. When you are finished adding process information, select either Save or Save Node After from the Form menu, depending on when you want the process to run.
5. Click Next.

Defining Additional Install Processes

If you chose the option to add additional installation processes, the Additional Install Processes form appears.



Additional Install Processes

To define additional install processes:

1. On Additional Install Processes, complete these fields:

   Third Party The name of the third-party component.

   Description A description of third-party software.

   Sequence A number that identifies the order in which this process will run relative to the other additional install processes.

2. These options require you to select two fields that interact with each other. Select the combination for the desired outcome:

   - Synchronous and Execute After Install

     Simultaneous Execution turned off and Execute After Install option turned on. The third-party process waits for the PeopleSoft client install to finish before running.

   - Synchronous and Execute Before Install

     Simultaneous Execution turned off and Execute Before Install option turned on. The PeopleSoft client install will run the third-party process and wait until it finishes before installing the client.

   - Asynchronous and Execute After Install

     Simultaneous Execution turned on and Execute After Install option turned on. The PeopleSoft client install finishes, then starts the third-party process. Neither process waits for the other to finish before proceeding.

- Asynchronous and Execute Before Install

  Simultaneous Execution turned on and Execute Before Install option turned on. The PeopleSoft client install begins, then immediately starts the third-party process and resumes the client install without waiting for the third-party process to finish.

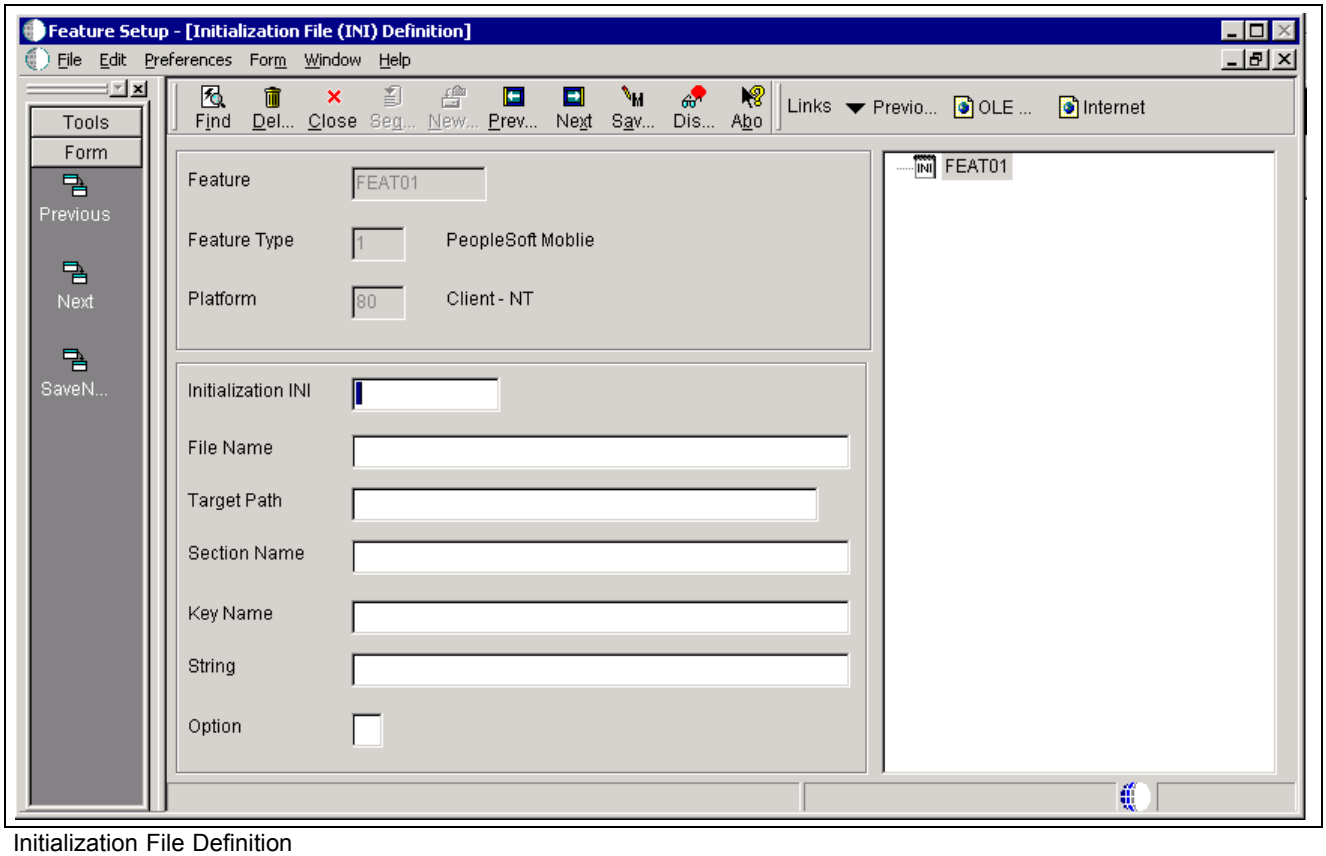
3. Complete the remaining fields:

Executable Name	The name of the program that launches the third-party software.
Target Path	The path to the executable file. Do not include the name of the file.
Parameters	The executable parameters that the system passes to the third-party program.

4. When you finish adding third-party product information, select Save from the Form menu.
5. To delete a definition for a third-party product, select the item on the tree structure and then select Delete from the Form menu.
6. Click Next.

Defining an Initialization File Component

If you selected the initialization file component, the Initialization File (INI) Definition form appears.



Initialization File Definition

To define an initialization file component:

1. On Initialization Files (INI), complete these fields:

Initialization INI	The identifier of an initialization file component.

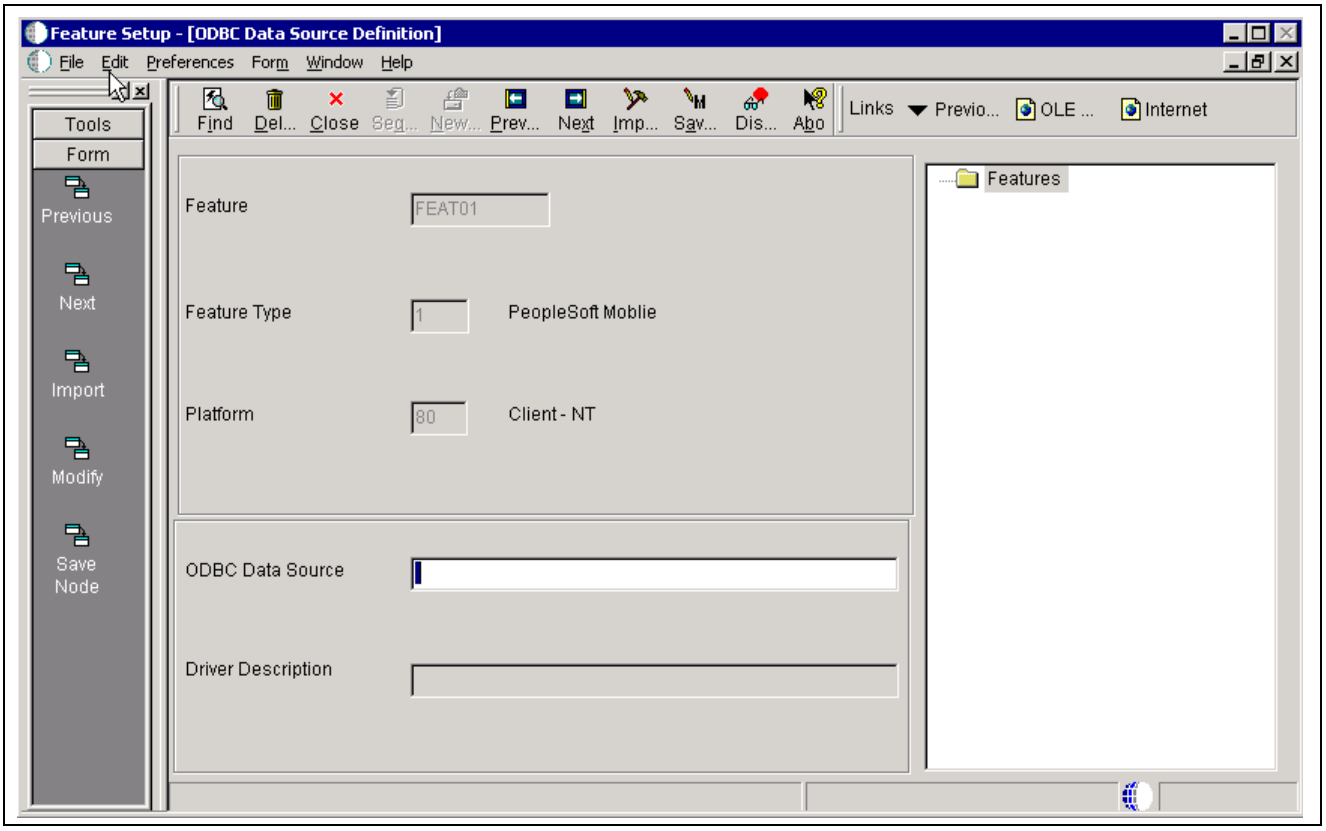
File Name	The name of the initialization file.
Target Path	The path of the INI file.
Section Name	The name of the application section in an initialization file.
Key Name	A key in the initialization file that is to be added, modified, or removed.
String	The value of the key in an initialization file.
Option	The option that identifies the action associated with the key in the initialization file.

You can use this form to modify or delete any previous initialization file definitions. Existing definitions appear in the tree structure on the right side of the form. To modify an initialization file definition, select the item in the tree structure and modify any of the fields for the definition. To delete an initialization file definition, select the item and click Delete.

2. When you finish adding initialization information, select Save Node from the Form menu.
3. Click Next.

Defining a new ODBC Data Source Component

If you selected the ODBC Data Sources component, the ODBC Data Sources Definition form appears.



ODBC Data Source Definition

To create new ODBC data sources:

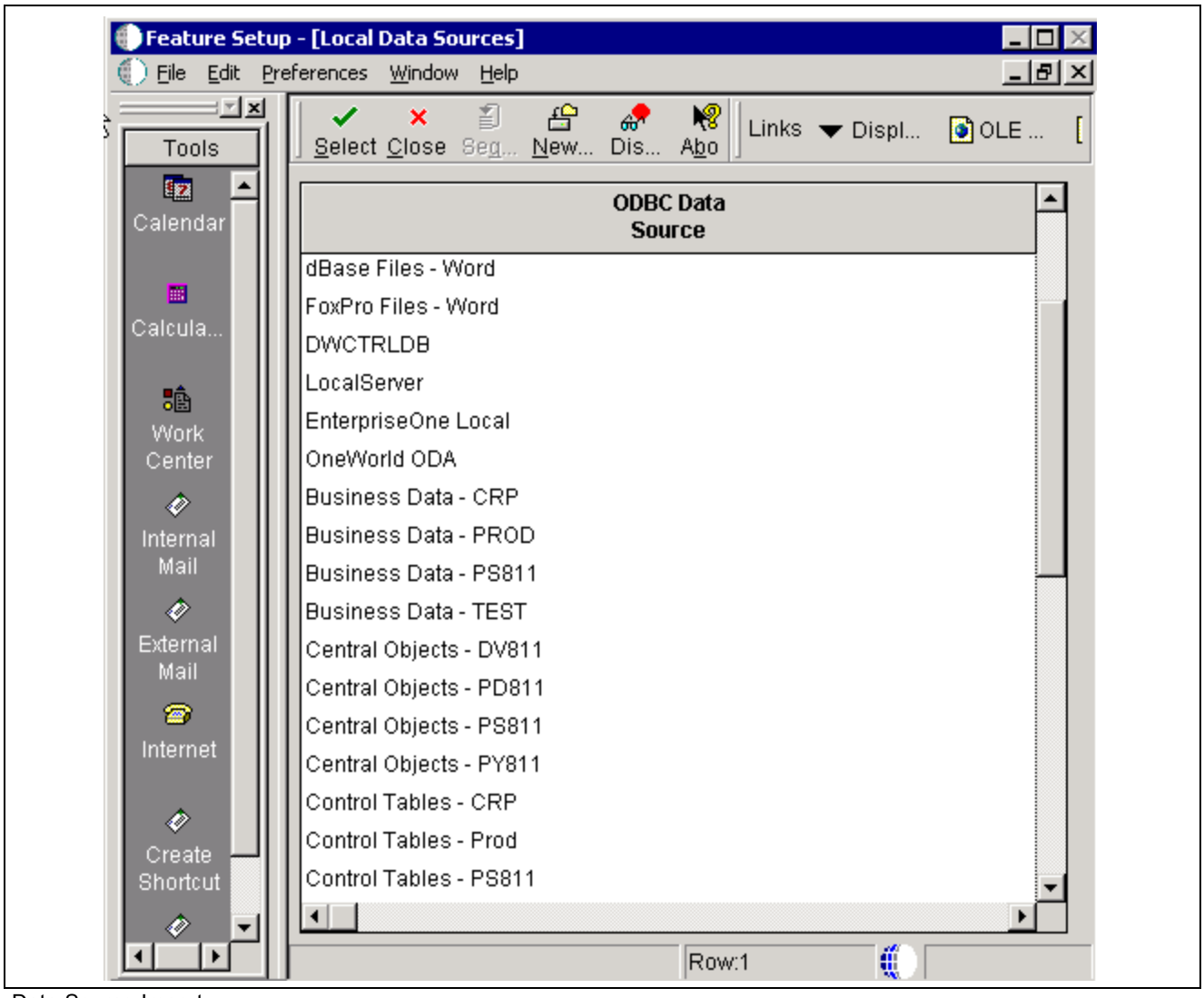
1. On ODBC Data Source Definition, complete the ODBC Data Source field.
2. Select Save Node from the Form menu.

The system activates the Windows control panel applet that displays the ODBC Data Source forms.

3. Enter the data source information on the Windows-based forms.
4. Repeat this process to create additional data sources.

Import an existing ODBC Data Source Component

Access the Local Data Sources form.



Data Source Import

To import existing ODBC data sources:

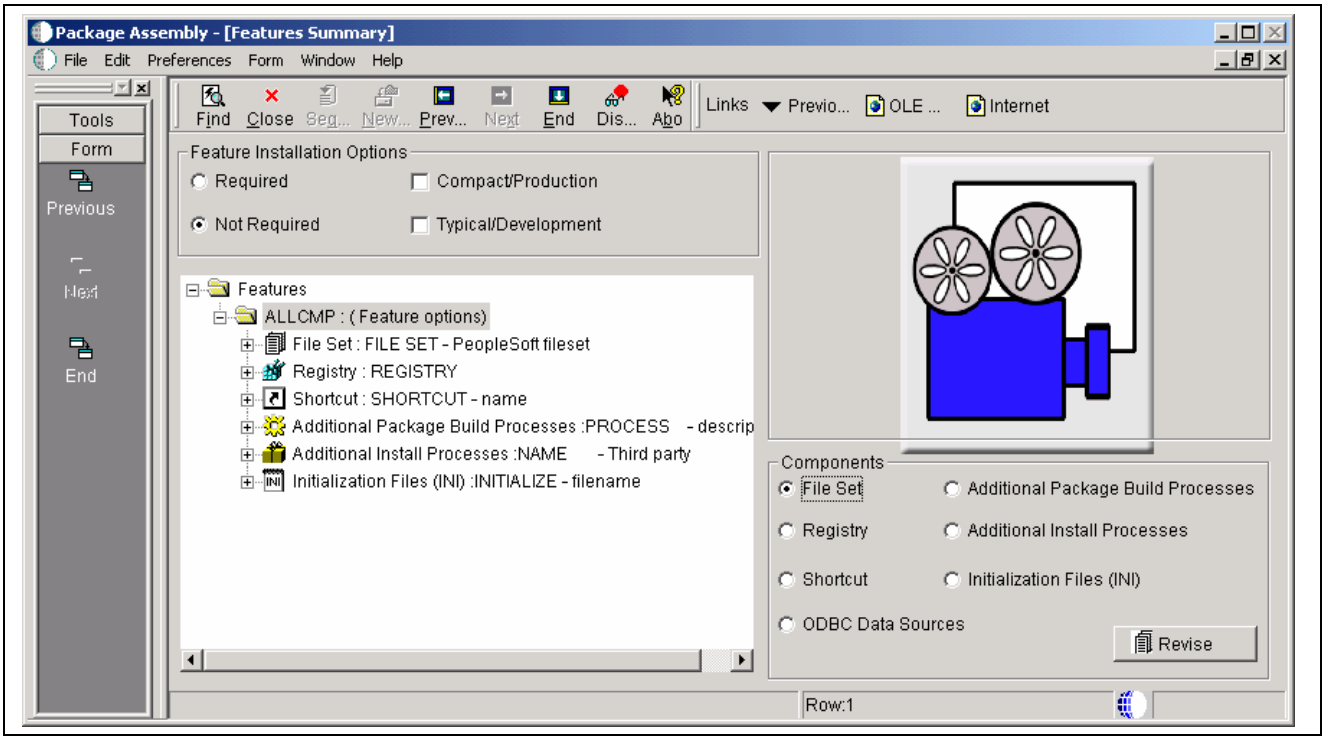
1. On Local Data Sources, use the Control or Shift key to select one or several data sources, and click Select to add the data sources to the feature.

   The ODBC Data Source Definition form reappears.
2. When you are finished adding data source information, select Save Node from the Form menu.
3. Click Next.

4. To modify existing data sources, enter the data source name and then select Modify from the Form menu. The ODBC Data Source Revisions form appears. Use this form to make changes to the data source.
5. When you are finished, click OK to return to the ODBC Data Source Definition form.

Reviewing the Feature Components

Access the Features Summary form.



Feature Summary

To review the feature components:

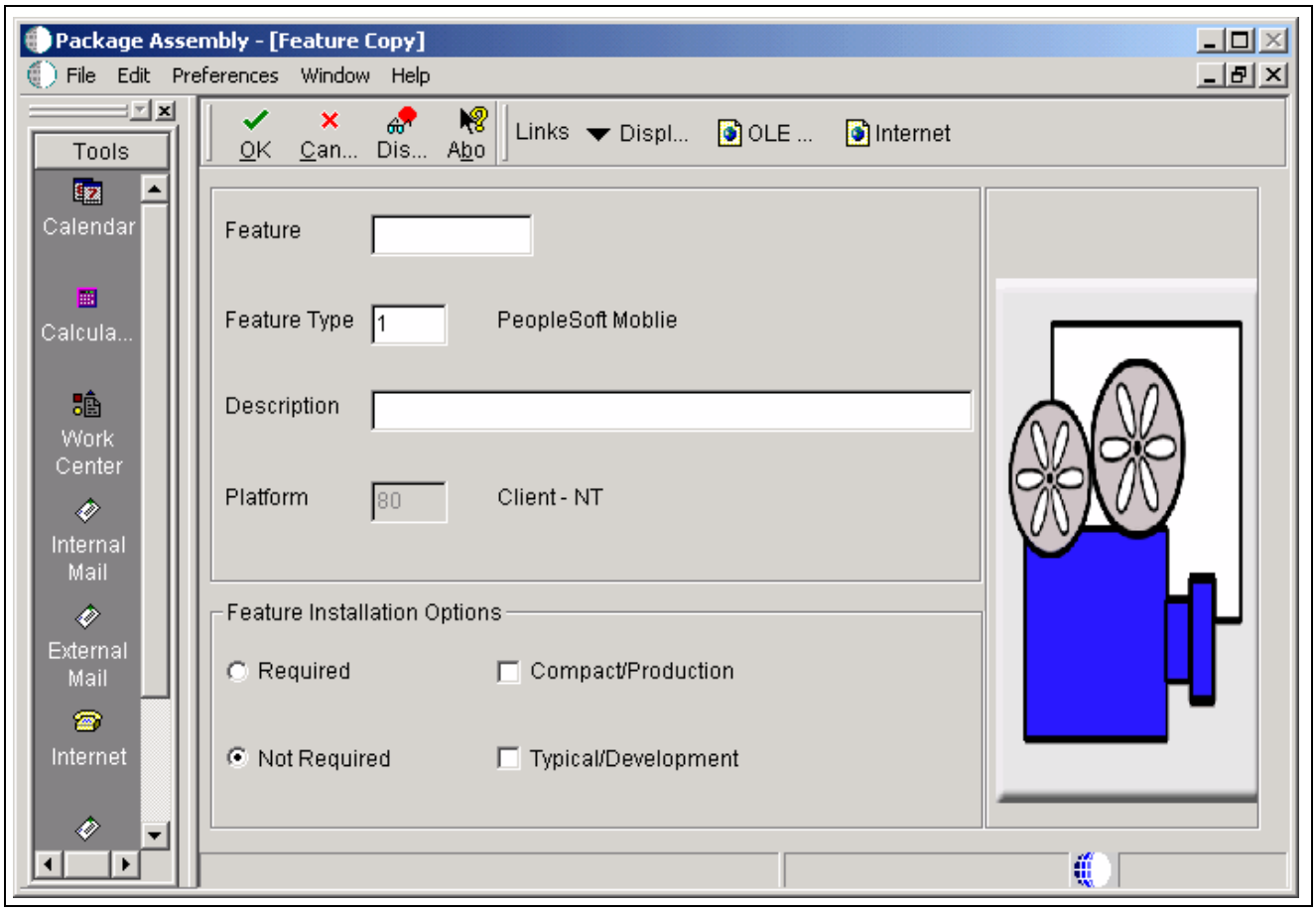
1. On the Features Summary form, select a component in the right pane and click the Revise button to review the information for that component.
2. If needed, change the field values for the selected component and click Save.
3. Repeat the previous steps to modify other components.
4. When you are finished defining the feature, click End.

See Also

Chapter 4, “Assembling Packages,” Revising an Existing Package, page 76

Copying Features

Access the Feature Copy form.



Feature Copy

To copy a feature:

1. On Feature Copy, complete these fields:
   - Feature
   - Feature Type
   - Description
2. Select one:

   Required The installation of this feature is mandatory for both Compact/Production and Typical/Development installs. Inclusion of this feature cannot be overridden when the package is installed.

   Not Required The installation of this feature is optional. Whether the feature is installed depends on the options that you select (Compact/Production and Typical/Development). Inclusion of the feature can be overridden when the package is installed.
3. Select one or both of the options that follow. (If you chose Required, both of these options are automatically turned on.)
   - Compact/Production

     When turned on, this feature is included in a Compact/Production install by default. This option can be overridden when the package is installed if Not Required is also turned on.

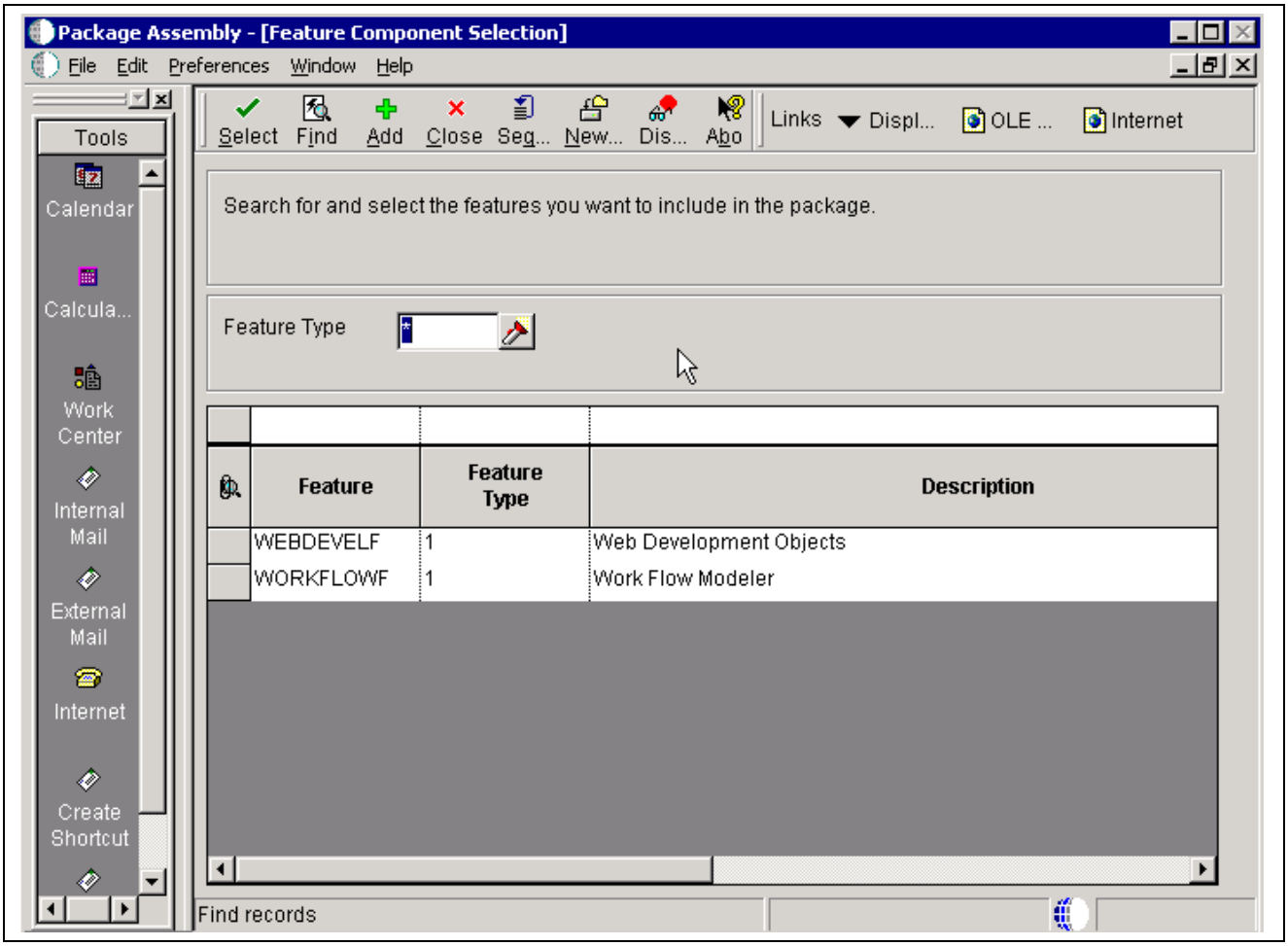
- Typical/Development

  When turned on, this feature is included in a Typical/Development install by default. This option can be overridden when the package is installed if Not Required is also turned on.

4. Click OK.
5. To revise the new feature definition, select the feature and select Revise Feature from the Form menu.

Adding a Feature to a Package

Access the Feature Component Selection screen.



Feature Component Selection

To add existing features to a new package:

1. On Feature Component Selection, click Find to display the list of available features.

   Note. Before the feature is available for inclusion in the package, you must first define the feature.

2. Use one of these methods to select one or more features to include in your package:
   - Select a feature and click the Select button. (Use the Control or Shift key to select multiple features.)
   - Double-click each feature.

3. When you are finished adding features, click Close to return to the Features Component form. The selected features appear.
4. Click Next and complete the remaining forms to finish assembling the package.

To add features to an existing assembled package:

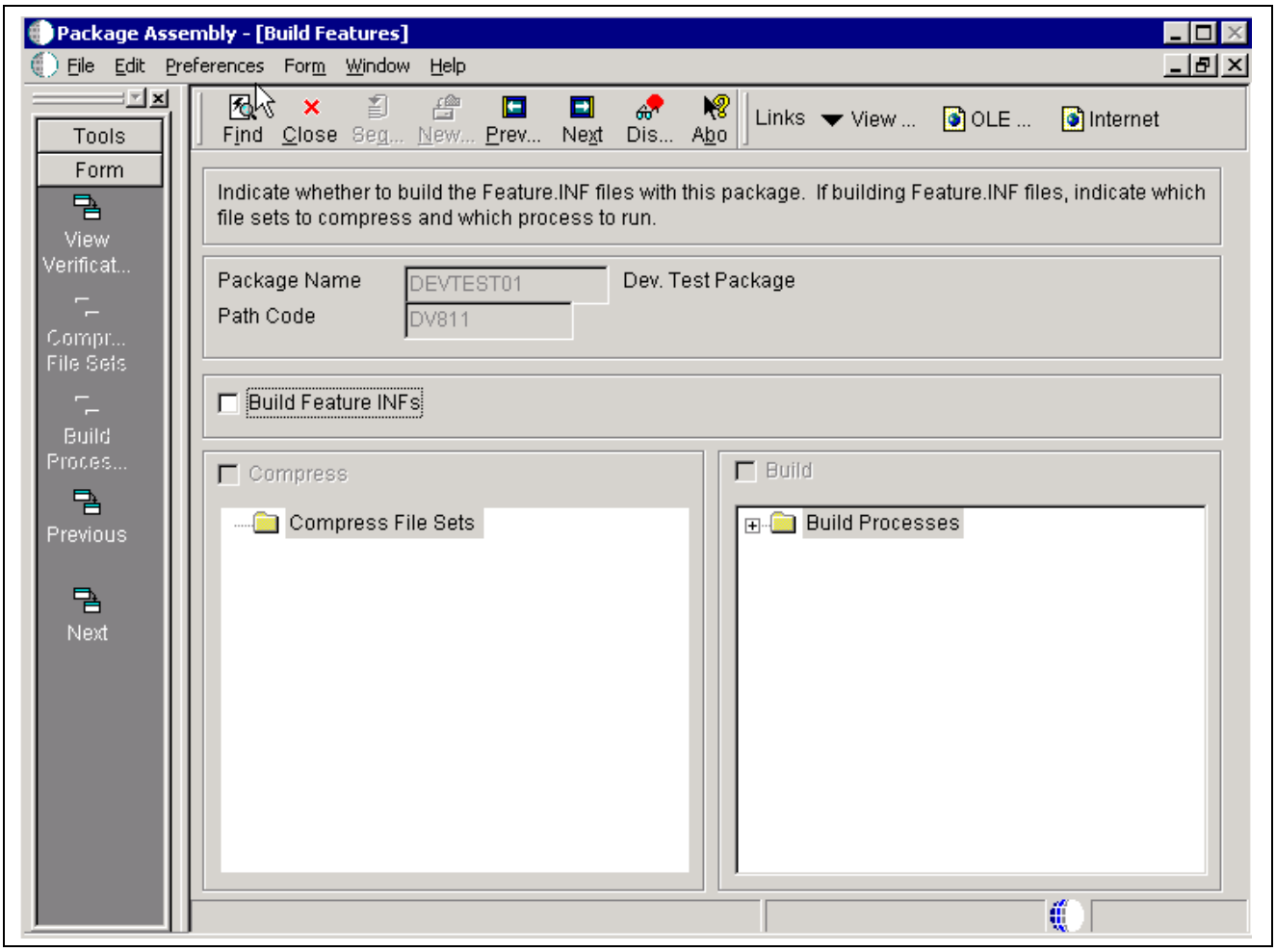
1. On Feature Component Selection, click Find to display defined features.
2. Use one of these methods to select one or more features that you want to add:
   - Select a feature and click the Select button. (Use the Control or Shift key to select multiple features.)
   - Double-click each feature.

   A check mark appears in the left column of each feature that you selected.
3. When you are finished adding features, click Close.
4. Click Close to return to the Package Component Revisions form.
5. Click OK.

Note. To delete a feature that was previously included in the package, on Features Component, select the feature and then click Delete.

Configuring Features During the Package Build Definition

Access the Build Features form during the Package Build Definition process.



Build Features

To configure features for a new package build definition:

1. If you want to build a feature.inf file with the package, select Build Feature INFs.

   When you select this option, the Compress and Build fields become available if file sets or additional package build process components are included in the package.
2. Continue with one or both of these tasks:
   - To compress file sets
   - To build processes

To compress file sets:

1. Select Compress, and then select Compress File Sets from the Form menu.
2. On File Set Selection, select each feature that you want to include by choosing a file set and clicking Select.
3. When you are finished selecting file sets, click Close.
4. Continue either by completing the steps in the task To build processes, or by clicking Next and completing the remaining forms to finish defining the package build.

To build processes:

1. To build processes, select Build, and then click Select Build Processes.

2. On Build Processes Selection, select each process that you want to build by choosing a process and clicking Select.
3. When you are finished selecting processes to build, click Close.
4. From the Form menu, select Build Processes and manually select each process to run during the package build.

   You must complete this step or none of the processes will run, even though they are included in the feature.
5. Click Next and complete the remaining forms to finish defining the package build.

Configuring Features for an Existing Package Build Definition

Access the Build Features form from an existing package.

To configure features for an existing package build definition:

1. Modify or add to any of these existing build feature settings:
   - Build Feature INFs
   - Compress
   - Build
2. If you select Compress, select Revise File Sets from the Form menu to modify file sets.
3. When you are finished modifying file sets, click Close.
4. If you chose Build, click Revise Processes to modify processes.
5. When you are finished modifying processes, click Close.
6. If you selected Build, from the Form menu, select Build Processes and manually select each process to run during package build.

   You must complete this step or none of the process will run, even though they are included in the feature.
7. Click OK to complete the package build definition.

Viewing Package Build Records and Resubmitting Builds

This section provides overviews of package build records and build status, and discusses how to:

- View package build history
- View log files
- Resubmit a package build
- Change the build status
- Reset the spec build and pack build status

Understanding Package Build History

The Package Build History program (P9622) enables you to view information pertaining to the build process, including the options and objects that you specified when you created the build definition. This program provides build information including:

- Package name
- Path code
- Date and time built
- Name of the server for which the package was built
- Current build status and status description
- Current status of selected specification tables
- Number of specifications written
- Package records written and read

The View Logs option on the Form menu enables you to view four logs that contain additional information about the build process. Refer to these logs in the event that the build does not finish successfully and you need to review the errors that occurred during the build.

If a build does not finish successfully, you can use the Resubmit Build option to resume the build from the point at which the process stopped. Only the business functions and objects that did not build successfully will be built; the entire package will not be rebuilt.

In some cases, if a build is interrupted or otherwise unable to finish, you might need to reset the build status from Build Started to Build Definition Complete. Unlike the Resume Build feature, which continues the build from the point at which it failed, resetting the status enables you to start the build process from the beginning.

Package Build History

The system maintains a history of the package build in the Software Package Build Detail - History table (F96225). This table contains details about the package build statuses of any package components.

If you encountered errors during the build process and your package failed to build successfully, you can resubmit the package and continue building at the point at which the build failed. In this situation, the system reviews the F96225 table and rebuilds only the business functions or other package components that have a status of Not Built or Error. The system builds only the package components that failed, not the entire package. This feature can save you a tremendous amount of time, especially if only a few package components failed to build successfully.

If you originally specified package compression, when you resubmit the package to resume building, the system automatically compresses the directories after it successfully builds the package.

Viewing Logs

After you build your package, you can view logs that list any errors that occurred during the build process. In particular, you can view these logs:

- Package statistics log
- Package build log
- Business function errors log
- Missing business function source errors log

Each log contains a header, which includes the package name, date, build machine, and path code.

Where to Find the Error Logs

To review error logs without using the Package Build History program (P9622), locate the desired log in the correct directory. Error logs are stored on the deployment server in directories that are subordinate to the directory for the package itself. The package build log is stored in the package directory. The package statistics log, business function source errors log, and missing business function source errors log are stored in the work directory for the package.

You can view the error logs by accessing the appropriate directory and opening the log with Microsoft Notepad or a similar application that enables you to display text files.

In the following examples, PD811FA is used as the package name. To determine your actual directory, substitute your package name for PD811FA.

- Package statistics: \PD811FA\work\buildreport.log
- Package build log: \PD811FA\builderror.log
- Business function errors log: \PD811FA\work\buildlog.txt
- Missing business function source log: \PD811FA\work\NoSource.txt

The Package Statistics Log

The package statistics log summarizes the outcome of the package build, showing statistics for the directories in the package, including the size and file count of each directory. This log displays a complete build that you can use to review your build directories. The report shows a breakdown of the files in the spec directory and the size of each spec file, as well as the total count and size. You can use this log to verify that the package built successfully.

The Package Build Log

The package build log appears in the package name directory. This log lists the steps completed in building the package, as well as any errors that occurred during the package build process. It also describes the steps involved in building the package. The final page of the log tells you whether the package was successfully built.

The Business Functions Errors Log

The business functions errors log enables you to view any errors that occur while business functions are being built. The final page of the log describes whether the business functions were successfully built or were built with errors. Business functions that appear on this report might be business functions that are still in development and have not yet been checked in. Business functions that have never been checked in do not have source, and therefore, are listed in the missing business function source errors log.

The Missing Business Function Source Errors Log

The missing business function source errors log describes any business functions in the package that are defined in the Object Librarian and have a record, but could not be built because no source existed.

Server Logs

All compile logs for the enterprise server are located on the server itself in the source directory of the DLL in which the object belongs. For example, suppose that you want to see the log for the Sales Order Entry Master Business Function (B4200310) in the package PACKAGE1 on an HP 9000 for which the BuildArea is /u02/PeopleSoft/packages. The system creates a file called /u02/PeopleSoft/packages/PACKAGE1/CompileLogs/CDIST/b4200310.log (or b4200310.err if there are errors) because B4200310 is in the CDIST.DLL.

If the system could not link the CDIST.DLL (shared library) on the HP 9000, it would create a file called /u02/PeopleSoft/packages/PACKAGE1/obj/CDIST/CDIST.log.

On the iSeries, logs for business functions that failed to compile are members in a file called FAILED in the package library. Using the previous example, you would review member B4200310 of the FAILED file in library PACKAGE1.

Understanding the Build Status

In some cases, you might need to attempt to rebuild the package rather than resume the build from the point at which the build failed. Before you can do so, you must first change the status of the package build from Build Started to Build Definition Complete.

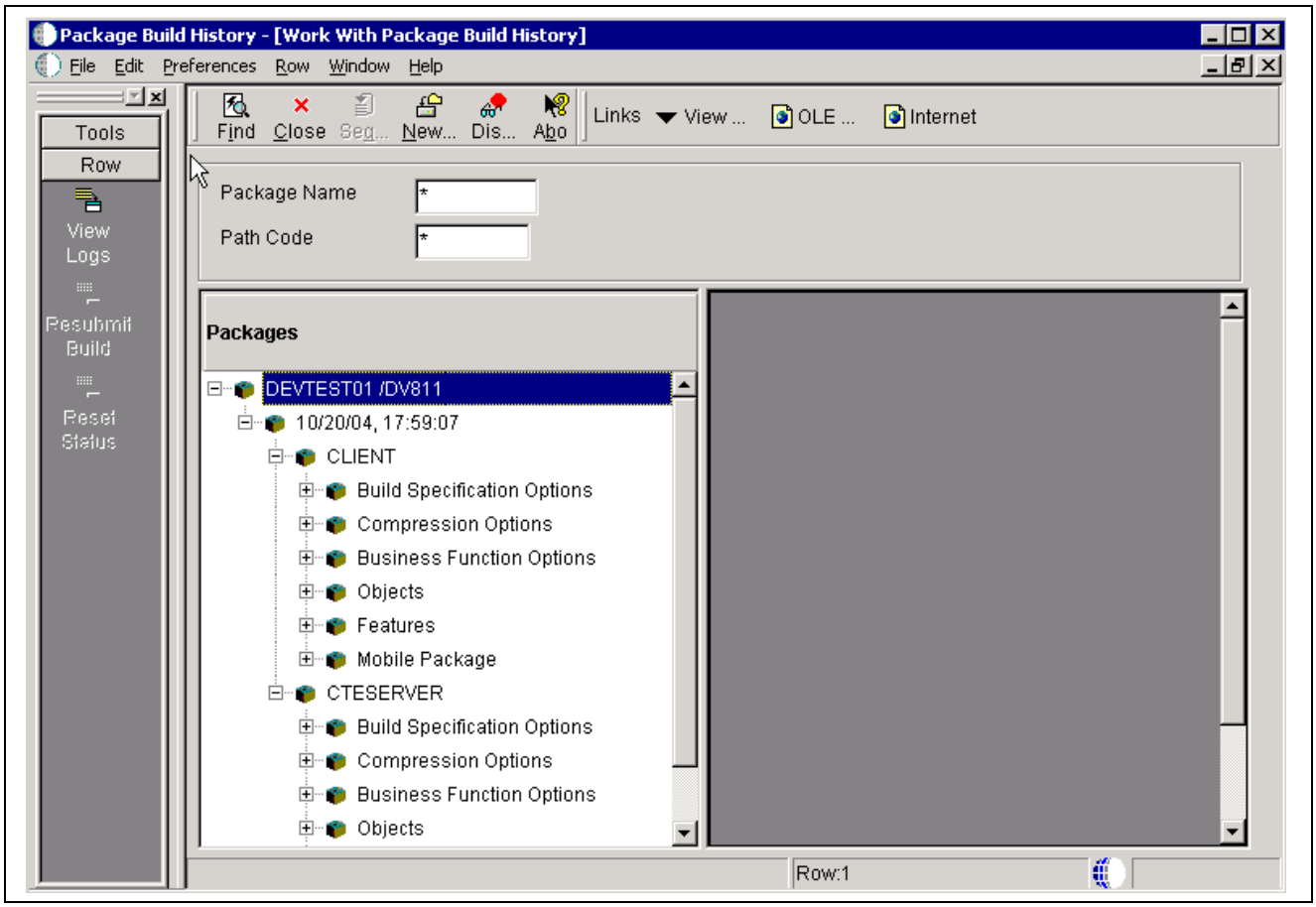
When you reset the status of the package build, you can reset the status for the server only or for all servers and client workstations for which you want to build the package.

Forms Used to View Package Build History and Logs

Form Name	Form ID	Navigation	Usage
Work with Package Build History	W9622A	From the Package and Deployment Tools menu (GH9083), select Package Build History.	Display information about the current build status and build options for selected computers.
View Logs	W9622B	From the Package and Deployment Tools menu (GH9083), select Package Build History. Select View Logs from the Form menu.	Check logs for errors that occurred during the build process.
Work with Package Build Definition	W9621L	From the Package and Deployment Tools menu (GH9083), select Package Build. W9621L	Change the package build status.
Reset Build Status	W9622C	From Work with Package Build History, find the package for which you want to reset the statuses, expand the package, and select an individual item. Select Reset Status from the Row menu.	Reset the spec status and pack status for a package to the statuses that you specify.

Viewing the Package Build History

Access the Work with Package Build History form.



Work with Package Build History

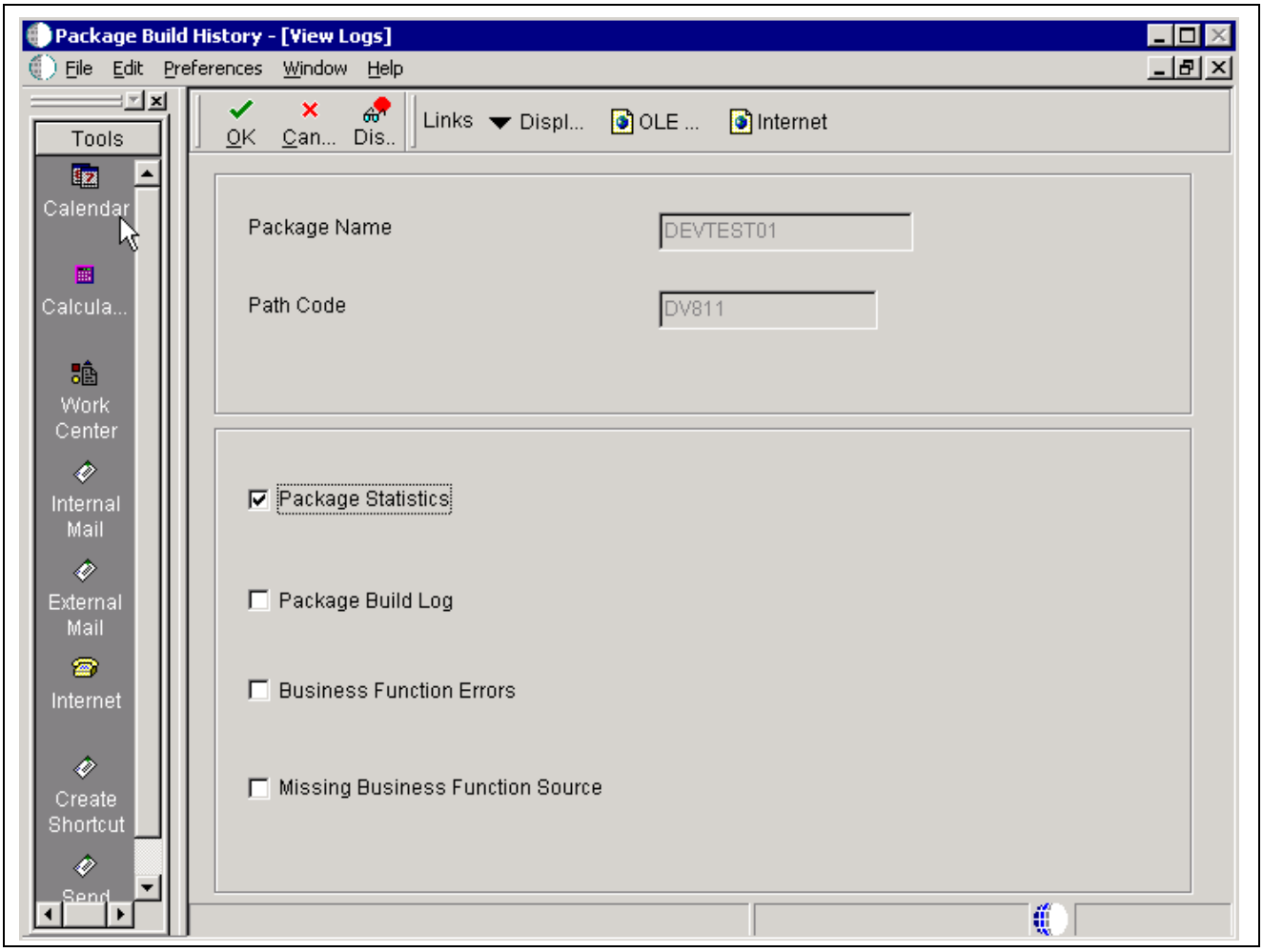
To view package build history:

1. On Work with Package Build History, select CLIENT or the server name to display information about the current build status for those computers. You can also expand the tree to view this information:
   - Build specification options
   - Compression options
   - Business function options
   - Objects

     These options and objects are those that you specified when you created the build definition for the package. For example, if you chose to build only selected specifications, you can determine the status for each specification, as well as other pertinent information.
2. When you are finished viewing build history information, click Close.

Viewing Log Files

Access the View Logs form.



View Logs

Package Statistics This option enables you to view count and size statistics for the package directories that were built.

Package Build Log This option enables you to view errors that may have occurred during a package build. These errors could have occurred while building the specification files or the objects for the package.

Business Function Errors This option enables you to view the results of the business function build for this package. Both errors and warnings display in this report. A summary appears at the end of the report that indicates how many errors and warnings occurred for each dll. Use this information to determine if a rebuild is necessary.

Missing Business Function Source This option lists all source members that were not available when the business function was created. The program attempted to find these members because each had a record in the Object Librarian table (F9860). However, a matching source could not be found in the source directory. To resolve these errors, either delete the Object Librarian record or provide a source member.

Resubmitting a Package Build

Access the Work with Package Build History form.

To resubmit a package build:

1. Select one of these options to find the package you want to resubmit.
   - Select a specific server to resubmit only the builds for that server
   - Select the CLIENT heading to resubmit only the workstation builds
2. Select Resubmit Build from the Row menu.

   If you generated NERs when you initially submitted the build, the system displays a window that asks whether you want to regenerate the NERs.
3. Click OK to regenerate NERs, or click Cancel to skip this process.

   Note. If you do not want to regenerate NERs, you can prevent this window from appearing by entering *2* in the Generate NER processing option for the Package Build History program.
4. Select one of these destinations for the build report, and click OK:
   - On Screen
   - To Printer

   The form closes, and the system begins to build the package. Build time varies, depending on the number and size of the items in your package. When the build is finished, the report either appears on the screen or prints, depending on the destination you specified.
5. Review the report to verify that the system successfully generated all components in your package. If the report indicates any errors, review the error logs for more detail.

See Also

Chapter 6, "Deploying Packages," page 125

Changing the Build Status

Access the Work with Package Build Definition form.

To change the build status:

1. Find the package for which you want to reset the status. Underneath the package name, select the server or servers and client workstation for which you want to build the package.
2. From the Row menu, select Advanced.
3. On Advanced Revisions, click Reset to change the status of the package build from Build Started to Build Definition Complete.
4. Click OK.
5. If desired, select the package name and select Submit Build from the Row menu.
6. The program asks whether you want to delete the current build or to continue without deleting it.

Reset the Spec Build and Pack Build Statuses

Access the Reset Build Status form.

To change the spec and pack statuses:

1. Enter the desired statuses in the Spec Build Status and Pack Build Status fields.

Both of these fields have a visual assist feature to help you determine the available statuses.

Note. The values of these two fields are dependent on each other. If you change one value, be sure you understand the dependency on the other value.

2. Click Reset.
3. Click OK.

CHAPTER 6

Deploying Packages

After you have assembled, defined, and built a package, you can select from several ways to deploy the package to workstations and servers.

This chapter describes the deployment process and discusses how to:

- Define deployment parameters
- Deploy packages
- Deploy server packages
- Use push installation
- Install Workstations from CD
- Deploy data

Understanding Package Deployment

After you build a package, you can select from several methods of deploying the package to workstations and servers throughout your enterprise. For workstations, the method that you select depends on whether PeopleSoft EnterpriseOne is already installed on the workstation.

This section discusses:

- Deploying to workstations without EnterpriseOne
- Deploying to workstations with EnterpriseOne already installed
- Deploying to servers
- Deploying to tiered locations
- Deploying to workstations from CD

Deploying to Workstations without EnterpriseOne

If PeopleSoft EnterpriseOne is not currently installed on a workstation, you can deploy the package through the Workstation Installation program. You can use Workstation Installation to deploy full packages; but you cannot use Workstation Installation to deploy an update package to a workstation on which EnterpriseOne is not installed.

Workstation Installation retrieves items specified in the package. A package is like a bill of materials with instructions that describe from where the system retrieves all of the necessary components that the Workstation Installation program deploys to the local workstation. This program can be run interactively (initiated by a person at a workstation) or in silent mode and scheduled through the push installation feature.

If you use the push installation feature, you can use Package Deployment to deploy the package. Push installation enables the administrator to initiate the installation of a package from the deployment server to workstations without any user interaction. To use this feature, the push installation *listener* application must be installed on the workstation, and the machine must be defined through the Deployment Locations Application program (P9654A).

See Also

EnterpriseOne PeopleTools 8.11 Installation Guide

Deploying to Workstations with EnterpriseOne Already Installed

To reload a new package on workstations on which EnterpriseOne is already installed, use one of two methods:

- Workstation Installation (for full packages)
- Deployment Director (P9631) (for full and update packages)

After you assemble and build a package, use Deployment Director to schedule the package for deployment to individual workstations or to selected groups. On the specified deployment date, when the users who are scheduled to receive the package sign on, they are given the opportunity to load the package.

Unless you are using the Push Installation feature, Deployment Director requires that EnterpriseOne be already loaded on the workstation. You can schedule a new full package to replace the existing package, or an update package to be merged with the existing package on the workstation.

Both deployment methods have advantages. Workstation Installation is a good method to use when you want to install a package immediately or soon after it is built, without having to schedule the package. Alternatively, Deployment Director is useful if you need to control when the package becomes available, if you want to make the package installation mandatory, or if you want to deploy the package to servers as well as to workstations.

Deploying to Servers

Servers receive the same package that you build for the workstation, but in a different format. When you assemble the package and create the package build definition, you can specify the servers to which you want to build and deploy the package. To deploy the package, you use the Deployment Director application (P9631), which uses the same scheduling mechanism to deploy packages to workstations. In fact, you can easily schedule deployment to both client workstations and servers on the same form. You cannot use the Push Installation feature to deploy to servers.

Deploying to Tiered Locations

Multitier deployment enables you to install software on workstations from more than one deployment location and more than one deployment machine. Use this deployment method if your site has more than 50 workstations performing software installations per day, or when workstation installations over your WAN are too slow.

Deploying to Workstations from CD

If your system has a CD writer, you can define the CD writer as a deployment location. Essentially, you define the CD writer as a pseudo deployment server from which you can copy a package onto a blank CD. You can then use this CD to install the software on workstations by using the Workstation Installation program that is included on the CD.

Defining Deployment Parameters

This chapter provides an overview of the deployment configuration and discusses how to:

- Define machines
- Define locations
- Define package deployment groups
- Revise package deployment groups

Understanding Deployment Parameters

Before you deploy packages, you must identify the workstations, servers, groups, or locations that will receive your package. Identifying these ensures that, when you are ready to schedule packages using the Deployment Director, the machines, groups, or locations that you want to receive your package will be available as package recipients.

A deployment group is a group of workstations that are classified by a criterion such as job function, team, or any other grouping that you specify. For example, you might have a software development group, a testing group, a production group, and so on. The Package Deployment Groups Revisions program (P9652A) enables you to define or revise groups that include several workstations.

A location is a group of workstations and servers that corresponds to a physical location. For example, you might have locations for Corporate and Branch, or for Building 5 and Building 7. Locations are also useful if you use multitier deployment or deploy across a wide area network. In this case, you might define a location for each of your geographic locations. The Deployment Locations Application program (P9654A) enables you to define or revise machines and locations in your enterprise.

Both of these applications simplify the deployment process when you need to deploy a package to several users. Rather than requiring you to schedule deployment to each workstation or server, you can schedule deployment according to location or group.

When you enter a machine definition, you are really defining its usage in the configuration. For example, you can use a deployment server as a data server. When you enter machine definitions, consider these recommendations:

- A Java application server (JAS) can be defined only as a Java application server, not as a data server, enterprise server, and so on.
- A deployment server should not be used as a workstation.
- A deployment server can be used as a data server.
- A deployment server should not be used as an enterprise server for tuning and performance reasons.

Locations

In some cases, an enterprise might span several buildings, cities, or even countries. In these situations, you might deploy a package to a location rather than to individual workstations and servers. Then, a secondary deployment server at each location can deploy the package to the workstations and servers at that location.

The larger your enterprise, the more you can benefit from creating and deploying to locations. If you use multitier deployment to deploy packages to remote locations, the concept of locations is crucial.

In PeopleSoft EnterpriseOne, a *location* is essentially a user-defined group of machines, databases, and environments. In some cases, the location is an actual physical location connected by a wide area network (WAN), such as when you have remote offices that are geographically separate from your main office. For example, a location might be a floor in your office building, a separate building on your corporate campus, a branch office across town, or a facility in another city.

After you create a new location, you can add workstations and servers for that location by defining the machine names that are associated with that location.

The topmost location that appears when you launch the Deployment Locations Application program (P9654A) is the base location. You cannot change or remove this base location, but you can create or revise locations that are subordinate to it.

When you create a location that is subordinate to another location, the original location is the parent location, and its subordinate location is the child location. For example, if you have a location called Seattle and then create a location called Redmond that is subordinate to Seattle, Seattle is the parent location and Redmond is the child location.

Deployment Groups

You can create a deployment group based on department, team, or function. For example, you might have an administration group, a testing group, a production group, and so on.

Package deployment groups are particularly useful in large enterprises in which scheduling a package for deployment to several individual workstations is very time consuming. In these environments, you can deploy packages much more quickly when you use deployment groups.

A group can contain a subgroup (a group within a group). For example, you might have a group called Quality Assurance that is a subgroup of the larger Development group.

You can help the person who builds and schedules packages by creating easily identifiable names for deployment groups. For example, for a group that includes quality assurance specialists who are responsible for testing, name the group Testing, rather than Green Team.

See Also

Chapter 7, “Setting Up Multitier Deployment,” page 159

Forms Used to Define Deployment Parameters

Form Name	Form ID	Navigation	Usage
Machine Revisions	NA	From the Package and Deployment Tools menu (GH9083), select Machine Identification. Click Find. Underneath the appropriate location, select the type of machine that you want to add. Click Add to add a new machine, or click Select to revise an existing machine definition.	Create or revise the machines used in the deployment process.
Location Revisions	NA	From the Package and Deployment Tools menu (GH9083), select Machine Identification. Select the current location for which you want the new location to be subordinate. (For example, if you are adding a new remote location, click the Remote Locations button.) Click Add to add a new location, or click Select to revise an existing location.	Create or revise deployment locations.
Deployment Group Revisions form	NA	From the Package and Deployment Tools menu (GH9083), select Machine Group Identification. Click Add to add a new deployment group, or click Select to revise an existing group.	Create or revise package deployment groups.

Defining Machines

Before you use the Deployment Director to deploy a package to individual client workstations, verify that each machine that will receive the package has a record in the Machine Master table (F9650).

Select one of these ways to populate this F9650 table:

- Manually. For a machine that no user has ever used to sign on to PeopleSoft EnterpriseOne, use the Deployment Locations Application program (P9654A) to manually enter a record in the F9650 table.
- Automatically. The system automatically creates a record in the F9650 table when a user on a new machine signs on to EnterpriseOne for the first time. (The system also automatically updates existing records in the F9650 table each time a user signs on to the workstation.)

The simplest way to populate the F9650 table is to have all users on new machines sign on. In cases where you need to deploy a package before the users can log on, you must manually enter machine information. The Deployment Locations Application program enables you to perform this task.

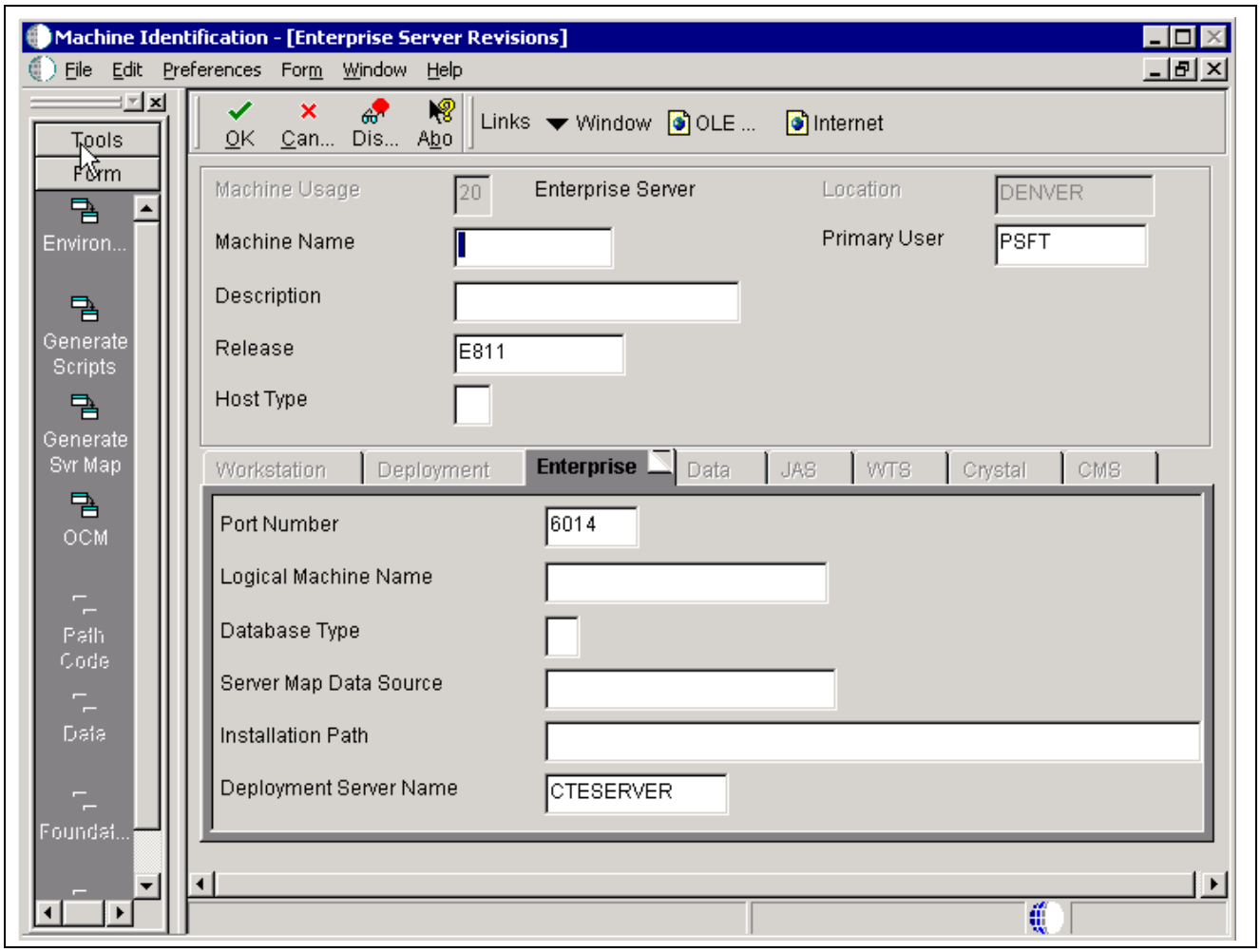
In addition to defining workstations, you can also use the Deployment Locations Application program to enter or revise definitions for these machines:

- Deployment servers
- Enterprise servers
- Data servers
- Java application servers
- Windows terminal servers

You can define a machine to be a workstation and a data server or a data server only, but no other combinations are enabled. For example, you cannot define a machine as a workstation and a deployment server.

You can enter or revise definitions for these machines in multiple locations, including remote locations.

Access the Machine Revisions form.



Enterprise Server Revisions

Workstation

Deployment Server Name The name of the specific server that is being used for deployment.

When you define a secondary deployment server, options on the Form menu enable you to select path codes, data items, foundation modules, and help items. (These options are not available for the primary deployment server.)

Deployment Server

Primary Deployment Server

This field specifies whether a deployment server is the primary deployment server for a specific location.

If you have set up a primary deployment server, you cannot access the Primary Deployment Server field when you define a new deployment server. You can change the value in this field only when you revise the primary deployment server definition or when you change your primary deployment server to a secondary server. In this case, you can specify a different server as your primary deployment server

Server Share Path

The shared directory for this path code. The objects that are stored on a file server will be found in this path.

Enterprise Server

Server Map Data Source

The name that identifies the data source.

Deployment Server Name

The name of the specific server that is being used for deployment.

When you define a secondary deployment server, options on the Form menu enable you to select path codes, data items, foundation modules, and help items. (These options are not available for the primary deployment server.)

Database Type

The type of database.

Logical Machine Name

The logical machine name assigned to this unique machine and port. A machine can be a workstation. Because you can have more than one instance of EnterpriseOne running on a given machine, you must assign a logical machine name that identifies the unique physical machine name and port where this instance runs.

It is recommended that the logical machine name represent the release and purpose of the machine, such as Financial Data Server-B73.3 or Distribution Logic Server-B73.3.

Installation Path

Path on which EnterpriseOne is installed

Port Number

Identifies the port for a given instance of PeopleSoft EnterpriseOne. Because the JDE.ini file controls the port to which a workstation will connect, for workstations this port number is for reference only.

Data Server

Data Source Type

The type of database.

JAS Server

Protocol

Method of communication (for example, http)

Server URL

URL to the Web Server

Port

Port number of the Web Server

Default Login

Login path

Installation Path

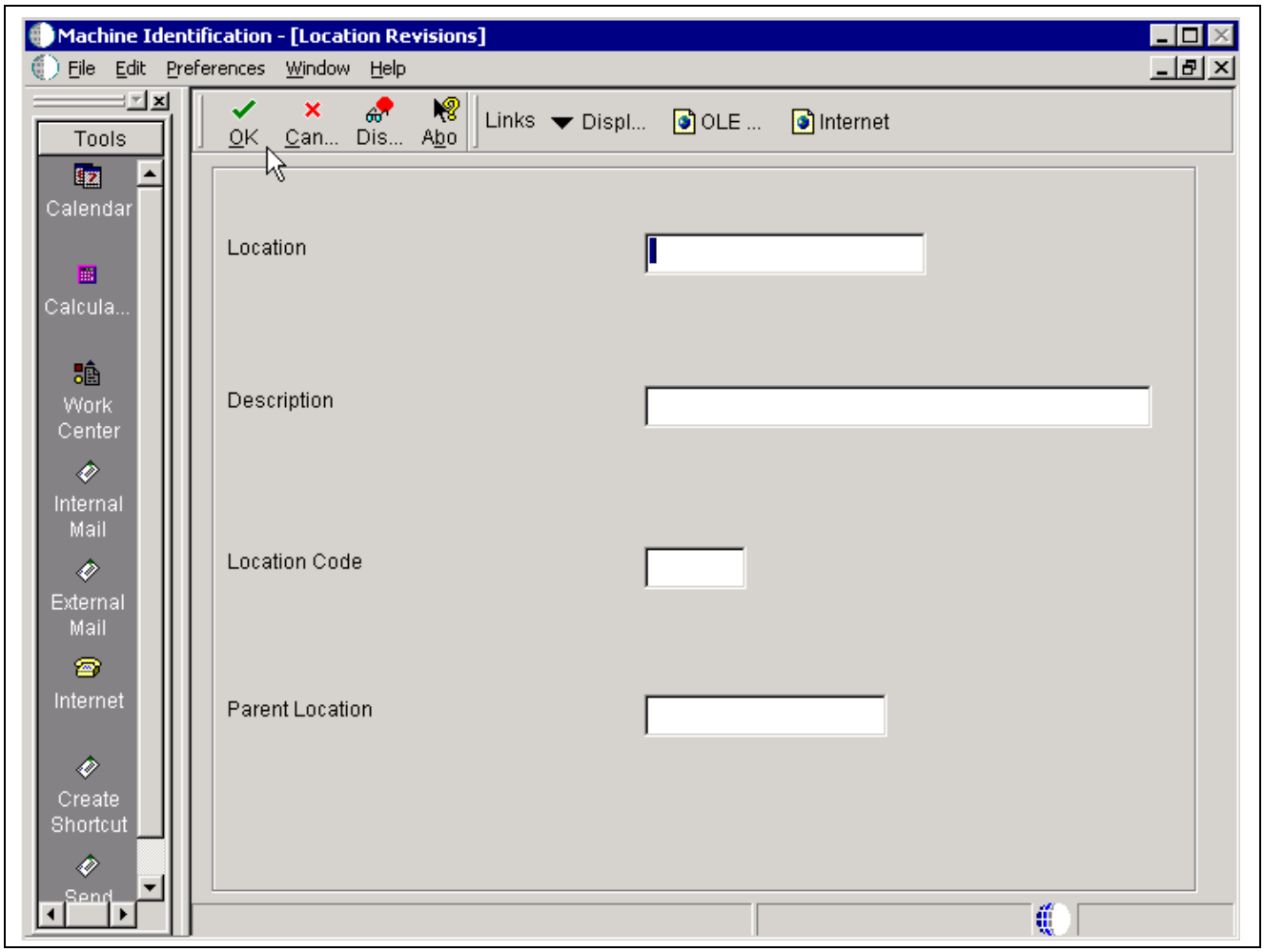
Path on which EnterpriseOne is installed

WTS

Installation Path Path on which EnterpriseOne is installed

Defining Locations

Access the Location Revisions form.



Location Revisions

Location The name of the deployment location.

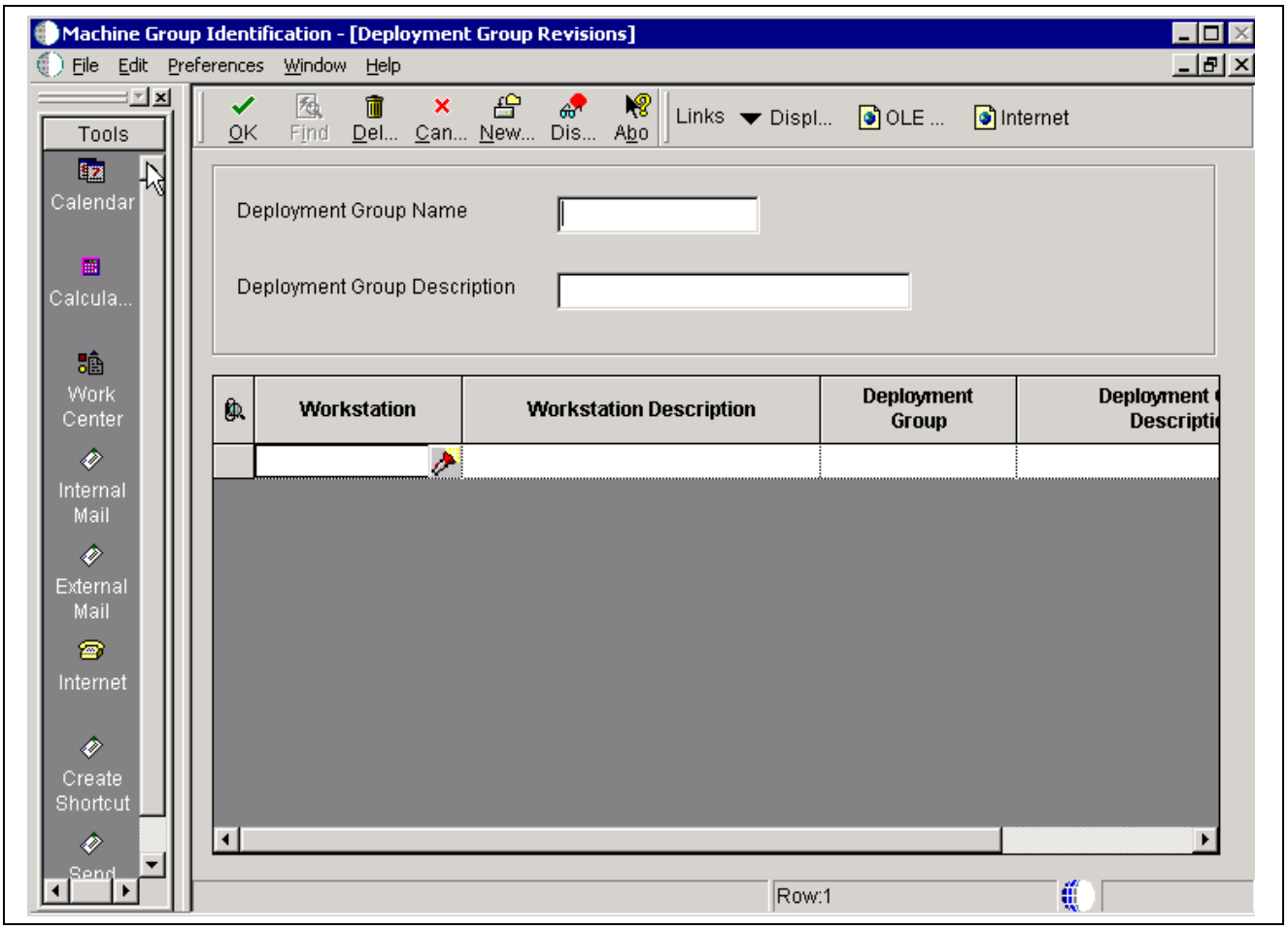
Location Code This field represents the current location for system deployment.

Parent Location The name of the parent location.

Defining Package Deployment Groups

To define a package deployment group:

Access the Deployment Group Revisions form.



Deployment Group Revisions

Deployment Group Name A profile that you use to classify users into groups for system security purposes. You use group profiles to give the members of a group access to specific programs.

Some rules for creating a profile for a user class or group are as follows:

The name of the user class or group must begin with an asterisk (*) so that it does not conflict with any system profiles.

The User Class/Group field must be blank when you enter a new group profile.

Deployment Group Description The description for the selected deployment group.

Workstation The Location or Machine Key indicates the name of the machine on the network (server or workstation).

Workstation Description A user defined name or remark.

Deployment Group This field represents a group that is defined to be part of a parent group.

Revising Package Deployment Groups

Access the Deployment Group Revisions form.

- When you revise an existing group, you cannot change the group name, but you can change the description.
- To add to the group, select the last row (the empty one) and enter the name of the workstation or deployment group to which you want to add members.
- You can do this by typing the name in the Workstation or Deployment Group field or by using the search button for those fields.
- When you use the search button for the Workstation field, the Machine Select form appears. When you use the search button for the Deployment Group field, the Deployment Group Search form appears.

Deploying Packages

This section provides an overview of the Deployment Director and discusses how to:

- Schedule a package for deployment
- Activate a scheduled package
- Install a scheduled package
- Deploy server packages

Understanding the Deployment Director

After you define and build a package, use the Deployment Director program (P9631) to schedule the package for deployment to individual workstations, deployment servers, or enterpriser servers. On the specified deployment date, users who are scheduled to receive the package can load the package when they sign on to PeopleSoft EnterpriseOne.

Alternatively, you can schedule the package to deployment groups or locations instead of specific machines. Deployment groups are useful in large enterprises that routinely deploy packages to many workstations and servers.

The Deployment Director program (P9631) simplifies and expedites the process of scheduling and deploying built packages to workstations and servers. The director displays a series of forms that enable you to specify the package that you want to deploy, the deployment destinations, and the deployment time.

After specifying the package that you want to deploy, you specify any of these destinations:

- Client workstation
- Enterprise server
- Deployment server or Deployment groups
- Locations

You can deploy a package either to specific workstations and servers, or you can schedule the deployment based on deployment groups or location. You cannot do both; you must select one of these methods.

You can make the package mandatory, which means that users cannot access the software until they have installed the package. If the package is optional, users will be given the option of installing the package every time that they sign on until they either install or decline the package.

In addition, you can specify a *push installation*, which means that the package can be deployed from the deployment server to the workstations that you specify, without requiring any interaction from the user.

The Deployment Director requires that PeopleSoft EnterpriseOne already be loaded on the workstation, unless you are using push installation. You can schedule a new full package to replace the existing package, or an update package to be merged with the existing package on the workstation.

The Deployment Director uses these tables:

- Machine Master (F9650)
- Machine Detail (F9651)
- Deployment Group Header (F9652)
- Deployment Group Detail Definitions (F9653)
- Deployment Locations Definition (F9654)
- Package Deployment Scheduling (F98825)
- Package Deployment on Servers Information (F98826)
- Software Package Header (F9603)

This table summarizes the function of each form in the Deployment Director:

Package Deployment Director form	View this form for a description of the Deployment Director.
Package Selection form	Use this form to find and select the package that you want to deploy.
Package Deployment Targets form	Use this form to specify the destination for your package. You can select individual client workstations, deployment servers, and enterprise servers, or you can deploy the package to a deployment group or location.
Package Deployment Attributes form	Use this form to enter the date and time that you want to deploy the package. Also specify whether the package is mandatory (that is, it must be installed by every package recipient) and whether you want to use Push Installation to deploy the package.
Deployment Client Workstation Selection form	Use this form to select each of the client workstations that will receive the package.
Deployment Server Selection form	Use this form to select each of the deployment servers that will receive the package.
Enterprise Server Selection form	Use this form to select each of the enterprise servers that will receive the package.
Deployment Location Selection form	Use this form to specify the deployment location that will receive the package.
Deployment Groups Selection form	Use this form to specify the deployment groups whose members will receive the package.

Build Selection form	For multitier deployment, use this form to specify the server or client package that you want to deploy to the destination deployment server.
Work with Package Deployment form	Use this form to review and revise the locations and package recipients that you entered through the Deployment Director.

Using the Deployment Director

After you have assembled and built your package, defined all machines, and verified your deployment groups, you are ready to use the Deployment Director to specify package recipients and schedule the package for deployment.

Throughout the deployment process, you can select to either proceed to the next form or return to the previous form. Also, regardless of where you are in the process, you can cancel it.

When you schedule a package for deployment to a machine rather than a deployment group or location, you can schedule to deploy the package to client workstations, deployment servers, enterprise servers, or a combination. The forms that appear vary depending on your selection. For example, if you indicate that you want to schedule a package for deployment to client workstations and a deployment server, the forms for selecting specific workstations and deployment servers appear. If you schedule a package for deployment only to client workstations, the server selection form does not appear.

When you access the Deployment Director, the Work with Package Deployment form enables you to view deployed package information by either machines, deployment groups, locations, or packages.

Depending on your display selection, the tree displays different information when you expand it. The following describes the information that appears as you expand the tree level by level:

- Machines

  Level One: Client Workstation, Deployment Server, and Enterprise Server headings

  Level Two: Specific machines under each of these three headings

  Level Three: Specific packages deployed to the machine, if any

- Deployment Groups

  Level One: Specific groups

  Level Two: Members of those groups

  Level Three: Specific packages deployed to the group member

- Locations

  Level One: Specific locations

  Level Two: Client Workstation, Deployment Server, Enterprise Server, and Remote Locations headings

  Level Three: Specific machines under the Client Workstation, Deployment Server, and Enterprise Server headings

  Level Three under Remote Locations only: Defined remote locations

  Level Four: Specific packages deployed to each machine, if any

  Level Four under Remote Locations only: Client Workstation, Deployment Server, and Enterprise Server headings

Level Five under Remote Locations only: Specific machines under the Client Workstation, Deployment Server, and Enterprise Server headings

Level Six under Remote Locations only: Specific packages deployed to each machine, if any

- Packages

  Level One: Package names

  Level Two: Client Workstation, Deployment Server, and Enterprise Server headings

  Level Three: Package deployment dates and times for each heading

  Level Four: Specific machines that have deployed that package for that date and time

Activating scheduled packages

After you successfully define a package deployment, you must activate the package so that it is available for installation using the Workstation Installation program. If you do not activate the package, it will not be included in the list of available packages when users launch the Workstation Installation program.

In some situations, you might need to control which packages are available for installation. If, for example, you have a package that is for the testing group only, you would want to make that package inactive so that it is not available for installation through the Workstation Installation program. Instead, you can use Deployment Director program (P9631) to schedule this package for deployment to the members of the testing group.

Installing a Scheduled Package

When users receive a package, they can select to install it when they sign on to PeopleSoft EnterpriseOne on or after the scheduled deployment date.

If the package is mandatory, users cannot access the system until they load the package.

If the package is optional, users can decline the package or postpone the installation until later. If they decide to postpone the installation, the software launches, and they will be given the opportunity to install the package the next time that they sign on.

If a package that is scheduled for push installation fails to load for some reason (such as if the power to the workstation was turned off during the time that the package was scheduled to deploy), that package will be included in the list of available packages when the user signs on.

See Also

Chapter 6, “Deploying Packages,” Scheduling the Push Installation Batch Application, page 150

Forms Used to Deploy Packages

Form Name	Form ID	Navigation	Usage
Work with Package Deployment	W9631J	From the Package and Deployment Tools menu (GH9083), select Package Deployment.	Select the package to deploy and review your selections.
Package Selection	W9631C	From the Package and Deployment Tools menu (GH9083), select Package Deployment. Click Add to launch the Deployment Director. Click Next	Select the package to deploy.

Page Name	Object Name	Navigation	Usage
Package Deployment Targets	W9631B	On Package Selection, click Next.	Select the types of machines on which to deploy the package.
Package Deployment Attributes	W9631D	On Package Deployment Targets, click Next.	Select the type of installation and the time.
Deployment Client Workstation Selection	W9631F	On Package Deployment Targets, select Client Workstation and click Next until the Deployment Client Workstation Selection form appears.	Select the workstations to which the package will be deployed.
Deployment Server Selection	W9631G	On Package Deployment Targets, select Deployment Server and click Next until the Deployment Server Selection form appears.	Select the Deployment Servers to which the package will be deployed.
Build Selection	W9631N	On Package Deployment Targets, click Next until the Build Selection form appears.	Select the server package build to deploy to the destination Deployment Server.
Enterprise Server Selection	W9631E	On Package Deployment Targets, select Enterprise Server and click Next until the Deployment Server Selection form appears.	Select the Enterprise Servers to which the package will be deployed.
Server Package Deployment Properties Revisions	W9631M	From the Package and Deployment Tools menu (GH9083), select Package Deployment. Select Machines and click Find to display information according to machine name. Find and select the deployed package for which you want to modify the options, and then select Properties from the Row menu.	Revise server package deployment options.

Scheduling a Package for Deployment

Access the Package Selection form.

To schedule a package for deployment to a client workstation or server:

1. On Package Selection, select the package that you want to deploy, and then click Next.
2. On Package Deployment Targets, click any of these options to indicate the type of machines to which you want to deploy your package, and then click Next:
   - Client Workstation
   - Deployment Server

- Enterprise Server

3. On Package Deployment Attributes, complete these fields:
   - Mandatory Installation
   - Enable Push Installation
   - Date/Time
4. If you want to deploy the package using push installation, which pushes the package to workstations from the deployment server, click the Enable Push Installation option, and then click Next.

   If you are deploying to workstations, the Deployment Client Workstation Selection form appears. If you are not deploying to workstations, skip the next step.
5. Find and select the workstations to which you want to deploy the package, and then click Next.

   Select a workstation by double-clicking in its row header. A check mark appears in the row header for each workstation that you select.

   If you are deploying to a deployment server, the Deployment Server Selection form appears. If you are not deploying to a deployment server, skip the next step.
6. Find the deployment server to which you want to deploy the package, and then click Next.

   Select a server by double-clicking in its row header. A check mark appears next to each server that you select.
7. On Build Selection, select the server package build that you want to deploy to the destination deployment server, and then click Close.
8. Click Next.

   If you are deploying to an enterprise server, the Enterprise Server Selection form appears. If you are not deploying to an enterprise server, skip the next step.
9. Find and select the enterprise server to which you want to deploy the package, and then click Next.

   Select a server by double-clicking in its row header.
10. On Work with Package Deployment, review your deployment selections.
11. To change any of your selections, click Prev to return to the appropriate previous form.
12. When you are finished reviewing and changing your deployment selections, click End.
13. If you are deploying a server package, find and select the server package on the Work with Package Deployment form, and then select Deploy from the Row menu.

After you schedule your package for deployment, at the specified time on the date that you specified, the package deploys to workstations. This package becomes available to the user when the user signs on.

If you are using Push Installation, the package automatically installs at the time that you specify in the Schedule Jobs program (P91300).

To schedule a package for deployment to a deployment group or location:

1. On Package Selection, select the package that you want to deploy, and then click Next.
2. On Package Deployment Targets, select either Deployment Group or Locations, and then click Next.
3. On Package Deployment Attributes, complete these fields:
   - Mandatory Installation
   - Enable Push Installation

- Date/Time

4. If you want to deploy the package using push installation, which pushes the package to workstations from the deployment server, click the Enable Push Installation option.
5. Click Next.

   If you are deploying to a deployment group, the Deployment Groups Selection form appears. If you are deploying to a location, skip the next step.
6. Find and select the deployment group that you want to receive the package, and then click Next.

   Select a group by double-clicking its row header.
7. If you are deploying to a location, the Deployment Location Selection form appears. Skip the next step if you are deploying to a deployment group.
8. Find and select the deployment location that you want to receive the package, and then click Next.

   To select a location, double-click the row header.
9. On Work with Package Deployment, review your deployment selections.
10. To change any of your selections, click Prev to return to the appropriate previous form.
11. When you are finished reviewing or changing your deployment selections, click End.
12. If you are deploying a server package, find and select the server package on the Work with Package Deployment form, and then select Deploy from the Row menu.

After you schedule your package for deployment, at the specified time on the date that you specified, the package deploys to workstations. This package becomes available to the user when the user signs on.

If you are using Push Installation, the package automatically installs at the time that you specify in the Schedule Jobs program (P91300).

See Also

See the following topics in the Package Management Guide:

Chapter 6, “Deploying Packages,” Scheduling the Push Installation Batch Application, page 150

Chapter 6, “Deploying Packages,” Scheduling a Package for Push Installation, page 149

Chapter 6, “Deploying Packages,” Understanding the Deployment Director, page 134

Revising Deployment Options

Access the Server Package Deployment Properties Revisions form.

Package Name

A package describes the location on the server where components you want to deploy to workstations or servers reside. There are two package types:

Full: Contains the full suite of applications (all specifications).

Update: Objects contained in this type of package are loaded after the workstation or server receives the package and the user signs on to the system. If the update package includes just-in-time applications, old versions of the application are deleted from the workstation and replaced by the current version the first time the user accesses that application. Update packages are always deployed on the date and time specified by the system administrator.

	With the exception of just-in-time applications included in an Update package, all packages are a snap shot at a point in time of the central objects for a particular path code. Just-in-time applications are dynamic, not built.
Path Code	The path code is a pointer to a set of objects, and is used to keep track of sets of objects and their locations.
Mandatory Installation	Indicates whether the package is mandatory or optional. Valid choices are: Y The deployment is mandatory. The user must install the package. N The deployment is optional to the user.
Enable Push Installation	A 1 in the field enables the package to be installed through Push Installation.
Date	Date to deploy updated objects to the listed machine.

Activating the Scheduled Package

Access the Work with Package Deployment form.

To activate a package:

1. Click Find.
2. Select from the list the packages that you want to activate or inactivate.

   Alternatively, you can enter the package name in the Package field.
3. Select Active/Inactive from the Row menu.

Installing a Scheduled Package

Sign on to PeopleSoft EnterpriseOne.

To install a scheduled package:

When you are scheduled to receive a package, the Just-In-Time Installation program launches and the Scheduled Packages form appears.

1. Perform one of these steps:
   - To load the package immediately, skip to the next step.
   - To decline the package permanently, select Decline from the Row menu.
   - To list all items in the package, select Package Detail from the Row menu.
   - To load the package at another time, click Close. If the package is mandatory, you will be unable to access the system until you load the package.
2. To load the package, select one or more packages you want to install and click Select.

   The Workstation Installation program loads the package. If you selected more than one package, the program installs them sequentially. When the installation is complete, the software launches.

Deploying Server Packages

This section provides an overview and discusses how to deploy server packages.

Understanding Server Package Deployment

The process for deploying a server package is nearly identical to deploying a package to a workstation. In both cases, you need to assemble, define, build, and schedule the package for deployment by using the Package Assembly (P9601), Package Build Director (P9621), and Deployment Director (P9631) programs.

After you schedule a server package for deployment, you must complete an additional step to launch the batch program that enables you to deploy to servers. You must perform this task whenever you deploy a package to an enterprise server or deployment server.

Important! Deploy server packages only when necessary, because the enterprise server is not available to process business applications and batch processes during the installation process. The enterprise server does not actually shut down during package installation. Instead, the system queues any jobs that are submitted to the enterprise server and runs them as soon as the installation finishes. For this reason, it is recommended that you schedule enterprise server packages to be deployed after hours to minimize impact on users. (Before you deploy a package to an enterprise server, verify that the services will be running during the deployment.)

To further minimize impact on the network and users, if your development environment is on the same enterprise server as your production environment, consider preventing developers from moving their own objects through server packages. Instead, require that an administrator perform this function.

To deploy a server package, select Deploy from the Row menu on Work with Package Deployment. This is the same function that you use to deploy packages to deployment servers during multitier deployment.

The system determines which of the batch programs to call, based on what is currently selected on the Work with Package Deployment form when you select Deploy from the Row menu:

- If a specific deployment server is selected, the system launches the Multi Tier Deployment batch program (R98825C).
- If the Deployment Server folder is selected, the system launches the Multi Tier Deployment batch program for every deployment server that has a package scheduled.
- If a specific enterprise server is selected, the system launches the Enterprise Server Deployment batch program (R98825D).
- If the Enterprise Server folder is selected, the system launches the Enterprise Server Deployment batch program for every enterprise server that has a package scheduled.
- If a specific package is selected, the system launches the Multi Tier Deployment batch program, and then the Enterprise Server Deployment batch program for the selected package.
- If you sort by packages and the Deployment folder is selected, the system launches both the Multi Tier Deployment batch program and Enterprise Server Deployment batch program for all packages.

If a specific workstation or the Workstations folder is selected, the Deploy option is unavailable.

When the system launches a batch program for all servers or all packages, deployment does not occur unless the package has been previously scheduled for a specific server.

Prerequisites

Before you complete the tasks in this section:

- Assemble the server package.
- Define the server package.
- Build the server package.
- Schedule the package for deployment to the appropriate server.

Forms Used to Deploy Server Packages

Form Name	Form ID	Navigation	Usage
Work with Package Deployment	W9631J	From the Package and Deployment Tools menu (GH9083), select Package Deployment.	Select the package to deploy and review your selections.

Deploying a server package

To deploy a server package:

1. On Work With Package Deployment, locate the server package that you want to deploy.

   Alternatively, select the enterprise server or, if the package is scheduled to deploy to more than one server, the Enterprise Server folder.
2. Select Deploy from the Row menu.
3. On Report Output Destination, select On Screen.
4. Click OK.

Using Push Installation

This section provides an overview of Push Installation and discusses how to:

- Preparing the Enterprise Server for push installation
- Preparing workstations for push installation
- Installing the listener
- Installing the listener using silent installation
- Stopping and uninstalling the listener
- Changing default parameter settings
- Scheduling a package for push installation
- Scheduling the push installation batch application
- Running the push package installation results report

Understanding Push Installation

Push installation is the only deployment method that provides automatic and unattended package deployment. This means that the system administrator can deploy a package (or several packages) to a workstation or group without requiring any action from workstation users.

For example, an administrator might schedule a package to deploy to a particular group after hours. When members of that group report to work the following morning, that package is available for immediate use.

Push installation is particularly useful in situations in which you need to quickly deploy packages with a minimum of intrusion or impact upon your normal production and development routines. By planning and scheduling package deployment judiciously, administrators can also minimize the impact upon network performance that can accompany large numbers of package deployments. Administrators can also use push installation to install the software on a workstation for the first time. This ability can greatly minimize downtime and provide maximum deployment flexibility.

During push installation, package contents are pushed from the deployment server to the workstation. In contrast to push installation, the Workstation Installation program pulls package contents from the deployment server to the workstation. Installations that are set up to use Deployment Director (P9631) for scheduled packages that are not push enabled also pull packages.

The end result of the deployment is the same, regardless of whether package contents are pushed or pulled. However, the advantage of a push installation is that no action is required from the workstation user other than to leave the workstation turned on during the time when the package is scheduled to deploy.

For an update package that contains program specifications, the term *package contents* refers to specifications. For a full package or an update package that does not contain application specifications, *package contents* refers to objects.

Push Installation process

This list summarizes the steps in the push installation process:

1. Install a push installation Listener program on each workstation that will receive pushed packages. This Listener monitors the progress of the Push Package Installation batch program (R98825) that runs on the server and performs functions such as monitoring installation status. The Listener can run as either a local service or a network service.
2. Schedule the package using the Deployment Director program (P9631). The Push Package Installation batch program reads the scheduling table and sends a message to the Listener on all workstations for which a package has been scheduled.
3. Use the Schedule Jobs program (P91300) to launch the batch program on the enterprise server. This program enables you to specify the job name, version, start date, start time, and recurrence.
4. At the specified start time, the Schedule Jobs program launches the Push Package Installation batch program (R98825), which initiates the package installation process from the deployment server. The Listener and the batch program interact during the process until installation is complete. Codes are passed from the Listener to the batch program to indicate the installation status (such as failed, successful, in progress, and so on).
5. When the installation finishes, the system sends an email message to the primary user of the workstation. This message indicates whether the installation was successful. Email notification works only if the package recipient is listed in the Machine Master table (F9650) and has an email address in the profile.
6. If the push installation fails for some reason (such as when the package recipient neglects to leave the workstation turned on), the installation status changes to Failed. If you want to reschedule the installation, you must first delete the row with the failed job, and then schedule the job again.

If the push installation is not successful, when the user signs on, the standard scheduling screen appears. At this point, the user can either accept the mandatory package or exit.

Installing the Listener

When you install the Listener on the workstations in your enterprise, you specify whether to run a local service or a network service. If you run the service locally on the workstation, the user must be signed on to receive a package that has been scheduled for push installation. If you run the service on the network, the Listener runs as a network account and the user does not have to be signed on to receive a package through push installation. The network service must have an administrator's user ID.

The disadvantage of running the service on the network is that it can be difficult to administer for all users on the enterprise. For example, because the parameters of the Listener apply to every user on the network, push installation users must install to and from the same locations. One user could have the software on drive C and another user have the same release on drive D. Also, every time users change their sign on passwords, the system administrator must update the Listener service with the new passwords in order for the service to work for those users. For these reasons, it is recommended that you install the Listener locally on each workstation.

Whether you run the Listener as a local service or network service, the workstation must be turned on to receive a scheduled package.

You can select from one of these ways to install the Listener on a workstation:

- Use a third-party software distribution system, such as the Tivoli Management Environment (TME10) Software Distribution System or the Microsoft System Management Server (SMS) software
- Distribute an executable installation program (a setup.exe file) and the accompanying ancillary files using an intranet Web site or the Worldwide Web
- Use Windows logon scripts (a .bat file) to call a C program
- Install from the Worldwide Web

Important! If the Listener is not installed and running on the workstation (or if the workstation is turned off), the push installation cannot occur. After you schedule a package, remind package recipients to leave their workstations on and the Listener service running, even during off-hours. If you set up the Listener to run as a local service, also remind users to remain signed on.

Installing the Listener Using Silent Installation

In some cases, it might be more convenient to install the Listener on workstations using silent installation. Typically, the administrator performs this task. Using this method, the administrator enters configuration settings for all workstations and distributes a batch file that automatically initiates the Listener installation the next time that the user signs on to the workstation.

The advantage of using silent installation to install the Listener is that the process is transparent to workstation users, and users are not required to enter configuration information or step through the installation process.

Forms Used to Push Packages

Form Name	Form ID	Navigation	Usage
Scheduling Information	W91300A	From the Job Scheduler menu (GH9015), select Schedule Jobs. Select the time zone that applies to your setup and click Select. Click Add to enter a new job.	Schedule the Push Installation Batch Application.
Push Package Installation Results	NA	From the Package and Deployment Tools menu (GH9083), select Push Package Installation Results.	Reports the status of a pushed package.

Preparing the Enterprise Server for Push Installation

To set up your server for push installation, you must first install and configure the Microsoft Domain Name Service (DNS) that is included with the Microsoft Windows Server. If you have not yet set up a domain name service, you can install Microsoft DNS by selecting the Network button in the Control Panel, then selecting the Services tab, and then adding Microsoft DNS Server.

After you add Microsoft DNS, you must configure the DNS by specifying your domain name and servers.

UNIX and iSeries Considerations

In an environment configured for Dynamic Host Configuration Protocol (DHCP), a server must run Windows Server to resolve workstation addresses because the Windows server dynamically assigns them.

To enable name resolution, you need to configure your servers to resolve their IP address lookup through a Windows DNS server, which, in turn, must be configured to review the WINS database when DHCP is enabled in the network domain.

Configuring your servers in this way ensures that the following process flow occurs during the push installation process:

1. From the host server on which the Push Installation batch program (R98825) runs, a business function attempts to retrieve the machine host address from the DNS server.
2. Because the DNS server does not contain IP addresses, it retrieves the address from the WINS server.
3. The WINS server returns the address to the DNS server.
4. The DNS server returns the address to the host server.
5. The host server finds the Listener on the client workstation and sends workstation installation information.
6. The workstation installation process starts.

Preparing Workstations for Push Installation

Before you can push an installation to a workstation, you must install a Listener on the workstation, which interacts with a business function that runs on the server. You must install this Listener on all workstations that you want to be enabled for push installation, regardless of whether you want to deploy packages to a machine on which PeopleSoft EnterpriseOne is already installed or a machine on which you are installing EnterpriseOne for the first time.

The Listener runs continuously on the Windows workstation as a service (and on a Windows 95 machine as a pseudo-service), and administrators can monitor it using the Task Manager. The Listener communicates with the batch application on the server, receiving and sending messages during the installation process. The Listener also monitors the progress of the installation and saves the installation completion code.

Installing the Listener

The following task describes how to install the Listener by launching an installation program (that is, a setup.exe program). You can distribute this program to users on your enterprise by using email to send them either the program or a shortcut, or by describing where the program is located on your server.

If you have a previous version of the Listener already installed, the installation program removes the previous version before copying the new version to your workstation.

Before you begin this task, close any applications that are currently open, and verify that the destination directory where you will be installing the Listener has sufficient disk space. You need approximately 2MB of free disk space to install all of the Listener files and components.

To install the Listener on workstations:

1. Launch the installation program by double-clicking setup.exe.
2. On the first Client Listener Setup form, click Next.
3. On the second Client Listener Setup form, complete these fields:

Release	Select the release that you want to install through push installation.
Path name	For the release that you selected, enter the full path name on the deployment server from which to initiate the installation.
Installation drive	Specify the drive on which you want to install the specified release.
Uninstall previous releases	Select the uninstall option to uninstall existing versions of the software before installing a new full package.
Start the Listener automatically on startup	Select the autostart option to automatically start the Listener service whenever your workstation starts up.

4. Click Next to proceed to the next installation form.
5. Select one of these options:
   - Local
   - Network

   Unless your system administrator tells you to install the Listener on the network, click Local to install the Listener on your local workstation.
6. In the Folder field, specify the destination drive and folder in which the Listener files will reside.
7. Click Finish to complete the installation.

After you have successfully installed the Listener on the workstation, a small ear icon appears on the Windows taskbar, indicating that the Listener has been loaded. By right-clicking this icon, you can start or stop the Listener, or change the default parameter settings.

See Also

Chapter 6, “Deploying Packages,” Changing Default Parameter Settings, page 149

Installing the Listener Using Silent Installation

To install the Listener using silent installation:

1. Edit these settings in the listen_silent_setup.inf file that is included on the software CD:

   ServiceType Enter Local or Network, depending on where you want to run the Listener service.

   WorkstationDirPath Enter the location on the workstation where you want to install the Listener program and related files. For example, C:\Program Files\PeopleSoft Client Listener.

   Release Enter the base release. For example, B9. Do not enter a cumulative update release, such as E811.

   InstallPath Enter the location on the workstation where the software is installed. For example, D:\E811.

   LaunchPath Enter the deployment server name and the location from which the Client Workstation Installation program runs. For example, `\\`*server name*`\b9\PeopleSoft Client Install\setup.exe`.

   AutoStart Enter 1 to automatically start the Listener service when the workstation starts up. Enter 0 if you do not want to enable Autostart.

   UninstallPackage Enter 1 if you want to automatically uninstall previous versions before installing a new full package. Enter 0 if you do not want to enable automatic uninstall.

2. Create or modify a batch file to include the silent installation parameter `/s` for the ListenSetup.exe program. The batch file must reside in the same location as the ListenSetup.exe program.

   For example, your batch file might contain this line:

   ```
   start \\servername\E811\client\misc\ListenSetup.exe /s listen_silent_setup.inf
   ```

3. Distribute the inf file and the batch file to workstation users. You can distribute these files or place them on a network server where workstation users can copy the files to their workstation.
4. Instruct users to restart their workstations to run the batch file and load the Listener using silent installation.

After workstation users have successfully installed the Listener, the Listener icon appears on the Windows taskbar. Users can click this icon to start and stop the Listener or change Listener settings.

Important! You cannot change the name of the Listener silent installation file that is shipped with the software. The file name must be listen_silent_setup.inf.

Stopping and Uninstalling the Listener

You can stop the Listener if you are certain that you do not want to use push installation to install packages. If you change your mind later, you can restart the Listener.

To stop the Listener:

The easiest way to stop the Listener is to right-click the Listener icon on the Windows task bar and select Stop Listener.

Alternatively, you can stop the Listener using these steps:

1. Open the Control Panel.
2. Select Services.
3. Select PeopleSoft Client Listener.
4. Click Stop.

To uninstall the Listener:

1. Open the Control Panel.
2. Select Add/Remove Programs.
3. Select PeopleSoft Client Listener.
4. Click Remove All Components.

Changing Default Parameter Settings

You can use the icon for the Listener that appears on the Windows taskbar to change the default parameter settings that you entered when you initially installed the Listener. You can specify:

- The release and client installation path for that release
- The default installation directory
- Whether to uninstall the previous package before installing a new full package

To change default parameter settings:

1. Right-click the Listener icon in the Windows taskbar.
2. Select Change Default Parameters.
3. Complete these fields:

Release	Select the release that you want to install through push installation.
Path name	For the release that you selected, enter the full path name on the deployment server from which to initiate the installation. If necessary, click the button to the right of the field to browse for available paths.
Default installation directory	Specify the drive on which you want to install the specified release.
Uninstall previous package before installation of Full Package	Indicate whether you want to uninstall previous packages before you install a full package. When you select this option, the system completely uninstalls the existing package before it installs a new full package.
Automatically activate this service on start-up	Select this option if you want the Listener service to automatically start when the user signs on to the workstation.

4. Click OK when you are finished making changes.

Scheduling a Package for Push Installation

After you have installed the Listener on your workstations, you can schedule a package for deployment.

The process for scheduling a package for push installation is identical to the process for scheduling a package using the Deployment Director program (P9631). When you schedule the package using this program, select the Enable Push Installation option on the Package Deployment Attributes form. If you do not select this option, the package will be deployed through normal scheduled deployment.

When you use the Schedule Jobs program (P91300), you can set recurrence, which determines how frequently the job runs until it finishes successfully. If you do not set recurrence, the job runs only one time. In the case of Push Installation, recurrence determines the interval of time between installation attempts. After the package is successfully deployed, the job ceases to run.

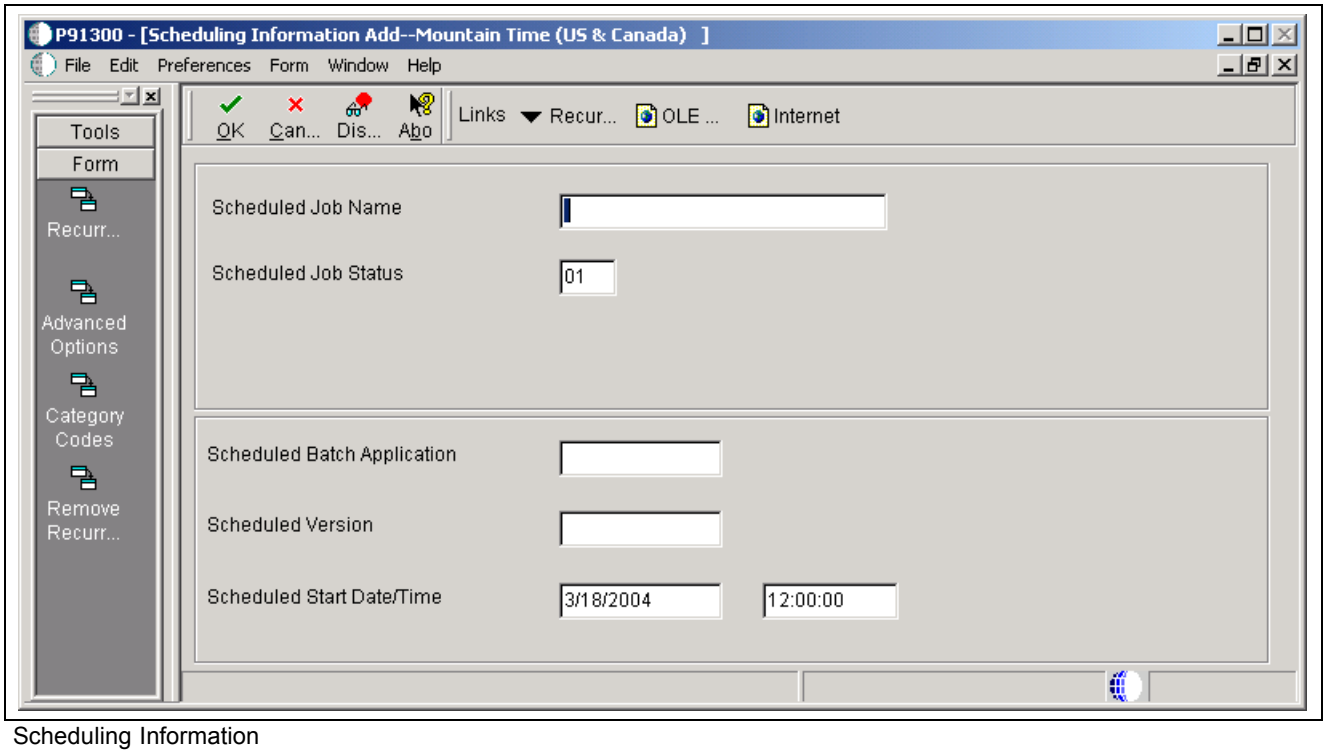
As with scheduling any other package for deployment, all machine names (that is, package recipients) must be defined in the Machine Master (F9650) table. This table is populated when users sign on. Alternatively, you can enter machine names manually using the Deployment Locations Application program (P9654A).

Scheduling the Push Installation Batch Application

After you have installed the Listener on all affected workstations and have scheduled the package through the Deployment Director program (P9631), you must use the Schedule Jobs program (P91300) to run the Push Package Installation batch program (R98825) on the server.

Before you begin to schedule the Push Installation batch application, complete these steps:

- Remind package recipients to leave their workstations turned on, even after hours.
- Remind users who are using a local service that they must be signed on.
- Remind package recipients to verify that the Listener is running.



Scheduling Information

To schedule the Push Installation batch application:

1. On Scheduling Information, complete these fields:

   Scheduled Job Name A name that uniquely identifies to the system and the user a scheduled job.

	Use this name to indicate the job function, such as Monthly Close or Nightly Back Up.
Scheduled Job Status	The current status of the scheduled job. As long as the status is active, the Scheduler determines if the job should be submitted to the server for execution. When the scheduled end date for the job has been reached, the status changes to Not Active. To stop the Scheduler from considering the job for submission, you can change the status to Not Active (or suspended) at any time prior to the end date. You can reactivate the job if you want the scheduler to include the job again, but you can reactivate a job only if the end date is in the future.
Scheduled Batch Application	The object name of the report that the Scheduler submits to the server.
Scheduled Version	The version of the report scheduled to run. A version identifies a specific set of data selections and sequencing settings the batch job uses.
Scheduled Start Date/Time	The next date on which the Scheduler submits the scheduled job to the server for execution.

2. To set the job recurrence (that is, to specify how frequently the job runs) select Recurrence from the Form menu.

   If you do not specify a recurrence by completing the fields on this form, the job runs only one time. It is recommended that for the Push Installation batch program, you set recurrence to run every 30 minutes.
3. On Recurring Scheduling Information Revisions, click OK.
4. On Scheduling Information, to enter any overrides, resubmissions, or expiration options, select Advanced Options from the Form menu.
5. Click the tab that corresponds to the information that you want to enter or revise:
   - Launch Overrides
   - Job Expiration
   - Job Resubmission
   - Batch Application Overrides
6. Revise the information and click OK.

After scheduling the job, you can use the Object Configuration Manager (P986110) to verify that the server on which the Push Installation batch program is running points to the same Package Deployment Scheduling (F98825) and Machine Master (F9650) tables that the Deployment Director program uses.

See Also

EnterpriseOne Tools 8.94 PeopleBook: System Administration, "Using the Scheduler Application," Working with the Job Scheduler

Running the Push Package Installation Results Report

Access the Push Package Installation Results form.

This report provides the same information that you get when you run the Push Package Installation batch program (R98825).

The report includes this information:

- Machine key
- Package name and path code
- User class or group
- Package status and status description
- Install status
- Package installation description
- Mandatory install (yes or no)

Push Installation Status Codes

The following table lists the status codes and descriptions that the Push Package Installation program (R98825) uses. Codes marked with an asterisk indicate conditions in which the Push Package Installation program continues to attempt the installation the next time that the Push Package Installation program runs.

Status Code	Description
200*	Scheduled
210*	In Progress
220	Successful Install
230	Install Failed
240*	Install Running
250*	PeopleSoft EnterpriseOne Running
260*	Listener Not Started/Installed
270	General Error
280	Already Installed
290	Invalid Package
300	Install Attempted
310	Machine Down

Installing Workstations from CD

This section provides an overview of installing workstations from CD and discusses how to:

- Defining the CD Writer location
- Deploying a package to the CD Writer location
- Creating the installation CD

Understanding how to install workstations from CD

If your system includes a CD writer, you can build and deploy packages to the CD writer location. After copying the package to a CD, you can then use the CD as a portable deployment tier from which to perform workstation installations. That is, you can run from the CD the setup.exe program that launches the Workstation Installation program.

You can set up your enterprise so that you can deploy packages to the CD writer and install the software from a CD.

The first step in the process of configuring your system for deployment from CD is to define the CD writer location, if it is not already defined. In this step, you essentially create a pseudo deployment server from which you will later copy package data onto the CD by using the software for your CD writer.

When you define the CD writer location in the Machine Identification application, you must also add the correct path codes to the Environments exit.

The process for defining this location is identical to the process for defining any other new deployment server.

Prerequisite

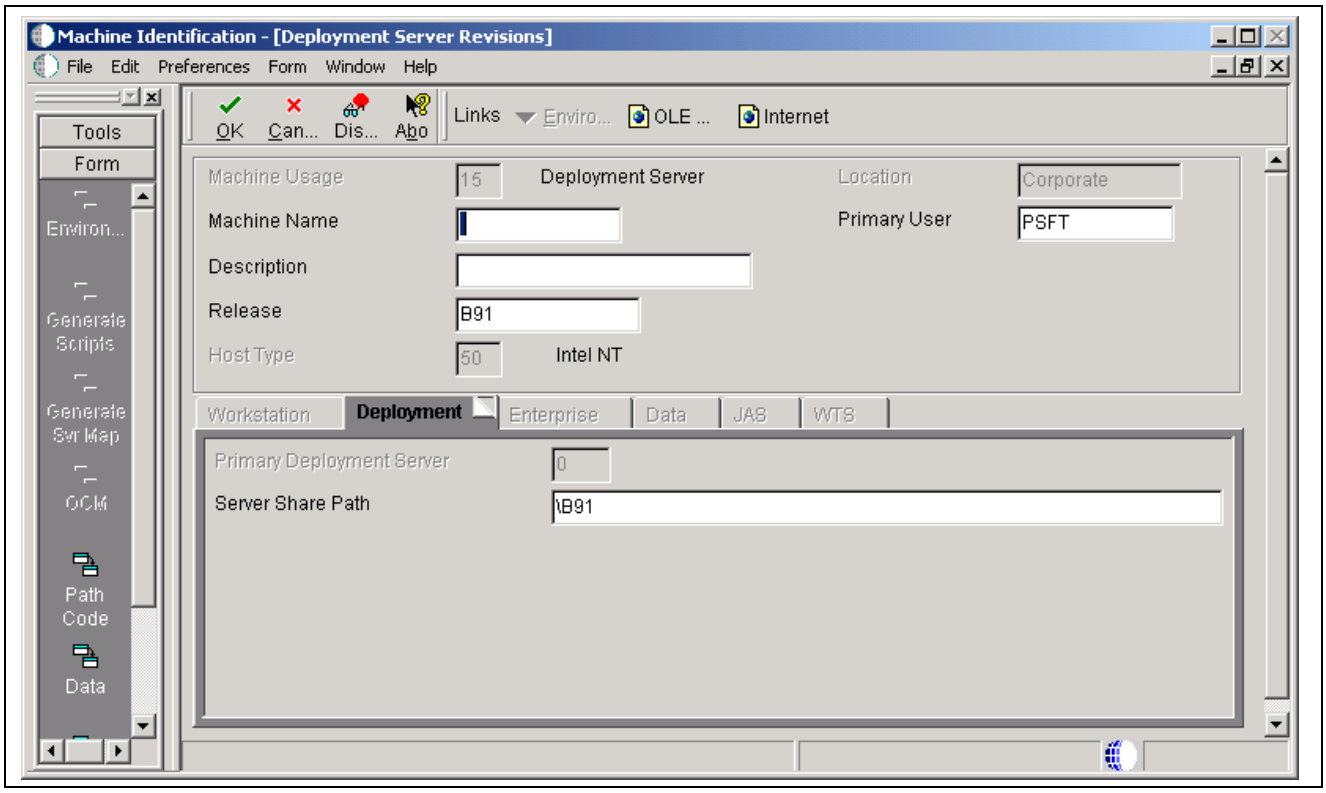
Assemble, define, and build the package that you want to write to the CD.

Forms Used to Install Workstations from a CD

Form Name	Form ID	Navigation	Usage
Deployment Server Revisions	W9654AC	From the Package and Deployment Tools menu (GH9083), select Machine Identification. Subordinate to the appropriate location, select Deployment server. Click Add to add a new machine.	
Work with Package Deployment	W9631J	From the Package and Deployment Tools menu (GH9083), select Package Deployment.	Select the package to deploy and review your selections.

Defining the CD Writer Location

Access the Deployment Server Revisions form.



Deployment Server Revisions

To define the CD writer location:

1. On Deployment Server Revisions, complete these fields:

   Machine Name The Location or Machine Key indicates the name of the machine on the network (server or workstation).

   Release The release number as defined in the Release Master.

   Primary User Primary User for the listed Machine.

   Server Share Path The shared directory for this path code. The objects that are stored on a file server will be found in this path.

2. If you want to specify a location for data, a foundation, or help files, do so by choosing Data, Foundation, or Helps from the Form menu.

   If you do not specify a location for data, foundation, or helps, the system uses the default locations.

3. Click OK.
4. Click Close to exit from the Work with Locations and Machines form.
5. In Windows Explorer, locate the folder named Client Install.
6. Copy this folder by dragging the folder to the CD writer location.

   The location is the server share path that you entered on the Deployment Server Revisions form.

Deploying a Package to the CD Writer Location

After you define the CD writer as a deployment server, you are ready to deploy a package to the CD writer location that you specified. This task involves these two procedures:

- Deploy to the CD writer location the package that you want to write to the CD.
- Modify the Install.inf and Package.inf files in preparation for writing the package contents to the CD.

Access the Work with Package Deployment form.

To deploy the package to the CD writer location:

1. On Work With Package Deployment, click Add.
2. Complete the forms in the Deployment Director in the same way as you would for any other package.
3. From the Work with Package Deployment form, find and select the package that you just scheduled for deployment, and then select Deploy from the Row menu to deploy the package.

After you deploy the package to the CD writer location, the directory structure for that location will look similar to this example:

```
Multitier\package_inf\Appl_B.inf
Multitier\systemcomp\system.cab
Multitier\datacomp\data.cab
Multitier\helpscomp\helps.cab
Multitier\Appl_pgf\Package\Appl_B
Multitier\package_inf\Appl_B.inf
```

In the previous example, Multitier is the name of the server share path. Appl_B is the package name.

Note. The server share path name is not included when you copy folders to the CD. In the previous example, the items copied to the CD are \package_inf\Appl_B.inf, \systemcomp\system.cab, and so on.

To modify the Install.inf and Package.inf files:

1. In Windows Explorer, find the CD writer location and open the folder that contains the package that you deployed.

   This folder has the name that you entered in the Server Share Path field on the Deployment Server Revisions form when you defined the CD writer location. In the previous example, the server share path name is Multitier.

2. Open the folder Client Installation, and then open the file Install.inf.

   That is, double-click the icon for the file to launch the Microsoft Notepad application.

3. In the section [FileLocations], modify the line so that two periods and a backslash (\) precede the package_inf entry. The line should look like this example after you modify it:

   ```
   [FileLocations]
   PackageInfs=..\package_inf
   ```

4. Similarly, open the Package_inf folder and open the *package name*.inf file.

   In the previous example, this file is named Appl_b.inf.

5. In the section [SrcDirs], modify each of the lines so that two periods and a backslash (\) precede each entry.

   After modification, the [SrcDirs] section should look similar to this example:

   ```
   [SrcDirs]
   SAPPL_PGF=..\APPL_PGF\package\APPL_B
   SSYS=..\systemcomp
   SAPPL_PGFDATA=..\datacomp
   ```

```
SHELP=..\helpscomp
```

Creating the Installation CD

After you deploy the package to the CD writer location and modify the Install.inf and Package.inf files, you are ready to copy the package contents to the CD. Use the software that came with your CD writer to accomplish this process, which typically involves copying the package contents to the CD. Refer to the documentation that came with your CD writer for more information about this process.

You copy the package to the CD by copying the subdirectories that are subordinate to the server share path directory. The server share path directory is not created on the CD. (In the previous example, the server share path directory is called Multitier, and it is the same name that you entered in the Server Share Path field on the Deployment Server Revisions form.

When you are finished copying the directories to the CD, the CD should contain these directories:

- Appl_pgf (contains package information)
- datacomp (contains the database cabinet file)
- helpscomp (contains the helps cabinet file)
- systemcomp (contains the foundation cabinet file)
- package_inf (contains the package.inf file)
- Client Install (contains the Workstation Installation program)

Note. Actual names might not be the same as those listed because each system might be different.

Deploying Data

This section discusses how to:

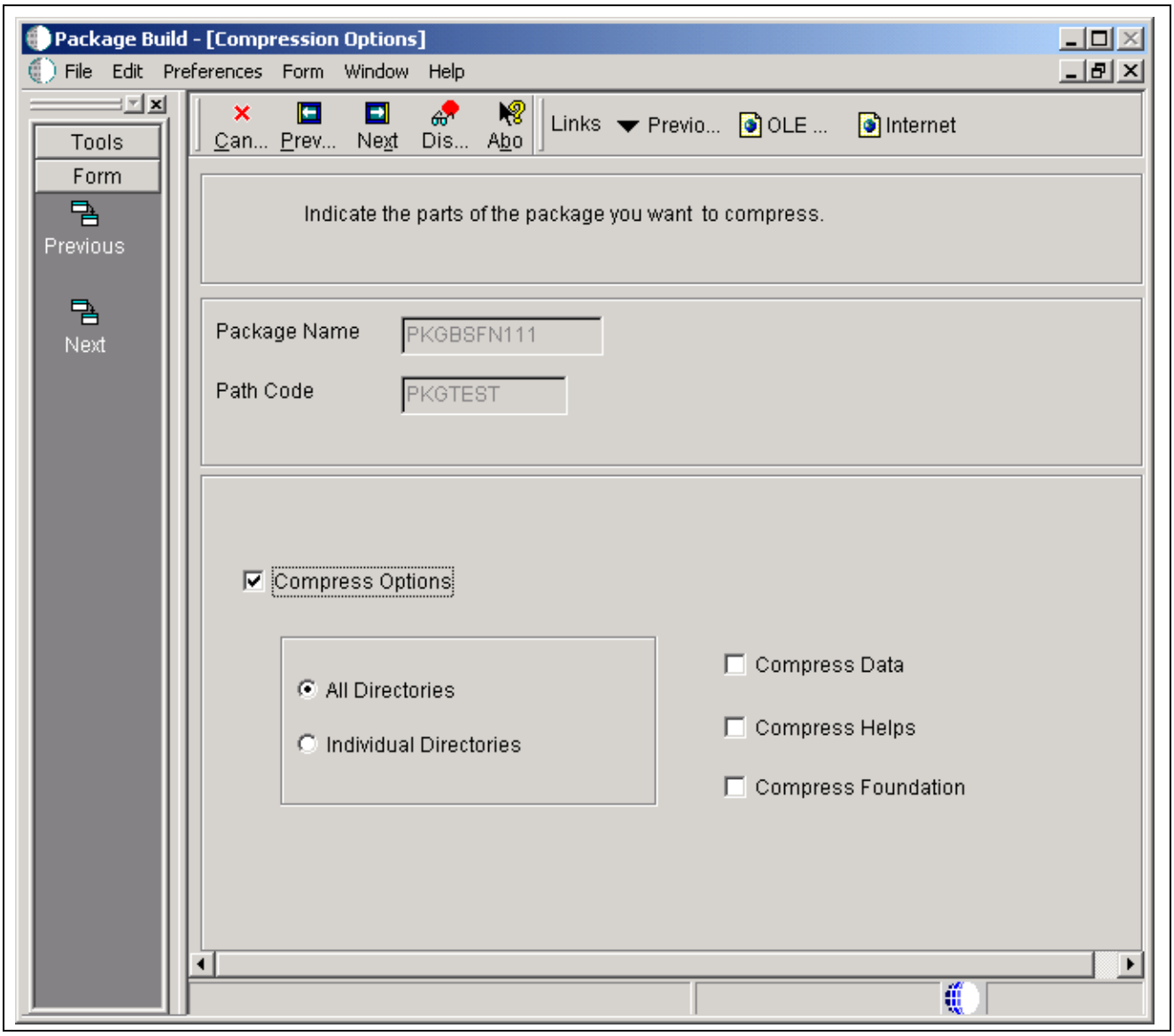
- Deploy a new replicated local database
- Deploy a special database for store-and-forward users

Forms Used to Deploy Data

Form Name	Form ID	Navigation	Usage
Compression Options	W9621M	From the Package and Deployment Tools menu (GH9083), select Package Assembly. Select the package and select Define Build from the Row menu. Proceed through the Package Assembly Director forms until you reach the Compression Options form.	

Deploy a new replicated local database

Access the Compression Options form.



Compression Options

To deploy a new replicated local database:

1. On Compression Options, select only the Compress Options and Compress Data options.

   These options recompress the database so that anyone who installs this package will receive current data.

2. Proceed through the remaining screens to complete the package build process.

Note. To verify that compression was successful, review the size and date of the .cab file that appears in the PeopleSoft/E811/*pathcode*/package/data folder. If the file is 1KB or smaller, the compression failed.

Deploying a Special Database for Store-and-Forward Users

Store-and-forward users require Supported Local Databases that are much larger than the databases required for other users. Therefore, you should build an update package with one line item for the database for store-and-forward users.

Because store-and-forward users cannot connect to the network to install their applications using just-in-time installation, they must install full packages. Therefore, store-and-forward users should use an environment that has just-in-time installation turned off. Instruct your store-and-forward users to install the normal full package, and then install the update package that was created for store-and-forward users.

Access the Compression Options form.

To deploy a special database for store-and-forward users:

1. On Compression Options, select the Compress Data option.
2. Schedule the package to deploy to all workstations that will be performing store-and-forward transactions.

CHAPTER 7

Setting Up Multitier Deployment

This chapter provides an overview of multitier deployment and discusses how to:

- Define deployment servers
- Distribute software to deployment locations
- Deploy server packages in a multitier network

Multitier Deployment

PeopleSoft EnterpriseOne software is normally distributed to workstations from a centralized location. In many cases, you can minimize the performance impact on a single deployment server by using systematic scheduling for software installations. For example, if your site has more than 50 workstations that require a package installation, but you release software only four times a year, you can effectively schedule the installations over a weekend, at night, or during off-peak hours.

While this distribution method is the simplest approach for software deployment, network capacity constrains configurations that have either multiple remote sites or large numbers of users at a single site. For example, software installations to workstations that are connected to the centralized deployment location by a 56KB circuit can take 4 to 6 hours to run.

Multitier deployment provides sites the flexibility of installing packages on workstations and servers from more than one deployment location and more than one deployment server. These additional deployment locations and servers are called deployment tiers. Specifically, instead of installing multiple workstations across a WAN circuit, multitier deployment enables you to transfer a compressed package from the centralized location to the remote workgroup server that acts as a second deployment tier. Hence, multitier deployment means deploying from more than one deployment tier.

For example, you might have one deployment server at your main location, and a second deployment server that serves a remote location. Because the server at the remote location is responsible for deploying to workstations and servers at that location, you do not need to deploy packages from the main deployment server across a WAN, as you would in a single-tier deployment configuration.

Workgroup servers can also be used as second tier deployment locations in a LAN environment where large numbers of workstations need to install software concurrently. It is recommended that you implement multitier deployment if your site has more than 50 workstations receiving installations per day.

The primary function of multitier deployment is to reduce network traffic (and the delays that result from heavy traffic) by enabling enterprises with more than one geographic location to deploy from a secondary deployment server at the remote site. Instead of installing packages across the WAN from the deployment server to workstations at a remote location, you can copy the package and the package.inf file from the deployment server at the primary location to the deployment server at the remote location. The server at the remote location can then deploy the package to the workstations and servers at that location.

Consider implementing a multitier deployment configuration when either of the following applies to you:

- Too many workstations are installing packages from the same location, causing the server and network performance to suffer.
- The workstations are remotely connected to the deployment server over a WAN, which requires unacceptably long installation times.

Normally, you decide whether to implement multitier deployment during Configurable Network Computing (CNC) implementation. Although you can enable this function at any time, you typically set it up after you have already installed the software and are preparing your production sites to go live.

To set up multitier deployment, you must define the machines (and their associated path codes) that are used for deployment. In addition, you can use a scheduler function to define when software should be pushed to the tiered deployment location.

You must also define individual user characteristics for multitier deployment. Normally, you do this by modifying the user profiles to indicate the deployment location from which a user pulls a package.

Multitier Deployment Terminology

The following table lists and describes multitier deployment terms:

Primary Deployment Location (tier 1)	The primary or base deployment location is the location in which the package to be deployed to secondary or remote locations is built. The system requires at least one deployment server for installing and maintaining the software. The server at the primary deployment location should be dedicated solely to deploying and operating PeopleSoft EnterpriseOne and should not be used for any other purposes in your company.
Tier Deployment Location (tier 2)	Tier deployment locations (also known as remote or secondary locations) have one or more deployment servers that enable you to install the software on the workstations at that location. These servers receive their packages from the deployment server at the primary or base location. Machines at the tier deployment locations cannot use Object Librarian functions, such as object check-in and check-out. These machines are designed for deployment use only and are not designed to be used for remote development. The number of tiered locations that you can have depends on the network and server capacity.
Tier Workstations	Tier workstations are workstations that connect to a tier deployment location for software installation. The number of workstations per deployment location depends on the network and machine load. All tiered workstations must also have a Windows domain connection that enables them to connect to, read, and copy files from the shared drives of their respective deployment locations.

See Also

EnterpriseOne PeopleTools 8.11 Installation Guide

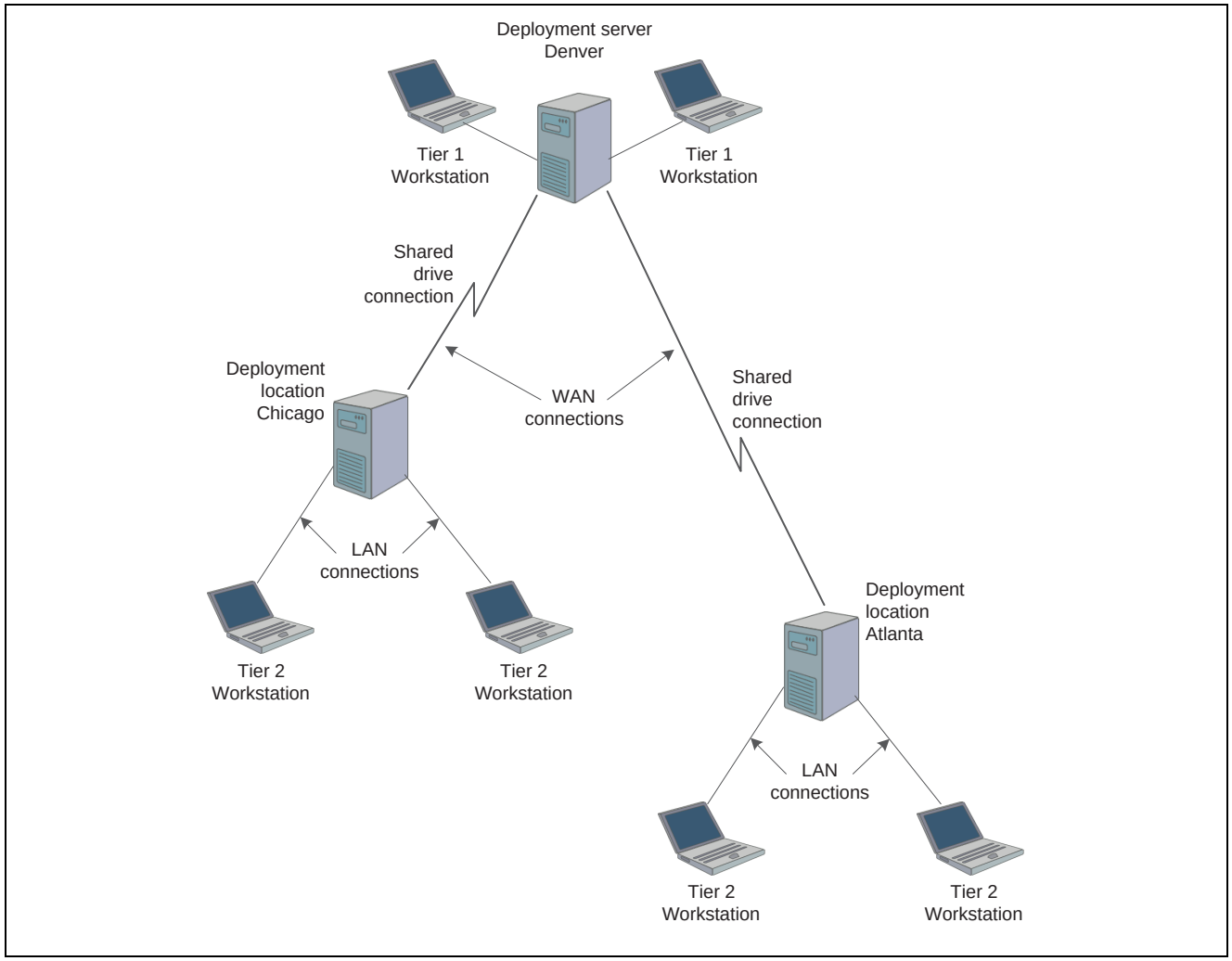
Multitier Deployment Features

Multitier deployment offers these features:

- You can deploy workstations from any number of deployment servers.
- You can easily add deployment locations, and the deployment machine does not need to be a server; it could be a Windows workstation.
- Setup and administration are straightforward tasks.
- You maintain central control over files and data that is transferred to remote deployment locations.
- You can easily schedule software for deployment to remote sites.
- Multitier deployment is integrated into the Deployment Director, so the process for deploying is essentially the same as the process for deploying in a single-tiered setup.

Example: Two-Tier Deployment Strategy

This diagram illustrates a typical two-tier deployment strategy:



Two-tier deployment strategy

Multitier Implementation

Packages are always built on the deployment server at the primary location. After you build the package that you want to deploy to remote locations, you must follow these steps to implement multitier deployment.

1. Define deployment locations

   You must define each physical location to which you want to deploy. For example, if your main office is in Denver and you want to deploy to your branch office in Seattle, you must define the Seattle deployment location.

2. Create Deployment Server Definitions

   You must also define the deployment server at each remote location.

   Note. This step is not necessary if you used the Remote Location Workbench to create deployment server definitions when you installed the software.

3. Schedule the Package

   The next step in the process is to schedule the package for deployment using the Deployment Director program (P9631). The process of scheduling a package for multitier deployment is identical to the process for scheduling any other package.

4. Deploy the Package to Workstations

   After you deploy the package to the deployment server at the remote location, that server can deploy to the workstations at that location.

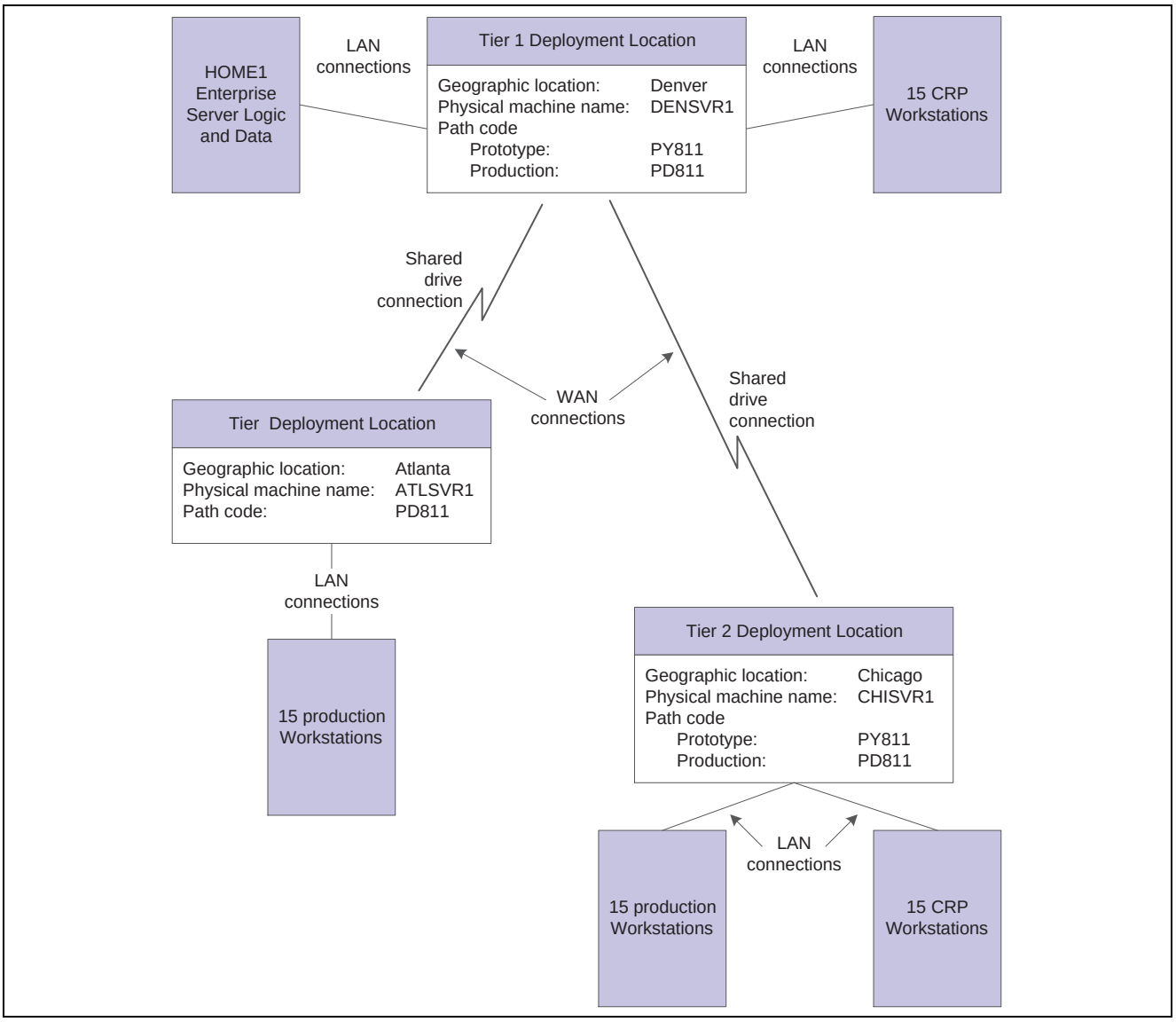
A Multitier Deployment Case Study

This case study is for a two-tier deployment environment. As this case study shows, deploying packages to workstations using WAN connections is generally not efficient. Instead, you should deploy from a primary deployment server to tier deployment locations. After that, you can install packages to LAN-attached workstations from each local deployment location.

For package installations, a remote deployment location functions as a file server. You cannot build packages at a remote deployment location; packages must be built at the primary deployment location.

While locally-attached workstations can pull packages from the tier deployment location, these workstations still require enterprise server and database server connectivity.

The diagram presents an illustration of this case study.



Multi-tier deployment strategy

The following table describes the assumptions used by the Tier 1 location in this case study.

Denver (home office, Tier 1 deployment location)	
Characteristic	Setting
Enterprise server name	HOME1
Deployment server name	Denver: DENSVR1 Atlanta: ATLSVR1 Chicago: CHISVR1

Denver (home office, Tier 1 deployment location)	
Prototype workstations	Denver: 15 Atlanta: 0 Chicago: 15
Production workstations	Denver: 0 Atlanta: 15 Chicago: 15
PeopleSoft EnterpriseOne release	B9
Deployment tier	Denver: 1 Atlanta: 2 Chicago: 2
Path codes	Denver: PD811, PY811 Atlanta: PD811 Chicago: PD811, PY811

To configure your system for multitier deployment

These steps summarize the steps necessary to configure your system for multitier deployment.

1. Define the deployment locations.

   Define the deployment locations on the deployment server (DENSVR1 in this example). Use the Deployment Locations Application program (P9654A) to define all deployment locations.

   For this case study, complete these fields to define three locations, one for each deployment location: Denver, Atlanta, and Chicago.

   Location Enter the name of the deployment location.

   In this case study, you assign these names for each physical deployment location:

   Denver

   Atlanta

   Chicago

   Description Enter a description (any value up to 30 characters) for each deployment location.

   For example:

   Denver: Denver - Tier 1

   Atlanta: Atlanta - Tier 2

   Chicago: Chicago - Tier 2

Location Code Enter the current location for deployment.

For example, DEN.

Parent Location Enter the name of the parent location for the location that you are adding.

For example, Corporate.

2. Create deployment server definitions.

   Use the Deployment Locations Application program (P9654A) to create a definition for each deployment server at the deployment locations that you created. For this case study, you need to define a deployment server for Atlanta and Chicago. The deployment server in Denver is already defined because it is the primary (tier 1) server.

   For this case study, complete the fields on the Deployment Server Revisions form as described in the following table:

Field	Value
Machine Name	Enter the name of the physical machine. In this case study, enter these values: Denver: DENSVR1 Atlanta: ATLSVR1 Chicago: CHISVR1
Description	Enter a description (any value up to 30 characters). For example, Multitier Deployment - Denver.
Release	Enter the release. For example, E811.
Primary User	Enter the primary user for the machine that you entered.
Server Share Path	Enter the name of the shared directory for the path code in which system files and other files reside. For example, \E811.

3. Schedule the package.

   Schedule the package to be deployed from the tier 1 deployment server to the tier 2 deployment servers through the Deployment Director program (P9631) or the Multi Tier Deployment batch program (R98825C).

See Also

Chapter 6, “Deploying Packages,” Defining Deployment Parameters, page 127

Defining Deployment Servers

This section provides an overview of defining deployment servers and discusses how to:

- Define a new deployment server
- Revise an existing deployment server

Understanding how to define a deployment server

The Deployment Locations Application program (P9654A) enables you to either add a new deployment server definition or modify an existing definition. When you add a new deployment server definition, the system creates a record in the Deployment Locations Definition table (F9654) for each deployment location. Each server at each deployment location must be defined.

Typically, the table contains one record for the primary deployment server and one record per deployment location for each release. If you have multiple releases, you must create multiple records for the servers at each deployment location.

If you used the Remote Location Workbench to create deployment server definitions when you installed, you do not need to define deployment servers again.

In many situations, you might need to modify the definition for a deployment server that you already defined. For example, you would need to change the definition if the server share path or release changes, or if you want to designate a different server as your primary deployment server. The process for revising an existing deployment server definition is very similar to the process for adding a new definition.

See Also

Running the Installation Workbench in the *EnterpriseOne PeopleTools 8.11 Installation Guide*

Prerequisites

Before you complete the tasks in this section:

- Ensure that you have a thorough understanding of the CNC concepts that enables you to define and implement packages, workstation installations, path codes, central objects, replicated objects, and user profiles.
- Ensure that adequate disk space exists on the disk drive for the machine that you will be using as a tier deployment location. Also, remember that each full package that you deploy adds approximately 1.4GB. The amount of required disk space varies, depending on the amount of data that you replicate to a Supported Local Database.
- If you are adding a new definition for a deployment server at a remote location, you first need to define that location.

Note. For information about defining locations, see Defining Locations in the Package Management Guide.

It is recommended that you do not define deployment locations with a DEV path code because the DEV path code is normally associated with developers who are not located at remote locations.

Forms Used to Define a Deployment Server

Form Name	Form ID	Navigation	Usage
Deployment Server Revisions	NA	To define a new deployment server: From the Package and Deployment Tools menu (GH9083), select Machine Identification. Find the location where the deployment server resides, and expand the tree, if necessary, to display the different machine types. Select the Deployment Server heading and then click Add. To revise an existing deployment server: From the Package and Deployment Tools menu (GH9083), select Machine Identification. Find the deployment server for which you want to modify the definition, and click Select.	Define a new deployment server or revise an existing deployment server for multitier deployment.

Defining a New Deployment Server

Access the Deployment Server Revisions form.

To add a new deployment server definition:

1. On the Deployment Server Revisions form, complete these fields:

   Machine Name The Location or Machine Key indicates the name of the machine on the network (server or workstation).

   Release The release number as defined in the Release Master.

   Primary User Primary User for the listed Machine.

   Server Share Path The shared directory for this path code. The objects that are stored on a file server will be found in this path.

   Because you are entering a definition for a server other than your primary deployment server, the field Primary Deployment Server is not available. You can enter this field only if you first delete the definition for your current primary deployment server.

2. Select Path Code from the Form menu.
3. On the Machine Path Code Revisions form, enter the path code for the primary or base location from which the secondary location will receive the package. When you enter the path code, the server share path will default to the base server share path for that machine.

Note. You must specify the path code for each deployment server at all secondary locations. If you do not indicate the path code, multitier deployment may not work.

4. If your package includes a user-defined foundation, data, or help files, specify those items by choosing Foundation, Data, or Helps from the Form menu.
5. On the Deployment Server Revisions form, click OK.

Revising an Existing Deployment Server Definition

Access the Deployment Server Revisions form.

To revise an existing deployment server definition:

1. On Deployment Server Revisions, you can modify these fields:
   - Description
   - Release
   - Primary User
   - Server Share Path
   - Primary Deployment Server

     If you are modifying the primary deployment server, you can also change the primary deployment server. For any other server, you cannot change the value in this field. A 1 in this field indicates that the deployment server is the primary deployment server. You can have only one primary deployment server.
2. Select Path Code from the Form menu.
3. On the Machine Path Code Revisions form, enter the path code for the primary or base location from which the secondary location receives the package. When you enter the path code, the system displays the default server share path, which is the base server share path for that machine.

   Note. You must specify the path code for each deployment server at all secondary locations. If you do not indicate the path code, multitier deployment might not work.
4. If your package includes a user-defined foundation, data, or help files, specify those items by choosing Foundation, Data, or Helps from the Form menu.
5. On the Deployment Server Revisions form, click OK.

See Also

Chapter 6, “Deploying Packages,” Defining Deployment Parameters, page 127

Distributing Software to Deployment Locations

This section provides an overview of the multitier distribution process and discusses how to:

- Distributing software through package deployment
- Scheduling packages for multitier deployment

- Distributing software through the multitier deployment batch process
- Copying Workstation installation programs to deployment locations

Understanding how to Distribute Software to Deployment Locations

You use the Deployment Director program (P9631) or the Multi Tier Deployment batch program (R98825C) to distribute software to deployment locations and schedule them for deployment. Use the Deployment Director program to define the scheduling parameters or to deploy the package immediately. Otherwise, use a version of the Multi Tier Deployment batch application to distribute the software to deployment locations.

Whether you push or pull the software depends on the machine on which you run the deployment function or batch program for the scheduling program. You push the software if you run the program or report on the primary deployment server or from a workstation. Conversely, you pull the software if you run the program from a workstation at the tier deployment location. In either case, execute the application on the primary deployment server machine for push installation, or the destination deployment location for pull installation.

Important! If you push the software, you must have full read and write privileges for both deployment servers (that is, the primary deployment server and the tier deployment location or destination machine), even if you do not run the batch application. If you do not have read and write authority on both servers, the deployment will fail.

After the package software is distributed through the Deployment Director program or the Multi Tier Deployment batch program, you must manually copy the workstation installation programs from the primary deployment server to the tier deployment location. These programs are located in the client portion of the base installation directory. Workstations that install from tier deployment locations access the tier deployment server for the help files unless you select to deploy the help files to the workstation.

Forms Used to Distribute Software to Deployment Locations

Form Name	Form ID	Navigation	Usage
Work with Batch Versions - Available Versions form	W98305A	From the System Administration Tools menu (GH9011), select Batch Versions.	Distribute software that has been previously defined and scheduled through the Deployment Director program (P9631). Running the batch program deploys the software if the scheduled date and time have already passed.

Distributing Software through Package Deployment

Use this method when you want to distribute software immediately after you define a deployment schedule. If you select this option, the software will be distributed immediately, regardless of the timing parameters that you specify in the scheduling fields.

Access the Work With Package Deployment form.

To distribute software through Deployment Director:

From the Package and Deployment Tools menu (GH9083), select Package Deployment.

1. Click the Machines option, and then click Find.
2. Select the deployment server to which you want to deploy.

   If you have scheduled to deploy packages to more than one deployment server, you can select the Deployment Server folder to deploy to all applicable servers, rather than deploying to each individual server.
3. Select the deployment server name to deploy all packages that are listed under the server name.

   Alternatively, select only the package that you want to deploy.
4. From the Row menu, select Deploy.

   This option launches the Multi Tier Deployment batch program (R98825C), which enables you to deploy to deployment servers.

   You can also use this Deploy option to launch a batch application that deploys enterprise server packages. Therefore, if you select an enterprise server name or the Enterprise Server folder on the Work With Package Deployment form, the Enterprise Server Deployment batch program (R98825D) launches, not the Multi Tier Deployment batch program. When you select a workstation, the Deploy option is not available.

Scheduling Packages for Multitier Deployment

After you create the server package that you want to deploy using multitier deployment, you can use the Deployment Director program (P9631) to schedule that package to a server at the same location or a remote location. Verify that the server package is compressed because, unlike client packages, you cannot deploy the server package using multitier deployment unless it is compressed.

Also, before you can deploy a server package with server multitier deployment, the package status must be either 50 (Build Completed Successfully) or 70 (Build Completed with Errors). If the package does not have a status of 50 or 70, it will be unavailable when you schedule the deployment.

The process for scheduling packages to a secondary deployment server is identical to the process for scheduling a package to a primary server. Be sure to specify the secondary server location when you schedule the package.

Note. When you create a distribution schedule, remember that you cannot schedule multiple packages to deploy at the same time.

Distributing Software Through the Multitier Deployment Batch Process

Access the Work with Batch Versions - Available Versions form.

To distribute software through the Multi Tier Deployment batch application:

1. On Work With Batch Versions - Available Versions, enter *R98825C* in the Batch Application field and click Find:
2. Select the Multi Tier Deployment version that you want to use.
3. On Version Prompting, click Submit.
4. On Report Output Destination, select one of these options and click OK:
   - On Screen
   - Printer

These steps ensure that the system runs the Client Workstation Installation program locally at the deployment location. If you do not complete these two steps, the system searches the base location for the Client Workstation Installation program and its associated files, and copies the files across the WAN to the deployment location.

To copy workstation installation programs to deployment locations:

1. Connect to each deployment location and use Windows Explorer to drag this client directory to the tier 2 workstation:

   ```
   \\Tier1DeploymentServerName\E811\PeopleSoft Client Install
   ```

2. Open the package.inf file and locate the [FileLocations] section. Change the PackageInfs line to reflect the machine name of the deployment server and the environment at the deployment location. For example:

   ```
   PackageInfs=\\machine name\environment\ENVIRON\Evapps\appl_pgf\package_inf
   ```

3. Save your changes by choosing Save from the File menu.
4. Select Exit from the File menu.

Installing Workstation Packages from Deployment Locations

In the case of multitier deployment, this program resides on the tier deployment location in the client subdirectory that is subordinate to the base installation directory. The workstation that is attached to the deployment location must have read access to this directory on the deployment location.

See Also

Installing EnterpriseOne on the Workstations in the *EnterpriseOne PeopleTools 8.11 Installation Guide*

Deploying Server Packages in a Multitier Network

This section provides an overview of the server package deployment process and discusses how to schedule a server package for multitier deployment.

Understanding Multitier Deployment of Server Packages

Server multitier deployment lets you automatically deploy a server package from one deployment server to another. The target deployment server to which you are deploying the package can be either in the same location as the source deployment server that sends the package, or in one or more remote locations.

You typically build packages on the primary deployment server at the base location, but using a different server for installations from the base location might have some advantages. In some situations, you might prefer to deploy a server package to a server at a remote location, rather than require remote users to access the server package over a WAN.

Server package multitier deployment enables users to select which package builds they want to deploy. For example, if you are building a Prod package for multiple server platforms, you can select the build for the platform that has been successfully built. The benefit of having the ability to select builds is that users are no longer required to wait to install a new client package.

You use the Deployment Director program (P98631) for server multitier deployment. When you deploy a server package, the system copies these components:

- Foundation, data, and helps
- Subdirectories that are subordinate to the package name directory, including:
  - bin32
  - include
  - lib32
  - obj
  - source
  - spec
- Subdirectories that are subordinate to the server build directories, including:
  - bin32
  - spec
  - obj
  - lib32
- The ServerPackage.inf file in the server build directories, including:
  - The package.inf file
  - The pack directory that is subordinate to the package name directory

When server builds are deployed only without client builds, none of the subdirectories that are subordinate to the package name directory are copied, except for the pack subdirectory.

The following table describes where major package components are copied from and to during the deployment process. The server share path is derived from the Machine Detail table (F9651).

Package Component	Copied From	Copied To
Package	Copied from the path indicated in the SrcDirs section of the package.inf file.	Copied to<*server share path*>\<*path code*>\<*package name*>on the destination deployment server.
Foundation and data	Copied from the path indicated in the SrcDirs section of the package.inf file.	Copied to<*server share path*>*systemcomp*on the destination deployment server, if the default location is selected.
Package.inf file	If the source deployment server is the primary deployment server in the base location, this file is copied from the path indicated in the Object Path Master File table (F00942). Otherwise, this file is copied from the path to the package.inf file in the Machine Detail table (F9651).	Copied to<*server share path*>*package_inf*on the destination deployment server.

Smart Deployment

Server multitier deployment incorporates the smart deployment feature, which makes available only the server packages that match the available servers for the package destination. For example, if server packages exist at the base location for the HP9000 and the iSeries, but the destination location has only an HP9000, only the HP9000 package is made available for deployment to that location. In other words, you cannot deploy a server package unless the package destination supports that server platform.

Even if you have multiple locations, smart deployment ensures that only the server packages that match the platform of the destination are available.

Automatic Package.inf File Updating

When you deploy a server package to a remote location, these sections of the package.inf file are updated:

- SrcDirs
- Attributes
- DeploymentServerName
- Location

The package.inf file is not copied when you deploy to a deployment server in the base location.

Prerequisite

Assemble and build the server package as described in Package Build.

To schedule a server package for multitier deployment:

1. Select the server package that you want to deploy.
2. On Package Deployment Targets, click the Deployment Server option.
3. On Package Deployment Attributes, enter the deployment date and time, and select any deployment options.
4. On Deployment Server Selection, enter the machine name to which you want to deploy the server package.

   Select the machine by double-clicking the row header.
5. On Build Selection, select the build (or version) of the package that you want to deploy by double-clicking the row header.

   Because of the Smart Deployment feature, the system enables you to select only the package builds that match the configuration at the destination location. For example, if package builds exist for an iSeries and an HP 9000, but the destination location has only an HP9000, you cannot select the iSeries build.
6. Click Close to exit the Build Selection form.
7. On Work With Package Deployment, click End to finish the server package scheduling process.

See Also

Chapter 7, “Setting Up Multitier Deployment,” Distributing Software to Deployment Locations, page 168

Glossary of PeopleSoft Terms

absence entitlement This element defines rules for granting paid time off for valid absences, such as sick time, vacation, and maternity leave. An absence entitlement element defines the entitlement amount, frequency, and entitlement period.

absence take This element defines the conditions that must be met before a payee is entitled to take paid time off.

academic career In PeopleSoft Enterprise Campus Solutions, all course work that a student undertakes at an academic institution and that is grouped in a single student record. For example, a university that has an undergraduate school, a graduate school, and various professional schools might define several academic careers—an undergraduate career, a graduate career, and separate careers for each professional school (law school, medical school, dental school, and so on).

academic institution In PeopleSoft Enterprise Campus Solutions, an entity (such as a university or college) that is independent of other similar entities and that has its own set of rules and business processes.

academic organization In PeopleSoft Enterprise Campus Solutions, an entity that is part of the administrative structure within an academic institution. At the lowest level, an academic organization might be an academic department. At the highest level, an academic organization can represent a division.

academic plan In PeopleSoft Enterprise Campus Solutions, an area of study—such as a major, minor, or specialization—that exists within an academic program or academic career.

academic program In PeopleSoft Enterprise Campus Solutions, the entity to which a student applies and is admitted and from which the student graduates.

accounting class In PeopleSoft Enterprise Performance Management, the accounting class defines how a resource is treated for generally accepted accounting practices. The Inventory class indicates whether a resource becomes part of a balance sheet account, such as inventory or fixed assets, while the Non-inventory class indicates that the resource is treated as an expense of the period during which it occurs.

accounting date The accounting date indicates when a transaction is recognized, as opposed to the date the transaction actually occurred. The accounting date and transaction date can be the same. The accounting date determines the period in the general ledger to which the transaction is to be posted. You can only select an accounting date that falls within an open period in the ledger to which you are posting. The accounting date for an item is normally the invoice date.

accounting split The accounting split method indicates how expenses are allocated or divided among one or more sets of accounting ChartFields.

accumulator You use an accumulator to store cumulative values of defined items as they are processed. You can accumulate a single value over time or multiple values over time. For example, an accumulator could consist of all voluntary deductions, or all company deductions, enabling you to accumulate amounts. It allows total flexibility for time periods and values accumulated.

action reason The reason an employee's job or employment information is updated. The action reason is entered in two parts: a personnel action, such as a promotion, termination, or change from one pay group to another—and a reason for that action. Action reasons are used by PeopleSoft Human Resources, PeopleSoft Benefits Administration,

PeopleSoft Stock Administration, and the COBRA Administration feature of the Base Benefits business process.

action template In PeopleSoft Receivables, outlines a set of escalating actions that the system or user performs based on the period of time that a customer or item has been in an action plan for a specific condition.

activity In PeopleSoft Enterprise Learning Management, an instance of a catalog item (sometimes called a class) that is available for enrollment. The activity defines such things as the costs that are associated with the offering, enrollment limits and deadlines, and waitlisting capacities.

In PeopleSoft Enterprise Performance Management, the work of an organization and the aggregation of actions that are used for activity-based costing.

In PeopleSoft Project Costing, the unit of work that provides a further breakdown of projects—usually into specific tasks.

In PeopleSoft Workflow, a specific transaction that you might need to perform in a business process. Because it consists of the steps that are used to perform a transaction, it is also known as a step map.

address usage In PeopleSoft Enterprise Campus Solutions, a grouping of address types defining the order in which the address types are used. For example, you might define an address usage code to process addresses in the following order: billing address, dormitory address, home address, and then work address.

adjustment calendar In PeopleSoft Enterprise Campus Solutions, the adjustment calendar controls how a particular charge is adjusted on a student's account when the student drops classes or withdraws from a term. The charge adjustment is based on how much time has elapsed from a predetermined date, and it is determined as a percentage of the original charge amount.

administrative function In PeopleSoft Enterprise Campus Solutions, a particular functional area that processes checklists, communication, and comments. The administrative function identifies which variable data is added to a person's checklist or communication record when a specific checklist code, communication category, or comment is assigned to the student. This key data enables you to trace that checklist, communication, or comment back to a specific processing event in a functional area.

admit type In PeopleSoft Enterprise Campus Solutions, a designation used to distinguish first-year applications from transfer applications.

agreement In PeopleSoft eSettlements, provides a way to group and specify processing options, such as payment terms, pay from a bank, and notifications by a buyer and supplier location combination.

allocation rule In PeopleSoft Enterprise Incentive Management, an expression within compensation plans that enables the system to assign transactions to nodes and participants. During transaction allocation, the allocation engine traverses the compensation structure from the current node to the root node, checking each node for plans that contain allocation rules.

alternate account A feature in PeopleSoft General Ledger that enables you to create a statutory chart of accounts and enter statutory account transactions at the detail transaction level, as required for recording and reporting by some national governments.

analysis database In PeopleSoft Enterprise Campus Solutions, database tables that store large amounts of student information that may not appear in standard report formats. The analysis database tables contain keys for all objects in a report that an application program can use to reference other student-record objects that are not contained in the printed report. For instance, the analysis database contains data on courses that are considered for satisfying a requirement but that are rejected. It also contains information on

	courses captured by global limits. An analysis database is used in PeopleSoft Enterprise Academic Advisement.
AR specialist	Abbreviation for *receivables specialist.* In PeopleSoft Receivables, an individual in who tracks and resolves deductions and disputed items.
arbitration plan	In PeopleSoft Enterprise Pricer, defines how price rules are to be applied to the base price when the transaction is priced.
assessment rule	In PeopleSoft Receivables, a user-defined rule that the system uses to evaluate the condition of a customer's account or of individual items to determine whether to generate a follow-up action.
asset class	An asset group used for reporting purposes. It can be used in conjunction with the asset category to refine asset classification.
attribute/value pair	In PeopleSoft Directory Interface, relates the data that makes up an entry in the directory information tree.
audience	In PeopleSoft Enterprise Campus Solutions, a segment of the database that relates to an initiative, or a membership organization that is based on constituent attributes rather than a dues-paying structure. Examples of audiences include the Class of '65 and Undergraduate Arts & Sciences.
authentication server	A server that is set up to verify users of the system.
base time period	In PeopleSoft Business Planning, the lowest level time period in a calendar.
benchmark job	In PeopleSoft Workforce Analytics, a benchmark job is a job code for which there is corresponding salary survey data from published, third-party sources.
billing career	In PeopleSoft Enterprise Campus Solutions, the one career under which other careers are grouped for billing purposes if a student is active simultaneously in multiple careers.
bio bit or bio brief	In PeopleSoft Enterprise Campus Solutions, a report that summarizes information stored in the system about a particular constituent. You can generate standard or specialized reports.
book	In PeopleSoft Asset Management, used for storing financial and tax information, such as costs, depreciation attributes, and retirement information on assets.
branch	A tree node that rolls up to nodes above it in the hierarchy, as defined in PeopleSoft Tree Manager.
budgetary account only	An account used by the system only and not by users; this type of account does not accept transactions. You can only budget with this account. Formerly called "system-maintained account."
budget check	In commitment control, the processing of source transactions against control budget ledgers, to see if they pass, fail, or pass with a warning.
budget control	In commitment control, budget control ensures that commitments and expenditures don't exceed budgets. It enables you to track transactions against corresponding budgets and terminate a document's cycle if the defined budget conditions are not met. For example, you can prevent a purchase order from being dispatched to a vendor if there are insufficient funds in the related budget to support it.
budget period	The interval of time (such as 12 months or 4 quarters) into which a period is divided for budgetary and reporting purposes. The ChartField allows maximum flexibility to define operational accounting time periods without restriction to only one calendar.

business event In PeopleSoft Receivables, defines the processing characteristics for the Receivable Update process for a draft activity.

In PeopleSoft Sales Incentive Management, an original business transaction or activity that may justify the creation of a PeopleSoft Enterprise Incentive Management event (a sale, for example).

business unit A corporation or a subset of a corporation that is independent with regard to one or more operational or accounting functions.

buyer In PeopleSoft eSettlements, an organization (or business unit, as opposed to an individual) that transacts with suppliers (vendors) within the system. A buyer creates payments for purchases that are made in the system.

campus In PeopleSoft Enterprise Campus Solutions, an entity that is usually associated with a distinct physical administrative unit, that belongs to a single academic institution, that uses a unique course catalog, and that produces a common transcript for students within the same academic career.

catalog item In PeopleSoft Enterprise Learning Management, a specific topic that a learner can study and have tracked. For example, "Introduction to Microsoft Word." A catalog item contains general information about the topic and includes a course code, description, categorization, keywords, and delivery methods. A catalog item can have one or more learning activities.

catalog map In PeopleSoft Catalog Management, translates values from the catalog source data to the format of the company's catalog.

catalog partner In PeopleSoft Catalog Management, shares responsibility with the enterprise catalog manager for maintaining catalog content.

categorization Associates partner offerings with catalog offerings and groups them into enterprise catalog categories.

category In PeopleSoft Enterprise Campus Solutions, a broad grouping to which specific comments or communications (contexts) are assigned. Category codes are also linked to 3C access groups so that you can assign data-entry or view-only privileges across functions.

channel In PeopleSoft MultiChannel Framework, email, chat, voice (computer telephone integration [CTI]), or a generic event.

ChartField A field that stores a chart of accounts, resources, and so on, depending on the PeopleSoft application. ChartField values represent individual account numbers, department codes, and so forth.

ChartField balancing You can require specific ChartFields to match up (balance) on the debit and the credit side of a transaction.

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

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

checkbook In PeopleSoft Promotions Management, enables you to view financial data (such as planned, incurred, and actual amounts) that is related to funds and trade promotions.

checklist code In PeopleSoft Enterprise Campus Solutions, a code that represents a list of planned or completed action items that can be assigned to a staff member, volunteer, or unit. Checklists enable you to view all action assignments on one page.

class In PeopleSoft Enterprise Campus Solutions, a specific offering of a course component within an academic term.

See also *course.*

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

clearance In PeopleSoft Enterprise Campus Solutions, the period of time during which a constituent in PeopleSoft Contributor Relations is approved for involvement in an initiative or an action. Clearances are used to prevent development officers from making multiple requests to a constituent during the same time period.

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

cohort In PeopleSoft Enterprise Campus Solutions, the highest level of the three-level classification structure that you define for enrollment management. You can define a cohort level, link it to other levels, and set enrollment target numbers for it.

See also *population*and *division*.

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

collection rule In PeopleSoft Receivables, a user-defined rule that defines actions to take for a customer based on both the amount and the number of days past due for outstanding balances.

comm key See *communication key.*

communication key In PeopleSoft Enterprise Campus Solutions, a single code for entering a combination of communication category, communication context, communication method, communication direction, and standard letter code. Communication keys (also called *comm keys* or *speed keys*) can be created for background processes as well as for specific users.

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

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

condition In PeopleSoft Receivables, occurs when there is a change of status for a customer's account, such as reaching a credit limit or exceeding a user-defined balance due.

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

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

constituents In PeopleSoft Enterprise Campus Solutions, friends, alumni, organizations, foundations, or other entities affiliated with the institution, and about which the institution maintains information. The constituent types delivered with PeopleSoft Enterprise Contributor Relations Solutions are based on those defined by the Council for the Advancement and Support of Education (CASE).

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

context In PeopleCode, determines which buffer fields can be contextually referenced and which is the current row of data on each scroll level when a PeopleCode program is running.

In PeopleSoft Enterprise Campus Solutions, a specific instance of a comment or communication. One or more contexts are assigned to a category, which you link to 3C access groups so that you can assign data-entry or view-only privileges across functions.

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

control table Stores information that controls the processing of an application. This type of processing might be consistent throughout an organization, or it might be used only by portions of the organization for more limited sharing of data.

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

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

course In PeopleSoft Enterprise Campus Solutions, a course that is offered by a school and that is typically described in a course catalog. A course has a standard syllabus and credit level; however, these may be modified at the class level. Courses can contain multiple components such as lecture, discussion, and lab.

See also *class*.

course share set In PeopleSoft Enterprise Campus Solutions, a tag that defines a set of requirement groups that can share courses. Course share sets are used in PeopleSoft Enterprise Academic Advisement.

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

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

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

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

dataset A data grouping that enables role-based filtering and distribution of data. You can limit the range and quantity of data that is displayed for a user by associating dataset rules with user roles. The result of dataset rules is a set of data that is appropriate for the user's roles.

delivery method In PeopleSoft Enterprise Learning Management, identifies the primary type of delivery method in which a particular learning activity is offered. Also provides

	default values for the learning activity, such as cost and language. This is primarily used to help learners search the catalog for the type of delivery from which they learn best. Because PeopleSoft Enterprise Learning Management is a blended learning system, it does not enforce the delivery method. In PeopleSoft Supply Chain Management, identifies the method by which goods are shipped to their destinations (such as truck, air, rail, and so on). The delivery method is specified when creating shipment schedules.
delivery method type	In PeopleSoft Enterprise Learning Management, identifies how learning activities can be delivered—for example, through online learning, classroom instruction, seminars, books, and so forth—in an organization. The type determines whether the delivery method includes scheduled components.
directory information tree	In PeopleSoft Directory Interface, the representation of a directory's hierarchical structure.
division	In PeopleSoft Enterprise Campus Solutions, the lowest level of the three-level classification structure that you define in PeopleSoft Enterprise Recruiting and Admissions for enrollment management. You can define a division level, link it to other levels, and set enrollment target numbers for it. See also *population* and *cohort*.
document sequencing	A flexible method that sequentially numbers the financial transactions (for example, bills, purchase orders, invoices, and payments) in the system for statutory reporting and for tracking commercial transaction activity.
dynamic detail tree	A tree that takes its detail values—dynamic details—directly from a table in the database, rather than from a range of values that are entered by the user.
edit table	A table in the database that has its own record definition, such as the Department table. As fields are entered into a PeopleSoft application, they can be validated against an edit table to ensure data integrity throughout the system.
effective date	A method of dating information in PeopleSoft applications. You can predate information to add historical data to your system, or postdate information in order to enter it before it actually goes into effect. By using effective dates, you don't delete values; you enter a new value with a current effective date.
EIM ledger	Abbreviation for *Enterprise Incentive Management ledger.* In PeopleSoft Enterprise Incentive Management, an object to handle incremental result gathering within the scope of a participant. The ledger captures a result set with all of the appropriate traces to the data origin and to the processing steps of which it is a result.
elimination set	In PeopleSoft General Ledger, a related group of intercompany accounts that is processed during consolidations.
entry event	In PeopleSoft General Ledger, Receivables, Payables, Purchasing, and Billing, a business process that generates multiple debits and credits resulting from single transactions to produce standard, supplemental accounting entries.
equitization	In PeopleSoft General Ledger, a business process that enables parent companies to calculate the net income of subsidiaries on a monthly basis and adjust that amount to increase the investment amount and equity income amount before performing consolidations.
equity item limit	In PeopleSoft Enterprise Campus Solutions, the amounts of funds set by the institution to be awarded with discretionary or gift funds. The limit could be reduced by amounts equal to such things as expected family contribution (EFC) or parent contribution. Students are packaged by Equity Item Type Groups and Related Equity Item Types. This limit can be used to assure that similar student populations are packaged equally.

event A predefined point either in the Component Processor flow or in the program flow. As each point is encountered, the event activates each component, triggering any PeopleCode program that is associated with that component and that event. Examples of events are FieldChange, SavePreChange, and RowDelete.

In PeopleSoft Human Resources, also refers to an incident that affects benefits eligibility.

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

exception In PeopleSoft Receivables, an item that either is a deduction or is in dispute.

exclusive pricing In PeopleSoft Order Management, a type of arbitration plan that is associated with a price rule. Exclusive pricing is used to price sales order transactions.

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

financial aid term In PeopleSoft Enterprise Campus Solutions, a combination of a period of time that the school determines as an instructional accounting period and an academic career. It is created and defined during the setup process. Only terms eligible for financial aid are set up for each financial aid career.

forecast item A logical entity with a unique set of descriptive demand and forecast data that is used as the basis to forecast demand. You create forecast items for a wide range of uses, but they ultimately represent things that you buy, sell, or use in your organization and for which you require a predictable usage.

fund In PeopleSoft Promotions Management, a budget that can be used to fund promotional activity. There are four funding methods: top down, fixed accrual, rolling accrual, and zero-based accrual.

gap In PeopleSoft Enterprise Campus Solutions, an artificial figure that sets aside an amount of unmet financial aid need that is not funded with Title IV funds. A gap can be used to prevent fully funding any student to conserve funds, or it can be used to preserve unmet financial aid need so that institutional funds can be awarded.

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

gift table In PeopleSoft Enterprise Campus Solutions, a table or so-called *donor pyramid* describing the number and size of gifts that you expect will be needed to successfully complete the campaign in PeopleSoft Contributor Relations. The gift table enables you to estimate the number of donors and prospects that you need at each gift level to reach the campaign goal.

GL business unit Abbreviation for *general ledger business unit.* A unit in an organization that is an independent entity for accounting purposes. It maintains its own set of accounting books.

See also *business unit*.

GL entry template Abbreviation for *general ledger entry template.* In PeopleSoft Enterprise Campus Solutions, a template that defines how a particular item is sent to the general ledger. An item-type maps to the general ledger, and the GL entry template can involve multiple general ledger accounts. The entry to the general ledger is further controlled

by high-level flags that control the summarization and the type of accounting—that is, accrual or cash.

GL Interface process Abbreviation for *General Ledger Interface process.* In PeopleSoft Enterprise Campus Solutions, a process that is used to send transactions from PeopleSoft Enterprise Student Financials to the general ledger. Item types are mapped to specific general ledger accounts, enabling transactions to move to the general ledger when the GL Interface process is run.

group In PeopleSoft Billing and Receivables, a posting entity that comprises one or more transactions (items, deposits, payments, transfers, matches, or write-offs).

In PeopleSoft Human Resources Management and Supply Chain Management, any set of records that are associated under a single name or variable to run calculations in PeopleSoft business processes. In PeopleSoft Time and Labor, for example, employees are placed in groups for time reporting purposes.

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

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

incur In PeopleSoft Promotions Management, to become liable for a promotional payment. In other words, you owe that amount to a customer for promotional activities.

initiative In PeopleSoft Enterprise Campus Solutions, the basis from which all advancement plans are executed. It is an organized effort targeting a specific constituency, and it can occur over a specified period of time with specific purposes and goals. An initiative can be a campaign, an event, an organized volunteer effort, a membership drive, or any other type of effort defined by the institution. Initiatives can be multipart, and they can be related to other initiatives. This enables you to track individual parts of an initiative, as well as entire initiatives.

inquiry access In PeopleSoft Enterprise Campus Solutions, a type of security access that permits the user only to view data.

See also *update access.*

institution In PeopleSoft Enterprise Campus Solutions, an entity (such as a university or college) that is independent of other similar entities and that has its own set of rules and business processes.

item In PeopleSoft Inventory, a tangible commodity that is stored in a business unit (shipped from a warehouse).

In PeopleSoft Demand Planning, Inventory Policy Planning, and Supply Planning, a noninventory item that is designated as being used for planning purposes only. It can represent a family or group of inventory items. It can have a planning bill of material (BOM) or planning routing, and it can exist as a component on a planning BOM. A planning item cannot be specified on a production or engineering BOM or routing, and it cannot be used as a component in a production. The quantity on hand will never be maintained.

In PeopleSoft Receivables, an individual receivable. An item can be an invoice, a credit memo, a debit memo, a write-off, or an adjustment.

item shuffle In PeopleSoft Enterprise Campus Solutions, a process that enables you to change a payment allocation without having to reverse the payment.

joint communication In PeopleSoft Enterprise Campus Solutions, one letter that is addressed jointly to two people. For example, a letter might be addressed to both Mr. Sudhir Awat and Ms. Samantha Mortelli. A relationship must be established between the two individuals in the database, and at least one of the individuals must have an ID in the database.

keyword In PeopleSoft Enterprise Campus Solutions, a term that you link to particular elements within PeopleSoft Student Financials, Financial Aid, and Contributor Relations. You can use keywords as search criteria that enable you to locate specific records in a search dialog box.

KPI An abbreviation for *key performance indicator.* A high-level measurement of how well an organization is doing in achieving critical success factors. This defines the data value or calculation upon which an assessment is determined.

LDIF file Abbreviation for *Lightweight Directory Access Protocol (LDAP) Data Interchange Format file.* Contains discrepancies between PeopleSoft data and directory data.

learner group In PeopleSoft Enterprise Learning Management, a group of learners who are linked to the same learning environment. Members of the learner group can share the same attributes, such as the same department or job code. Learner groups are used to control access to and enrollment in learning activities and programs. They are also used to perform group enrollments and mass enrollments in the back office.

learning components In PeopleSoft Enterprise Learning Management, the foundational building blocks of learning activities. PeopleSoft Enterprise Learning Management supports six basic types of learning components: web-based, session, webcast, test, survey, and assignment. One or more of these learning component types compose a single learning activity.

learning environment In PeopleSoft Enterprise Learning Management, identifies a set of categories and catalog items that can be made available to learner groups. Also defines the default values that are assigned to the learning activities and programs that are created within a particular learning environment. Learning environments provide a way to partition the catalog so that learners see only those items that are relevant to them.

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

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

library section In PeopleSoft Enterprise Incentive Management, a section that is defined in a plan (or template) and that is available for other plans to share. Changes to a library section are reflected in all plans that use it.

linked section In PeopleSoft Enterprise Incentive Management, a section that is defined in a plan template but appears in a plan. Changes to linked sections propagate to plans using that section.

linked variable In PeopleSoft Enterprise Incentive Management, a variable that is defined and maintained in a plan template and that also appears in a plan. Changes to linked variables propagate to plans using that variable.

LMS Abbreviation for *learning management system.* In PeopleSoft Enterprise Campus Solutions, LMS is a PeopleSoft Student Records feature that provides a common set of interoperability standards that enable the sharing of instructional content and data between learning and administrative environments.

load In PeopleSoft Inventory, identifies a group of goods that are shipped together. Load management is a feature of PeopleSoft Inventory that is used to track the weight, the volume, and the destination of a shipment.

local functionality In PeopleSoft HRMS, the set of information that is available for a specific country. You can access this information when you click the appropriate country flag in the global window, or when you access it by a local country menu.

location Locations enable you to indicate the different types of addresses—for a company, for example, one address to receive bills, another for shipping, a third for postal deliveries, and a separate street address. Each address has a different location number. The primary location—indicated by a *1*—is the address you use most often and may be different from the main address.

logistical task In PeopleSoft Services Procurement, an administrative task that is related to hiring a service provider. Logistical tasks are linked to the service type on the work order so that different types of services can have different logistical tasks. Logistical tasks include both preapproval tasks (such as assigning a new badge or ordering a new laptop) and postapproval tasks (such as scheduling orientation or setting up the service provider email). The logistical tasks can be mandatory or optional. Mandatory preapproval tasks must be completed before the work order is approved. Mandatory postapproval tasks, on the other hand, must be completed before a work order is released to a service provider.

market template In PeopleSoft Enterprise Incentive Management, additional functionality that is specific to a given market or industry and is built on top of a product category.

mass change In PeopleSoft Enterprise Campus Solutions, mass change is a SQL generator that can be used to create specialized functionality. Using mass change, you can set up a series of Insert, Update, or Delete SQL statements to perform business functions that are specific to the institution.

See also *3C engine*.

match group In PeopleSoft Receivables, a group of receivables items and matching offset items. The system creates match groups by using user-defined matching criteria for selected field values.

MCF server Abbreviation for *PeopleSoft MultiChannel Framework server.* Comprises the universal queue server and the MCF log server. Both processes are started when *MCF Servers* is selected in an application server domain configuration.

merchandising activity In PeopleSoft Promotions Management, a specific discount type that is associated with a trade promotion (such as off-invoice, billback or rebate, or lump-sum payment) that defines the performance that is required to receive the discount. In the industry, you may know this as an offer, a discount, a merchandising event, an event, or a tactic.

meta-SQL Meta-SQL constructs expand into platform-specific Structured Query Language (SQL) substrings. They are used in functions that pass SQL strings, such as in SQL objects, the SQLExec function, and PeopleSoft Application Engine programs.

metastring Metastrings are special expressions included in SQL string literals. The metastrings, prefixed with a percent (%) symbol, are included directly in the string literals. They expand at run time into an appropriate substring for the current database platform.

multibook In PeopleSoft General Ledger, multiple ledgers having multiple-base currencies that are defined for a business unit, with the option to post a single transaction to all base currencies (all ledgers) or to only one of those base currencies (ledgers).

multicurrency The ability to process transactions in a currency other than the business unit's base currency.

national allowance In PeopleSoft Promotions Management, a promotion at the corporate level that is funded by nondiscretionary dollars. In the industry, you may know this as a national promotion, a corporate promotion, or a corporate discount.

need In PeopleSoft Enterprise Campus Solutions, the difference between the cost of attendance (COA) and the expected family contribution (EFC). It is the gap between the cost of attending the school and the student's resources. The financial aid package is based on the amount of financial need. The process of determining a student's need is called *need analysis.*

node-oriented tree A tree that is based on a detail structure, but the detail values are not used.

pagelet Each block of content on the home page is called a pagelet. These pagelets display summary information within a small rectangular area on the page. The pagelet provide users with a snapshot of their most relevant PeopleSoft and non-PeopleSoft content.

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

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

See also *compensation object.*

partner A company that supplies products or services that are resold or purchased by the enterprise.

pay cycle In PeopleSoft Payables, a set of rules that define the criteria by which it should select scheduled payments for payment creation.

payment shuffle In PeopleSoft Enterprise Campus Solutions, a process allowing payments that have been previously posted to a student's account to be automatically reapplied when a higher priority payment is posted or the payment allocation definition is changed.

pending item In PeopleSoft Receivables, an individual receivable (such as an invoice, a credit memo, or a write-off) that has been entered in or created by the system, but hasn't been posted.

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

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

PeopleSoft Internet Architecture The fundamental architecture on which PeopleSoft 8 applications are constructed, consisting of a relational database management system (RDBMS), an application server, a web server, and a browser.

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

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

person of interest A person about whom the organization maintains information but who is not part of the workforce.

personal portfolio In PeopleSoft Enterprise Campus Solutions, the user-accessible menu item that contains an individual's name, address, telephone number, and other personal information.

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

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

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

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

planning instance In PeopleSoft Supply Planning, a set of data (business units, items, supplies, and demands) constituting the inputs and outputs of a supply plan.

population In PeopleSoft Enterprise Campus Solutions, the middle level of the three-level classification structure that you define in PeopleSoft Enterprise Recruiting and Admissions for enrollment management. You can define a population level, link it to other levels, and set enrollment target numbers for it.

See also *division* and *cohort*.

portal registry In PeopleSoft applications, the portal registry is a tree-like structure in which content references are organized, classified, and registered. It is a central repository that defines both the structure and content of a portal through a hierarchical, tree-like structure of folders useful for organizing and securing content references.

price list In PeopleSoft Enterprise Pricer, enables you to select products and conditions for which the price list applies to a transaction. During a transaction, the system either determines the product price based on the predefined search hierarchy for the transaction or uses the product's lowest price on any associated, active price lists. This price is used as the basis for any further discounts and surcharges.

price rule In PeopleSoft Enterprise Pricer, defines the conditions that must be met for adjustments to be applied to the base price. Multiple rules can apply when conditions of each rule are met.

price rule condition In PeopleSoft Enterprise Pricer, selects the price-by fields, the values for the price-by fields, and the operator that determines how the price-by fields are related to the transaction.

price rule key In PeopleSoft Enterprise Pricer, defines the fields that are available to define price rule conditions (which are used to match a transaction) on the price rule.

primacy number In PeopleSoft Enterprise Campus Solutions, a number that the system uses to prioritize financial aid applications when students are enrolled in multiple academic careers and academic programs at the same time. The Consolidate Academic Statistics process uses the primacy number indicated for both the career and program at the institutional level to determine a student's primary career and program. The system also uses the

	number to determine the primary student attribute value that is used when you extract data to report on cohorts. The lowest number takes precedence.
primary name type	In PeopleSoft Enterprise Campus Solutions, the name type that is used to link the name stored at the highest level within the system to the lower-level set of names that an individual provides.
process category	In PeopleSoft Process Scheduler, processes that are grouped for server load balancing and prioritization.
process group	In PeopleSoft Financials, a group of application processes (performed in a defined order) that users can initiate in real time, directly from a transaction entry page.
process definition	Process definitions define each run request.
process instance	A unique number that identifies each process request. This value is automatically incremented and assigned to each requested process when the process is submitted to run.
process job	You can link process definitions into a job request and process each request serially or in parallel. You can also initiate subsequent processes based on the return code from each prior request.
process request	A single run request, such as a Structured Query Report (SQR), a COBOL or Application Engine program, or a Crystal report that you run through PeopleSoft Process Scheduler.
process run control	A PeopleTools variable used to retain PeopleSoft Process Scheduler values needed at runtime for all requests that reference a run control ID. Do not confuse these with application run controls, which may be defined with the same run control ID, but only contain information specific to a given application process request.
product category	In PeopleSoft Enterprise Incentive Management, indicates an application in the Enterprise Incentive Management suite of products. Each transaction in the PeopleSoft Enterprise Incentive Management system is associated with a product category.
programs	In PeopleSoft Enterprise Learning Management, a high-level grouping that guides the learner along a specific learning path through sections of catalog items. PeopleSoft Enterprise Learning Systems provides two types of programs—curricula and certifications.
progress log	In PeopleSoft Services Procurement, tracks deliverable-based projects. This is similar to the time sheet in function and process. The service provider contact uses the progress log to record and submit progress on deliverables. The progress can be logged by the activity that is performed, by the percentage of work that is completed, or by the completion of milestone activities that are defined for the project.
project transaction	In PeopleSoft Project Costing, an individual transaction line that represents a cost, time, budget, or other transaction row.
promotion	In PeopleSoft Promotions Management, a trade promotion, which is typically funded from trade dollars and used by consumer products manufacturers to increase sales volume.
prospects	In PeopleSoft Enterprise Campus Solutions, students who are interested in applying to the institution. In PeopleSoft Enterprise Contributor Relations, individuals and organizations that are most likely to make substantial financial commitments or other types of commitments to the institution.
publishing	In PeopleSoft Enterprise Incentive Management, a stage in processing that makes incentive-related results available to participants.

rating components In PeopleSoft Enterprise Campus Solutions, variables used with the Equation Editor to retrieve specified populations.

record group A set of logically and functionally related control tables and views. Record groups help enable TableSet sharing, which eliminates redundant data entry. Record groups ensure that TableSet sharing is applied consistently across all related tables and views.

record input VAT flag Abbreviation for *record input value-added tax flag*. Within PeopleSoft Purchasing, Payables, and General Ledger, this flag indicates that you are recording input VAT on the transaction. This flag, in conjunction with the record output VAT flag, is used to determine the accounting entries created for a transaction and to determine how a transaction is reported on the VAT return. For all cases within Purchasing and Payables where VAT information is tracked on a transaction, this flag is set to Yes. This flag is not used in PeopleSoft Order Management, Billing, or Receivables, where it is assumed that you are always recording only output VAT, or in PeopleSoft Expenses, where it is assumed that you are always recording only input VAT.

record output VAT flag Abbreviation for *record output value-added tax flag.*

See *record input VAT flag.*

recname The name of a record that is used to determine the associated field to match a value or set of values.

recognition In PeopleSoft Enterprise Campus Solutions, the recognition type indicates whether the PeopleSoft Enterprise Contributor Relations donor is the primary donor of a commitment or shares the credit for a donation. Primary donors receive hard credit that must total 100 percent. Donors that share the credit are given soft credit. Institutions can also define other share recognition-type values such as memo credit or vehicle credit.

reference data In PeopleSoft Sales Incentive Management, system objects that represent the sales organization, such as territories, participants, products, customers, channels, and so on.

reference object In PeopleSoft Enterprise Incentive Management, this dimension-type object further defines the business. Reference objects can have their own hierarchy (for example, product tree, customer tree, industry tree, and geography tree).

reference transaction In commitment control, a reference transaction is a source transaction that is referenced by a higher-level (and usually later) source transaction, in order to automatically reverse all or part of the referenced transaction's budget-checked amount. This avoids duplicate postings during the sequential entry of the transaction at different commitment levels. For example, the amount of an encumbrance transaction (such as a purchase order) will, when checked and recorded against a budget, cause the system to concurrently reference and relieve all or part of the amount of a corresponding pre-encumbrance transaction, such as a purchase requisition.

regional sourcing In PeopleSoft Purchasing, provides the infrastructure to maintain, display, and select an appropriate vendor and vendor pricing structure that is based on a regional sourcing model where the multiple ship to locations are grouped. Sourcing may occur at a level higher than the ship to location.

relationship object In PeopleSoft Enterprise Incentive Management, these objects further define a compensation structure to resolve transactions by establishing associations between compensation objects and business objects.

remote data source data Data that is extracted from a separate database and migrated into the local database.

REN server Abbreviation for *real-time event notification server* in PeopleSoft MultiChannel Framework.

requester In PeopleSoft eSettlements, an individual who requests goods or services and whose ID appears on the various procurement pages that reference purchase orders.

reversal indicator In PeopleSoft Enterprise Campus Solutions, an indicator that denotes when a particular payment has been reversed, usually because of insufficient funds.

role Describes how people fit into PeopleSoft Workflow. A role is a class of users who perform the same type of work, such as clerks or managers. Your business rules typically specify what user role needs to do an activity.

role user A PeopleSoft Workflow user. A person's role user ID serves much the same purpose as a user ID does in other parts of the system. PeopleSoft Workflow uses role user IDs to determine how to route worklist items to users (through an email address, for example) and to track the roles that users play in the workflow. Role users do not need PeopleSoft user IDs.

roll up In a tree, to roll up is to total sums based on the information hierarchy.

run control A run control is a type of online page that is used to begin a process, such as the batch processing of a payroll run. Run control pages generally start a program that manipulates data.

run control ID A unique ID to associate each user with his or her own run control table entries.

run-level context In PeopleSoft Enterprise Incentive Management, associates a particular run (and batch ID) with a period context and plan context. Every plan context that participates in a run has a separate run-level context. Because a run cannot span periods, only one run-level context is associated with each plan context.

search query You use this set of objects to pass a query string and operators to the search engine. The search index returns a set of matching results with keys to the source documents.

search/match In PeopleSoft Enterprise Campus Solutions and PeopleSoft Enterprise Human Resources Management Solutions, a feature that enables you to search for and identify duplicate records in the database.

seasonal address In PeopleSoft Enterprise Campus Solutions, an address that recurs for the same length of time at the same time of year each year until adjusted or deleted.

section In PeopleSoft Enterprise Incentive Management, a collection of incentive rules that operate on transactions of a specific type. Sections enable plans to be segmented to process logical events in different sections.

security event In commitment control, security events trigger security authorization checking, such as budget entries, transfers, and adjustments; exception overrides and notifications; and inquiries.

serial genealogy In PeopleSoft Manufacturing, the ability to track the composition of a specific, serial-controlled item.

serial in production In PeopleSoft Manufacturing, enables the tracing of serial information for manufactured items. This is maintained in the Item Master record.

service impact In PeopleSoft Enterprise Campus Solutions, the resulting action triggered by a service indicator. For example, a service indicator that reflects nonpayment of account balances by a student might result in a service impact that prohibits registration for classes.

service indicator In PeopleSoft Enterprise Campus Solutions, indicates services that may be either withheld or provided to an individual. Negative service indicators indicate holds that prevent the individual from receiving specified services, such as check-cashing privileges or registration for classes. Positive service indicators designate special services that are provided to the individual, such as front-of-line service or special services for disabled students.

session In PeopleSoft Enterprise Campus Solutions, time elements that subdivide a term into multiple time periods during which classes are offered. In PeopleSoft Contributor Relations, a session is the means of validating gift, pledge, membership, or adjustment data entry . It controls access to the data entered by a specific user ID. Sessions are balanced, queued, and then posted to the institution's financial system. Sessions must be posted to enter a matching gift or pledge payment, to make an adjustment, or to process giving clubs or acknowledgements.

In PeopleSoft Enterprise Learning Management, a single meeting day of an activity (that is, the period of time between start and finish times within a day). The session stores the specific date, location, meeting time, and instructor. Sessions are used for scheduled training.

session template In PeopleSoft Enterprise Learning Management, enables you to set up common activity characteristics that may be reused while scheduling a PeopleSoft Enterprise Learning Management activity—characteristics such as days of the week, start and end times, facility and room assignments, instructors, and equipment. A session pattern template can be attached to an activity that is being scheduled. Attaching a template to an activity causes all of the default template information to populate the activity session pattern.

setup relationship In PeopleSoft Enterprise Incentive Management, a relationship object type that associates a configuration plan with any structure node.

share driver expression In PeopleSoft Business Planning, a named planning method similar to a driver expression, but which you can set up globally for shared use within a single planning application or to be shared between multiple planning applications through PeopleSoft Enterprise Warehouse.

single signon With single signon, users can, after being authenticated by a PeopleSoft application server, access a second PeopleSoft application server without entering a user ID or password.

source key process In PeopleSoft Enterprise Campus Solutions, a process that relates a particular transaction to the source of the charge or financial aid. On selected pages, you can drill down into particular charges.

source transaction In commitment control, any transaction generated in a PeopleSoft or third-party application that is integrated with commitment control and which can be checked against commitment control budgets. For example, a pre-encumbrance, encumbrance, expenditure, recognized revenue, or collected revenue transaction.

speed key See *communication key.*

SpeedChart A user-defined shorthand key that designates several ChartKeys to be used for voucher entry. Percentages can optionally be related to each ChartKey in a SpeedChart definition.

SpeedType A code representing a combination of ChartField values. SpeedTypes simplify the entry of ChartFields commonly used together.

staging A method of consolidating selected partner offerings with the offerings from the enterprise's other partners.

standard letter code In PeopleSoft Enterprise Campus Solutions, a standard letter code used to identify each letter template available for use in mail merge functions. Every letter generated in the system must have a standard letter code identification.

statutory account Account required by a regulatory authority for recording and reporting financial results. In PeopleSoft, this is equivalent to the Alternate Account (ALTACCT) ChartField.

step In PeopleSoft Sales Incentive Management, a collection of sections in a plan. Each step corresponds to a step in the job run.

storage level In PeopleSoft Inventory, identifies the level of a material storage location. Material storage locations are made up of a business unit, a storage area, and a storage level. You can set up to four storage levels.

subcustomer qualifier A value that groups customers into a division for which you can generate detailed history, aging, events, and profiles.

Summary ChartField You use summary ChartFields to create summary ledgers that roll up detail amounts based on specific detail values or on selected tree nodes. When detail values are summarized using tree nodes, summary ChartFields must be used in the summary ledger data record to accommodate the maximum length of a node name (20 characters).

summary ledger An accounting feature used primarily in allocations, inquiries, and PS/nVision reporting to store combined account balances from detail ledgers. Summary ledgers increase speed and efficiency of reporting by eliminating the need to summarize detail ledger balances each time a report is requested. Instead, detail balances are summarized in a background process according to user-specified criteria and stored on summary ledgers. The summary ledgers are then accessed directly for reporting.

summary time period In PeopleSoft Business Planning, any time period (other than a base time period) that is an aggregate of other time periods, including other summary time periods and base time periods, such as quarter and year total.

summary tree A tree used to roll up accounts for each type of report in summary ledgers. Summary trees enable you to define trees on trees. In a summary tree, the detail values are really nodes on a detail tree or another summary tree (known as the *basis* tree). A summary tree structure specifies the details on which the summary trees are to be built.

syndicate To distribute a production version of the enterprise catalog to partners.

system function In PeopleSoft Receivables, an activity that defines how the system generates accounting entries for the general ledger.

TableSet A means of sharing similar sets of values in control tables, where the actual data values are different but the structure of the tables is the same.

TableSet sharing Shared data that is stored in many tables that are based on the same TableSets. Tables that use TableSet sharing contain the SETID field as an additional key or unique identifier.

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

tax authority In PeopleSoft Enterprise Campus Solutions, a user-defined element that combines a description and percentage of a tax with an account type, an item type, and a service impact.

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

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

3C engine Abbreviation for *Communications, Checklists, and Comments engine.* In PeopleSoft Enterprise Campus Solutions, the 3C engine enables you to automate business processes that involve additions, deletions, and updates to communications, checklists,

	and comments. You define events and triggers to engage the engine, which runs the mass change and processes the 3C records (for individuals or organizations) immediately and automatically from within business processes.
3C group	Abbreviation for *Communications, Checklists, and Comments group.* In PeopleSoft Enterprise Campus Solutions, a method of assigning or restricting access privileges. A 3C group enables you to group specific communication categories, checklist codes, and comment categories. You can then assign the group inquiry-only access or update access, as appropriate.
TimeSpan	A relative period, such as year-to-date or current period, that can be used in various PeopleSoft General Ledger functions and reports when a rolling time frame, rather than a specific date, is required. TimeSpans can also be used with flexible formulas in PeopleSoft Projects.
trace usage	In PeopleSoft Manufacturing, enables the control of which components will be traced during the manufacturing process. Serial- and lot-controlled components can be traced. This is maintained in the Item Master record.
transaction allocation	In PeopleSoft Enterprise Incentive Management, the process of identifying the owner of a transaction. When a raw transaction from a batch is allocated to a plan context, the transaction is duplicated in the PeopleSoft Enterprise Incentive Management transaction tables.
transaction state	In PeopleSoft Enterprise Incentive Management, a value assigned by an incentive rule to a transaction. Transaction states enable sections to process only transactions that are at a specific stage in system processing. After being successfully processed, transactions may be promoted to the next transaction state and "picked up" by a different section for further processing.
Translate table	A system edit table that stores codes and translate values for the miscellaneous fields in the database that do not warrant individual edit tables of their own.
tree	The graphical hierarchy in PeopleSoft systems that displays the relationship between all accounting units (for example, corporate divisions, projects, reporting groups, account numbers) and determines roll-up hierarchies.
tuition lock	In PeopleSoft Enterprise Campus Solutions, a feature in the Tuition Calculation process that enables you to specify a point in a term after which students are charged a minimum (or *locked*) fee amount. Students are charged the locked fee amount even if they later drop classes and take less than the normal load level for that tuition charge.
unclaimed transaction	In PeopleSoft Enterprise Incentive Management, a transaction that is not claimed by a node or participant after the allocation process has completed, usually due to missing or incomplete data. Unclaimed transactions may be manually assigned to the appropriate node or participant by a compensation administrator.
universal navigation header	Every PeopleSoft portal includes the universal navigation header, intended to appear at the top of every page as long as the user is signed on to the portal. In addition to providing access to the standard navigation buttons (like Home, Favorites, and signoff) the universal navigation header can also display a welcome message for each user.
update access	In PeopleSoft Enterprise Campus Solutions, a type of security access that permits the user to edit and update data. See also *inquiry access*.
user interaction object	In PeopleSoft Sales Incentive Management, used to define the reporting components and reports that a participant can access in his or her context. All Sales Incentive Management user interface objects and reports are registered as user interaction objects. User interaction objects can be linked to a compensation structure node through a compensation relationship object (individually or as groups).

variable In PeopleSoft Sales Incentive Management, the intermediate results of calculations. Variables hold the calculation results and are then inputs to other calculations. Variables can be plan variables that persist beyond the run of an engine or local variables that exist only during the processing of a section.

VAT exception Abbreviation for *value-added tax exception.* A temporary or permanent exemption from paying VAT that is granted to an organization. This terms refers to both VAT exoneration and VAT suspension.

VAT exempt Abbreviation for *value-added tax exempt.* Describes goods and services that are not subject to VAT. Organizations that supply exempt goods or services are unable to recover the related input VAT. This is also referred to as exempt without recovery.

VAT exoneration Abbreviation for *value-added tax exoneration.* An organization that has been granted a permanent exemption from paying VAT due to the nature of that organization.

VAT suspension Abbreviation for *value-added tax suspension.* An organization that has been granted a temporary exemption from paying VAT.

warehouse A PeopleSoft data warehouse that consists of predefined ETL maps, data warehouse tools, and DataMart definitions.

work order In PeopleSoft Services Procurement, enables an enterprise to create resource-based and deliverable-based transactions that specify the basic terms and conditions for hiring a specific service provider. When a service provider is hired, the service provider logs time or progress against the work order.

worker A person who is part of the workforce; an employee or a contingent worker.

workset A group of people and organizations that are linked together as a set. You can use worksets to simultaneously retrieve the data for a group of people and organizations and work with the information on a single page.

worksheet A way of presenting data through a PeopleSoft Business Analysis Modeler interface that enables users to do in-depth analysis using pivoting tables, charts, notes, and history information.

worklist The automated to-do list that PeopleSoft Workflow creates. From the worklist, you can directly access the pages you need to perform the next action, and then return to the worklist for another item.

XML schema An XML definition that standardizes the representation of application messages, component interfaces, or business interlinks.

yield by operation In PeopleSoft Manufacturing, the ability to plan the loss of a manufactured item on an operation-by-operation basis.

zero-rated VAT Abbreviation for *zero-rated value-added tax.* A VAT transaction with a VAT code that has a tax percent of zero. Used to track taxable VAT activity where no actual VAT amount is charged. Organizations that supply zero-rated goods and services can still recover the related input VAT. This is also referred to as exempt with recovery.

Index