



*Open Tools from Sybase, Inc.*

**PowerBuilder**

***Version Control Interfaces***

*Version 6*

**Power  
Builder®**

AA0524

October 1997

Copyright © 1991-1997 Sybase, Inc. and its subsidiaries.

All rights reserved.

Printed in Ireland.

Information in this manual may change without notice and does not represent a commitment on the part of Sybase, Inc. and its subsidiaries.

The software described in this manual is provided by Powersoft Corporation under a Powersoft License agreement. The software may be used only in accordance with the terms of the agreement.

No part of this publication may be reproduced, transmitted, or translated in any form or by any means, electronic, mechanical, manual, optical, or otherwise, without the prior written permission of Sybase, Inc. and its subsidiaries.

Sybase, Inc. and its subsidiaries claim copyright in this program and documentation as an unpublished work, revisions of which were first licensed on the date indicated in the foregoing notice. Claim of copyright does not imply waiver of other rights of Sybase, Inc. and its subsidiaries.

ClearConnect, ComponentPack, InfoMaker, PowerBuilder, Powersoft, S-Designor, SQL SMART, and Sybase are registered trademarks of Sybase, Inc. and its subsidiaries. Adaptive Component Architecture, Adaptive Warehouse, AppModeler, Column Design, DataArchitect, DataExpress, Data Pipeline, DataWindow, dbQueue, ImpactNow, InstaHelp, Jaguar CTS, jConnect for JDBC, MetaWorks, NetImpact, ObjectCycle, Optima++, Power++, PowerAMC, PowerBuilder Foundation Class Library, PowerDesigner, Power J, PowerScript, PowerSite, Powersoft Portfolio, Powersoft Professional, PowerTips, ProcessAnalyst, Runtime Kit for Unicode, SQL Anywhere, The Model For Client/Server Solutions, The Future Is Wide Open, Translation Toolkit, UNIBOM, Unilib, Uninull, Unisep, Unistring, Viewer, WarehouseArchitect, Watcom, Watcom SQL Server, Web.PB, and Web.SQL are trademarks of Sybase, Inc. or its subsidiaries. Certified PowerBuilder Developer and CPD are service marks of Sybase, Inc. or its subsidiaries. DataWindow is a patented proprietary technology of Sybase, Inc. or its subsidiaries.

AccuFonts is a trademark of AccuWare Business Solutions Ltd.

All other trademarks are the property of their respective owners.

# Contents

|                              |   |           |
|------------------------------|---|-----------|
| <b>About This Book .....</b> | <b>ix</b>   |           |
| <br>                         |   |           |
| <b>1</b>                     | <b>Preparing to Use Version Control with PowerBuilder .....</b> | <b>1</b>  |
|                              | About version control systems .....                             | 2         |
|                              | PowerBuilder libraries .....                                    | 2         |
|                              | Versions and version labels .....                               | 2         |
|                              | Version control interfaces .....                                | 3         |
|                              | Using a version control system .....                            | 5         |
|                              | Setting up your environment .....                               | 7         |
|                              | Using the PowerBuilder SCC API .....                            | 8         |
|                              | About version control features .....                            | 8         |
|                              | Setting up the PowerBuilder SCC API .....                       | 9         |
|                              | Working with the PowerBuilder SCC API .....                     | 12        |
| <br>                         |   |           |
| <b>2</b>                     | <b>Using the PowerBuilder ObjectCycle Interface .....</b>       | <b>19</b> |
|                              | About ObjectCycle .....   | 20        |
|                              | About the ObjectCycle interface .....                           | 21        |
|                              | Installing the required software .....                          | 23        |
|                              | Setting up projects .....                                       | 26        |
|                              | Connecting to ObjectCycle for the first time .....              | 27        |
|                              | Creating and saving the configuration file .....                | 28        |
|                              | Defining project nodes .....                                    | 30        |
|                              | Creating work libraries .....                                   | 31        |
|                              | Registering PowerBuilder objects .....                          | 34        |
|                              | Registering objects .....                                       | 34        |
|                              | Listing registered objects .....                                | 35        |
|                              | Clearing an object's registration .....                         | 36        |
|                              | Opening read-only versions of registered objects .....          | 37        |
|                              | Checking out objects from ObjectCycle .....                     | 38        |
|                              | Checking out objects .....                                      | 38        |
|                              | Viewing the status of checked-out objects .....                 | 40        |
|                              | Modifying checked-out objects .....                             | 41        |
|                              | Checking in objects to ObjectCycle .....                        | 42        |

|   |    |
|---|----|
| Creating a new release .....                            | 44 |
| About setting options for the new archive .....         | 44 |
| About specifying a starting version number .....        | 44 |
| Creating the release .....                              | 44 |
| Using version labels .....                              | 47 |
| Assigning version labels to a group of objects .....    | 47 |
| Filtering version lists by version labels .....         | 49 |
| Clearing the filter .....                               | 51 |
| Assigning version labels when you build a project ..... | 52 |
| Viewing an object's version history .....               | 53 |
| Displaying reports .....                                | 54 |
| Displaying an archive report .....                      | 54 |
| Displaying a revision report .....                      | 55 |
| Copying a report to a file .....                        | 56 |
| Restoring earlier versions of an object .....           | 57 |
| Deciding which version to restore .....                 | 57 |
| Restoring an earlier version .....                      | 57 |
| Restoring versions by version labels .....              | 58 |
| Restoring libraries .....                               | 60 |
| Listing the objects in a project .....                  | 60 |
| About the methods for restoring libraries .....         | 62 |
| Retrieving the project object from ObjectCycle .....    | 63 |
| Saving the project object with a new name .....         | 65 |
| Synchronizing objects .....                             | 66 |

**3**

|  |           |
|--|-----------|
| <b>Using the PowerBuilder PVCS Interface .....</b>       | <b>67</b> |
| About the PVCS interface .....                           | 68        |
| Configuring the PVCS interface .....                     | 69        |
| Confirming your installation .....                       | 69        |
| Connecting to PVCS the first time .....                  | 69        |
| Specifying a configuration file .....                    | 70        |
| Creating work libraries .....                            | 73        |
| Registering your objects .....                           | 74        |
| Viewing a list of registered objects .....               | 76        |
| Clearing an object's registered status .....             | 77        |
| Checking objects out of PVCS .....                       | 79        |
| Checking out objects .....                               | 79        |
| Viewing the check-out status of objects .....            | 80        |
| Modifying objects .....                                  | 82        |
| Checking objects in to PVCS .....                        | 83        |
| Using version labels .....                               | 85        |
| Assigning a version label to a group of objects .....    | 85        |
| Using version labels to filter a list of revisions ..... | 86        |
| Retrieving revisions by version label .....              | 87        |



|  |     |
|--|-----|
| Creating a new release .....                           | 88  |
| To create a new release .....                          | 88  |
| Viewing an object's change history .....               | 90  |
| Reporting.....   | 91  |
| Viewing or printing a revision report .....            | 91  |
| Viewing or printing an archive report.....             | 92  |
| Restoring an earlier revision level of an object ..... | 94  |
| Deciding which revision level to restore .....         | 94  |
| Performing the restoration .....                       | 94  |
| Restoring libraries .....                              | 97  |
| Viewing a list of objects in a project .....           | 97  |
| Two methods of restoring libraries .....               | 98  |
| Synchronizing objects .....                            | 102 |

|          |   |            |
|----------|---|------------|
| <b>4</b> | <b>Using the PowerBuilder MKS Source Integrity Interface.....</b> | <b>103</b> |
|          | Basic steps for using the Source Integrity interface .....        | 104        |
|          | About the Source Integrity interface .....                        | 105        |
|          | Installing the required software .....                            | 107        |
|          | Connecting to Source Integrity for the first time .....           | 108        |
|          | Defining the configuration file .....                             | 110        |
|          | Creating and saving the configuration file .....                  | 110        |
|          | Defining archive directories .....                                | 112        |
|          | Specifying a user ID .....  | 113        |
|          | Creating work libraries .....                                     | 115        |
|          | Creating the work library .....                                   | 115        |
|          | Adding the work library to the library search path .....          | 116        |
|          | Using objects from your work library .....                        | 117        |
|          | Registering PowerBuilder objects .....                            | 118        |
|          | Registering objects.....  | 118        |
|          | Listing registered objects.....                                   | 120        |
|          | Clearing an object's registration .....                           | 120        |
|          | Opening read-only versions of registered objects .....            | 121        |
|          | Checking out objects from Source Integrity .....                  | 123        |
|          | Checking out objects .....  | 123        |
|          | Viewing the status of checked-out objects .....                   | 124        |
|          | Modifying checked-out objects.....                                | 126        |
|          | Checking in objects to Source Integrity .....                     | 127        |
|          | Creating a new release .....                                      | 129        |
|          | About setting options for the new archive .....                   | 129        |
|          | About specifying a starting revision number.....                  | 129        |
|          | Creating the release .....  | 130        |
|          | Using revision labels .....                                       | 132        |
|          | Assigning revision labels to a group of objects .....             | 132        |
|          | Filtering revision lists by revision labels .....                 | 134        |

|   |     |
|---|-----|
| Clearing the filter .....                                 | 137 |
| Assigning revision labels when you build a project .....  | 137 |
| Viewing an object's revision history .....                | 139 |
| Displaying reports .....                                  | 141 |
| Displaying an archive report.....                         | 141 |
| Displaying a revision report .....                        | 143 |
| Copying a report to a file .....                          | 144 |
| Restoring earlier revisions of an object .....            | 145 |
| Deciding which revision to restore.....                   | 145 |
| Restoring an earlier revision.....                        | 145 |
| Restoring revisions by revision labels .....              | 147 |
| Restoring libraries .....                                 | 149 |
| Listing the objects in a project.....                     | 149 |
| About the methods for restoring libraries .....           | 151 |
| Retrieving the project object from Source Integrity ..... | 153 |
| Saving the project object with a new name .....           | 154 |
| Synchronizing objects .....                               | 157 |

|          |   |            |
|----------|---|------------|
| <b>5</b> | <b>Using the PowerBuilder ENDEVOR Interface .....</b>           | <b>159</b> |
|          | About the ENDEVOR interface .....                               | 160        |
|          | System requirements .....                                       | 160        |
|          | Installation .....  | 161        |
|          | Step 1: install and configure PowerBuilder and<br>ENDEVOR ..... | 161        |
|          | Step 2: install the PowerBuilder ENDEVOR interface .....        | 162        |
|          | Step 3: edit the repository definitions .....                   | 163        |
|          | Step 4: define the repository to ENDEVOR.....                   | 166        |
|          | Invoking the interface .....                                    | 167        |
|          | Using version control.....                                      | 168        |
|          | Registering objects.....  | 168        |
|          | Displaying object histories.....                                | 169        |
|          | Checking out objects .....                                      | 169        |
|          | Displaying object check-out status .....                        | 170        |
|          | Checking in objects .....                                       | 170        |
|          | Clearing object check-out status .....                          | 171        |
|          | Deleting objects.....   | 171        |
|          | Reverting to a previous level.....                              | 172        |
|          | Re-registering objects .....                                    | 172        |
|          | Synchronizing objects .....                                     | 173        |
|          | Cross-referencing object names .....                            | 174        |
|          | Running reports.....  | 175        |

|          |  |            |
|----------|--|------------|
| <b>6</b> | <b>Using the PowerBuilder Apple SourceServer Interface .....</b> | <b>177</b> |
|----------|--|------------|

|   |     |
|---|-----|
| Overview of the Apple SourceServer interface .....      | 178 |
| Installing Apple SourceServer .....                     | 179 |
| Connecting to Apple SourceServer .....                  | 180 |
| Configuring the Apple SourceServer interface .....      | 181 |
| Setting up an application for source control .....      | 181 |
| Configuring an individual work environment.....         | 183 |
| Viewing a list of registered objects .....              | 186 |
| Clearing an object's registered status .....            | 186 |
| Modifying registered objects .....                      | 188 |
| Viewing a list of checked-out objects .....             | 188 |
| Checking out objects .....                              | 188 |
| Modifying objects.....                                  | 189 |
| Checking objects in to Apple SourceServer .....         | 190 |
| Assigning and using version labels .....                | 191 |
| Assigning a version label to a group of objects .....   | 191 |
| Filtering a list of revisions by version label .....    | 191 |
| Using object histories and running reports .....        | 193 |
| Getting an overview of an object's change history ..... | 193 |
| Generating change history reports .....                 | 193 |
| Restoring an earlier revision level of an object .....  | 195 |
| Synchronizing objects .....                             | 196 |



# About This Book

## Subject

This book describes how to use version control systems to manage the objects you build in PowerBuilder.

## Audience

This book is for anyone managing PowerBuilder objects through a version control system. It assumes you are familiar with the version control system; if you are not, you should consult its documentation.

## Supported version control interfaces

The following version control interfaces, described in this book, are supported on the noted platforms:

| <b>Version control interface</b>    | <b>Windows</b> | <b>Macintosh</b> |
|-------------------------------------|----------------|------------------|
| CA-ENDEVOR from Computer Associates | x              |                  |
| ObjectCycle from Powersoft          | x              |                  |
| PVCS Version Manager from INTERSOLV | x              |                  |
| Source Integrity from MKS           | x              |                  |
| Apple SourceServer from Apple       |                | x                |



# Preparing to Use Version Control with PowerBuilder

About this chapter

This chapter provides an overview of version control systems, describes how to prepare to use PowerBuilder with your version control system, and how to set up and use the PowerBuilder SCC API.

Contents

| <b>Topic</b>                   | <b>Page</b> |
|--------------------------------|-------------|
| About version control systems  | 2           |
| Setting up your environment    | 7           |
| Using the PowerBuilder SCC API | 8           |

## About version control systems

|               |  |
|---------------|--|
| What they are | <b>Version control systems</b> (or source code control systems) track and store the evolutionary history of software system components. In the context of PowerBuilder development, your <b>source</b> is a collection of objects stored in PowerBuilder libraries and you use a version control system to manage these objects. |
| Why use one   | Most version control systems provide disaster recovery protection and functions to help manage complex development processes. With a version control system, you can track the development history of objects in your PowerBuilder application, maintain archives, and restore previous revisions of objects, if necessary.      |

## PowerBuilder libraries

When you create and save objects (such as windows and menus) in a painter, PowerBuilder stores the objects in **PowerBuilder library (PBL)** files. When you open an object in a painter, PowerBuilder retrieves the object from the PBL.

PowerBuilder libraries can be public or private. A **public library** is shared by all developers. Your **work library** is a private library, accessible only by you, where you save the working copies of your checked-out objects.

**FOR INFO** For complete information about organizing and managing PowerBuilder libraries, see the *PowerBuilder User's Guide*.

## Versions and version labels

Versions identify the changes you make to your PowerBuilder libraries as you develop an application. You can use versions and version labels to identify these developmental stages.

A **version** is an iteration of an object under version control. Each time you check in an object to the public library, the version control system increments the version number. For example, if you check out an object with a version number of 1.0, when you check it back in its new version number is 1.1. At any time you can display the version history of objects in your current application.



A **version label** is a descriptive name (such as *Release\_1*) that you assign to identify a version. You can assign a version label to a group of objects that comprise a particular version of an application so you can identify them quickly.

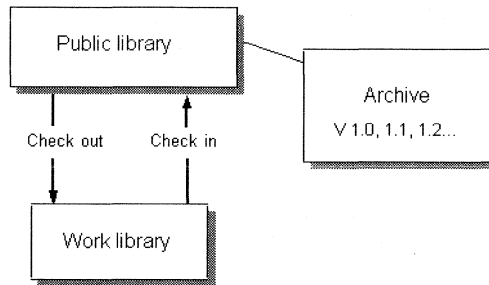
## Version control interfaces

You work with a version control system through a version control interface. Each version control system includes a unique version control interface.

### Version control interface components

Three components

A version control interface has three main components:



How you use them

| Component      | Description  | What you do  |
|----------------|--|--|
| Archive        | A location in the version control system that includes: <ul style="list-style-type: none"> <li>◆ The most recent version of PowerBuilder objects</li> <li>◆ A record of all changes made to the objects since they were put under version control</li> </ul> | Project manager: <ul style="list-style-type: none"> <li>◆ Creates an archive when projects are set up</li> <li>◆ Registers PowerBuilder objects in the archive (or project) folders</li> </ul>                                     |
| Public library | A library that: <ul style="list-style-type: none"> <li>◆ Is shared by all developers</li> <li>◆ Resides on a network server</li> <li>◆ Stores the most recent versions of objects</li> </ul>   | Developers: <ul style="list-style-type: none"> <li>◆ Check out an object from the public library to a work library to modify it</li> <li>◆ Restore an earlier version of an object from the archive to a public library</li> </ul> |

| Component    | Description  | What you do  |
|--------------|--|--|
| Work library | A library that: <ul style="list-style-type: none"><li>◆ Is accessible only by an individual developer</li><li>◆ Resides locally on a developer's computer</li><li>◆ Stores working versions of objects</li></ul> | Developers: <ul style="list-style-type: none"><li>◆ Create a work library</li><li>◆ Check in a modified object from a work library to the public library (this operation also updates the archive)</li></ul> |

## PowerBuilder interfaces

PowerBuilder includes two version control interfaces: a native interface and a source code control application programming interface (SCC API).

### PowerBuilder native interface

PowerBuilder has a native interface, PBnative, that provides only check-out/check-in facilities. This functionality is available to you from within PowerBuilder; you do not have to install any other version control system or prepare your environment.

### PowerBuilder SCC API

PowerBuilder also includes a common version control interface, SCC API, which is based on the Microsoft Common Source Code Control Interface Specification, Version 0.00.0823. You can use the PowerBuilder SCC API with any version control system that implements features defined in the Microsoft specification.

Among the advantages of using this interface with a version control system are that the PowerBuilder SCC API:

- ◆ Synchronizes the object in the public library with the archive as part of the check-out procedure
- ◆ Will be modified in future PowerBuilder releases to support functions added to version control systems to augment this Microsoft specification

FOR INFO For how to use the PowerBuilder SCC API, see "Using the PowerBuilder SCC API" on page 8.

## Other interfaces

Within PowerBuilder, you can work with other version control systems through their own proprietary interfaces (for example, ENDEVOR). These interfaces will not be updated.

If you are using one of these proprietary interfaces, you will need to install the version control interface after you install the version control system.

**FOR INFO** For how to use another version control interface, see the chapter in this book for your version control system.

## Comparing interfaces

The following table compares the two PowerBuilder version control interfaces with other version control interfaces currently supported by PowerBuilder:

| What the interface supports   | PowerBuilder native interface | PowerBuilder SCC API | Other interfaces |
|---|-------------------------------|----------------------|------------------|
| Basic version control functionality (such as check-out/check-in)                                  | x                             | x                    | x                |
| Advanced version control functionality (such as synchronizing the public library and the archive) |                               | x                    | x                |
| Microsoft Common Source Code Control Interface Specification, Version 0.00.0823                   |                               | x                    |                  |
| Future functionality  |                               | x                    |                  |

## Using a version control system

After the project manager installs a version control system, the developers can configure their individual environments to use that version control system.

### Installing a version control system

Setting up a version control system includes installing:

- ◆ Client-side software from the version control provider
 

**FOR INFO** For how to install this software, see the version control provider documentation.
- ◆ Appropriate PowerBuilder version control interface
 

**FOR INFO** For how to install an appropriate interface, see "Setting up your environment" on page 7.

Preparing your environment

Preparing your working environment includes:

- ◆ Defining a configuration file
- ◆ Creating a work library
- ◆ Adding your work library to your application library search path
- ◆ Registering your objects

**FOR INFO** For how to perform these tasks in PowerBuilder, see "Developer's tasks" on page 10 or the chapter for your version control system.

## Setting up your environment

How you prepare your environment for version control under PowerBuilder depends on whether you will use the PowerBuilder SCC API or the version control interface provided with your version control system. You do not need to install anything to use the PBnative interface; it is part of PowerBuilder and requires no preparation.

❖ **To set up your environment for the PowerBuilder SCC API:**

- ◆ Install the version control system, following the instructions from the version control system vendor.

**FOR INFO** For how to use the PowerBuilder SCC API with your version control system, see "Using the PowerBuilder SCC API" on page 8.

❖ **To set up your environment for another version control interface:**

- 1 Install the version control system, following the instructions from the version control system vendor.
- 2 Run the PowerBuilder setup program to install the PowerBuilder interface for the version control system you are using.
- 3 Prepare your environment for your version control interface.

**FOR INFO** For how to prepare your environment for your version control interface, see the appropriate chapter in this book.

## Using the PowerBuilder SCC API

The PowerBuilder SCC API provides a standard version control system interface.

### About version control features

Your version control system manages some version control functions while others are managed by the PowerBuilder SCC API:

| <b>This component</b>  | <b>Provides this functionality</b>  |
|------------------------|---|
| Version control system | Setting up a project<br>Assigning access permissions<br>Running reports<br>Retrieving earlier revisions of objects<br>Assigning revision labels<br>Using object histories                         |
| PowerBuilder SCC API   | Creating the configuration file<br>Creating a trace log<br>Configuring required check-in comments<br>Comparing working and registered objects<br>Synchronizing the public library and the archive |

**FOR INFO** For how your version control system handles these functions, see your version control system documentation. For how the PowerBuilder SCC API handles these functions, see the table below:

| <b>PowerBuilder SCC API feature</b>    | <b>Description</b>  | <b>See</b>  |
|--|---|---|
| Creating the configuration file        | Prompts you for information to create the project configuration file                                  | "Setting up the PowerBuilder SCC API" on page 9     |
| Creating a trace log                   | Creates a trace log of all version control activity that takes place during a Library painter session | "Enabling trace logging" on page 16                 |
| Configuring required check-in comments | Requires comments whenever an object is checked in to the public library                              | "Configuring required check-in comments" on page 17 |

| <b>PowerBuilder SCC API feature</b>              | <b>Description</b>  | <b>See</b>                             |
|--|---|--|
| Comparing working and registered objects         | Compares an object in your work library with the most recently registered version of the object | "Comparing library entries" on page 17 |
| Synchronizing the public library and the archive | Synchronizes objects in your public library with those in the version control system archive    | "Synchronizing objects" on page 18     |

## Setting up the PowerBuilder SCC API

Both the project manager and the developers must perform some set-up tasks before an application can be developed under version control.

FOR INFO See "Project manager's tasks" next for the project manager's tasks. See "Developer's tasks" on page 10 for the developer's tasks.

### Project manager's tasks

Before developers can work on an application under version control the project manager must configure the project by:

- ◆ Setting up a new project
- ◆ Assigning each developer permission to access the new project
- ◆ Registering the objects in the public library containing the application

### Setting up a project

A project, which points to the archive database, is set up in the version control system. The project manager must know the path to the project before configuring the application.

FOR INFO See the version control system documentation for how to set up a project.

### Assigning access permissions

Each developer who will work on a project must have access permissions to that project. The project manager sets these permissions in the version control system.

FOR INFO See the version control system documentation for how to set access permissions.

## Registering objects

Once the project manager has configured the application with its project, objects in the application must be registered in the public library. Developers must have access to the public library to check out/check in objects between this library and a work library, and to register new objects.

---

### To take advantage of version control

In order to take full advantage of version control and to avoid problems, you should register *all* objects in *all* your application libraries.

---

#### ❖ To register objects:

- 1 In the Library painter, find the library containing the objects you want to register.
- 2 Double-click the library to display the contents, and select the objects you want to register.
- 3 Select Source>Register from the menu bar and then complete the registration dialog box for each selected object.

PowerBuilder adds the object to the open project in the version control system and assigns a registration icon to each object in the public library.

---

### To view a registered object without making any changes

To view a registered object in read-only mode, open the object but do not check it out.

---

## Developer's tasks

Each developer working on the application must:

- ◆ Create a local configuration file
- ◆ Create one or more work libraries
- ◆ Modify the application's library search path
- ◆ Register objects created in the course of development



## Creating a local configuration file

Each developer needs a configuration file for each application. The file contains the developer's user ID and points to the local project path.

### ❖ To create a configuration file:

- 1 In the Application painter, open the application you want to configure.
- 2 In the Library painter, select Source>Connect from the menu bar.
- 3 Select SCC API in the Connect dialog box, then click OK.

If you have more than one version control system installed, you are prompted to select one. Click OK to accept your selection.

- 4 Click Yes when you are prompted to create a configuration file.
- 5 Complete the PowerBuilder SCC API Configuration dialog box:

**User ID** Type the user ID assigned to you in the version control system.

**Project name** Type the name of the project.

**Local project path** Type the path where temporary files will be written whenever you register, check out, or check in an object using the PowerBuilder SCC API.

---

### Specify a temporary path

Use a temporary local project path, such as C:\TEMP, instead of the default path, the directory containing the current application.

Temporary files are given the same name as files created when you select the Entry>Export menu item. If you export a file to the local project path directory, the exported file may be deleted as the PowerBuilder SCC API creates a temporary file with the same name.

---

- 6 (*Optional*) Click the Advanced button to access any advanced features the version control system supports.

This button is disabled if your version control system has no advanced configuration features.

---

### If you are using Object Cycle

In Object Cycle, use the advanced Object Cycle configuration features to set up your node list.

---

- 7 Click OK.

## Creating local work libraries

Each developer stores working copies of objects in a work library. When an object is checked out, a destination (work) library is required.

### ❖ To create a work library:

- 1 In the Library painter, click the Create button in the PainterBar.  
*or*  
Select Library>Create from the menu bar.
- 2 In the Create Library dialog box, navigate to where you want to locate your work library.
- 3 Specify a filename for your work library and click Save.
- 4 (*Optional*) Type a comment describing the purpose of the library.
- 5 Click OK.

## Modifying the library search path

When you create a work library, you must add that library to the library search path property of your application. If PowerBuilder cannot find your work library, objects in the application cannot be saved.

FOR INFO See the discussion of specifying the library search path in the application chapter of the *PowerBuilder User's Guide*.

## Registering new objects

You must register new objects in the version control system archive.

FOR INFO See "Registering objects" on page 10 for how to register objects.

## Working with the PowerBuilder SCC API

Once an application is configured, developers work with the PowerBuilder SCC API to modify the application.

## Connecting with the PowerBuilder SCC API

To take advantage of the PowerBuilder SCC API features, you must connect with the SCC API when you open the Library painter in the current application. After that, PowerBuilder associates the PowerBuilder SCC API with this application.

❖ **To connect to the PowerBuilder SCC API:**

- 1 In the Library painter, select Source>Connect from the menu bar.
- 2 Select SCC API in the Vendor listbox.

If your system has more than one version control system installed and configured for the PowerBuilder SCC API, you will be prompted to select one system.

## Modifying a registered object

Once objects are registered through PowerBuilder SCC API, you can:

- ◆ Check out an object
- ◆ Modify an object
- ◆ Check in an object
- ◆ Delete an object from the archive

**FOR INFO** For more information, see the discussion on working with an application object in the library chapter of the *PowerBuilder User's Guide*.

## Checking out an object

Checking out a registered object in a version control system gives you exclusive access to that object. When you check out an object through the PowerBuilder SCC API, PowerBuilder synchronizes the object in the public library with the archive.

You should only check out an object to a work library defined in your library search path. Your work library must exist before you check out an object, or the check-out process will fail.

---

### Checking out the application object

Always check out the application object by itself; PowerBuilder performs unique tasks when you check out the application object.

**FOR INFO** For more information, see the discussion on working with an application object in the library chapter of the *PowerBuilder User's Guide*.

---

❖ **To check out an object:**

- 1 Select the object.

You can check out multiple objects by selecting more than one object.

- 2 Click the Check Out button on the PainterBar.  
*or*  
Select Source>Check Out from menu bar.
- 3 In the Check Out Library Entries dialog box, select the work library and click Open.  
  
If the work library is not in the current application's library search path, you will not be able to save any modifications you make to the object.

#### What happens

When you check out an object, PowerBuilder:

- ◆ Makes a temporary copy of the object in the local project path and stores it in the public library you specified
- ◆ Copies the object from the public library to the work library
- ◆ Locks the object so no one else can modify it
- ◆ Displays a lock icon in the public library showing that the object is locked
- ◆ Displays a check-out icon in the work library showing the object has been checked out

---

#### **To build an executable using checked-out objects**

Be sure that your work library is listed *before* your public library in the library search path of your application. Otherwise, PowerBuilder will use the objects it finds in the public library instead of the objects in your work library.

---

#### **Modifying an object**

You should always check out objects before you modify them to make sure your work doesn't conflict with anyone else's. When you open an object that someone else has checked out, you get a read-only copy.

- ❖ **To open an object you have checked out for modification:**
  - ◆ Double-click the checked-out object in your work library.  
  
The painter associated with that object opens, displaying the object.

#### **Checking in an object**

Checking in an object updates the version control system archive and makes the object available to other developers.

It is best to check in objects as follows:

- ◆ **Your objects** Check in your objects as soon as you have finished modifying and testing them to preserve your changes and make the objects available for others to modify.
- ◆ **All objects** Check in all objects before a major build of the application. This ensures that the most recent versions are used.

---

### Checking in the application object

Always check in the application object by itself; PowerBuilder performs unique tasks when you check in the application object.

**FOR INFO** For more information, see the discussion on working with an application object in the library chapter of the *PowerBuilder User's Guide*.

---

### ❖ To check in an object:

- 1 Select the object to check in.  
You can check in multiple objects by selecting more than one object.
- 2 Click the Check In button.  
*or*  
Select Source>Check In from menu bar.
- 3 Type comments in the Check In dialog box and click OK.  
If your version control system is configured to require check-in comments, the OK button is disabled until you type comments.
- 4 (*Optional*) If you are checking in multiple objects, click Reuse Comments to apply these comments to the remaining selected objects.  
If this box is not checked, you are prompted for comments on each object that is checked in.

What happens

When you check in an object, PowerBuilder:

- ◆ Replaces the object in the public library with the working copy
- ◆ Deletes the object from your work library
- ◆ Checks the object in to the version control system, which:
  - ◆ Increments the version level
  - ◆ Releases the lock on that object

## Deleting an object

You can delete an object from the version control provider archive with the Clear Registration function.

---

### **Use this feature cautiously**

Use caution before clearing the registration of an object. This feature deletes the evolutionary history associated with the object.

---

#### ❖ **To clear an object's registration:**

- 1 In the Library painter, select the object.
- 2 Select Source>Clear Registration from the menu bar, and click Yes to confirm.

## Enabling trace logging

A trace log records all version control activity that takes place during a Library painter session. You can either overwrite or preserve an existing log file each time you start a Library painter session.

Trace logging is disabled by default.

#### ❖ **To enable trace logging:**

- 1 In the Library painter, select Design>Options from the menu bar.
- 2 Select the Source Management tab.
- 3 Click the Log All Source Management Activity box.
- 4 (*Optional*) Specify the path to the log file.

By default, PowerBuilder creates the log file PBSMS.LOG in the folder where the current library is located.

- 5 (*Optional*) Specify whether PowerBuilder is to overwrite this file or append new information to the existing log file.

**If you select Append To Log File** (Default) PowerBuilder appends new information to the log file at the start of each Library painter session.

**If you select Overwrite Log File** PowerBuilder prompts you before overwriting this file at the start of each session.

## Configuring required check-in comments

By default, PowerBuilder does not require a check-in comment. When a check-in comment is required, the OK button on the Check In dialog box is disabled until you type a comment.

❖ **To configure required check-in comments:**

- 1 In the Library painter, select Design>Options from the menu bar.
- 2 Select the Source Management tab.
- 3 Click the Required Comments On Check In box and click OK.

## Comparing library entries

The PowerBuilder SCC API lets you compare an object with the latest version in the archive. Using this feature, you can determine what changes have been made to a library entry in your work library since it was last checked in to the source management archive.

❖ **To compare an object in the work library with the archive:**

- 1 In the Library painter, select the object in the work library.
- 2 Select Source>Compare Differences from the menu bar.

Your version control system compares the two objects and displays any variations.

Some version control systems support additional comparison functions.

FOR INFO See your version control system documentation for more information.

## Viewing object status

As you modify objects, you may need to check the status of those objects or delete an object.

Checked-out objects Before checking out an object, you can verify its status.

❖ **To view a list of checked-out objects:**

- ◆ In the Library painter, select Source>View Check Out Status from the menu bar.

Registered objects      You can view the registered objects anytime.

❖ **To view a list of registered objects:**

- ◆ In the Library painter, click the Directory button.  
*or*  
Select Source>Registration Directory from the menu bar.

## Synchronizing objects

PowerBuilder maintains information about your libraries separately from your version control system archive. An archive is updated only when you perform an operation that affects the archive such as registering, checking out, or checking in an object. Therefore, you may need to resynchronize your PowerBuilder libraries periodically with the archive.

When to synchronize      You need to synchronize your public library with your version control system archive if one of the following events desynchronizes your PowerBuilder library entries:

- ◆ Unexpected system failure
- ◆ System crash during a registration, check-in, or check-out operation
- ◆ An archive is modified outside PowerBuilder

What synchronizing does      Your version control system maintains information about your objects separately from PowerBuilder. Synchronizing the version control system with PowerBuilder replaces the objects in your public library with those in the archive. The version control system archive is updated only when you perform an operation that affects that archive, such as checking in/checking out objects.

❖ **To synchronize objects:**

- 1 Select the objects in the Library painter.
- 2 Select Source>Synchronize from the menu bar.

PowerBuilder regenerates (recompiles) the objects from their definitions in the archive and displays on the status bar the name of each object as it is synchronized.



# Using the PowerBuilder ObjectCycle Interface

## About this chapter

This chapter describes how to use the PowerBuilder ObjectCycle version control interface to manage the objects in your PowerBuilder application.

## Contents

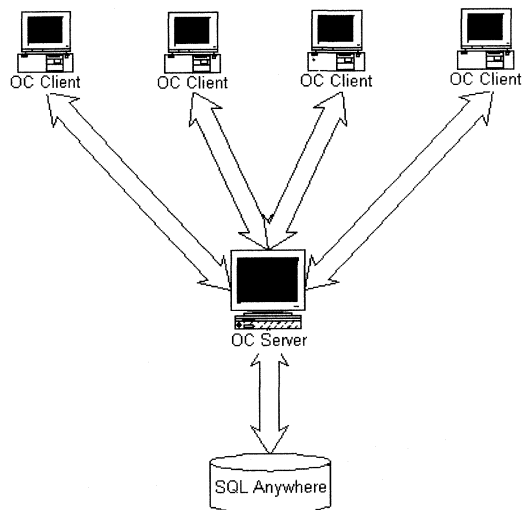
| <b>Topic</b>                                 | <b>Page</b> |
|--|-------------|
| About ObjectCycle                            | 20          |
| About the ObjectCycle interface              | 21          |
| Installing the required software             | 23          |
| Setting up projects                          | 26          |
| Connecting to ObjectCycle for the first time | 27          |
| Defining project nodes                       | 30          |
| Creating work libraries                      | 31          |
| Registering PowerBuilder objects             | 34          |
| Checking out objects from ObjectCycle        | 38          |
| Modifying checked-out objects                | 41          |
| Checking in objects to ObjectCycle           | 42          |
| Creating a new release                       | 44          |
| Using version labels                         | 47          |
| Viewing an object's version history          | 53          |
| Displaying reports                           | 54          |
| Restoring earlier versions of an object      | 57          |
| Restoring libraries                          | 60          |
| Synchronizing objects                        | 66          |

## About ObjectCycle

ObjectCycle is Powersoft's multithreaded network-based version control and configuration management service.

ObjectCycle is based on a client/server architecture:

- ◆ The ObjectCycle Server provides services to client applications that require a facility for storing and tracking the development history of PowerBuilder objects.
- ◆ Developers access the ObjectCycle Server from their own machines, which are configured as ObjectCycle clients.
- ◆ The objects are stored in a SQL Anywhere 5.0 relational database (RDBMS) via the ObjectCycle Server. The use of an RDBMS provides a degree of security and reliability not found on most version control systems.



## About the ObjectCycle interface

The PowerBuilder ObjectCycle interface lets you manage source using the ObjectCycle version control system. In the context of PowerBuilder development, your source is a collection of objects stored in PowerBuilder libraries whose versions may be managed using ObjectCycle.

**FOR INFO** For a general discussion of version control, see Chapter 1, "Preparing to Use Version Control with PowerBuilder".

### Access

You can access the ObjectCycle version control features from two PowerBuilder painters:

- ◆ **Library painter** Your principal change control activities take place here.
- ◆ **Project painter** You build the application executable here.

### Set up

After you install the ObjectCycle software and connect to the ObjectCycle Server, you must set up your environment to put your objects under ObjectCycle's control. Setup tasks include:

- ◆ Configuring your view of the archive by selecting nodes (folders)
- ◆ Creating work libraries and adding them to your application's library search path
- ◆ Registering PowerBuilder objects that you wish to place under ObjectCycle control

What you can do

After your initial set up work, the ObjectCycle interface makes it easy for you to manage your PowerBuilder objects. You can:

- ◆ Register objects and clear an object's registration
- ◆ List registered objects
- ◆ Check out and check in objects
- ◆ Display the status of checked-out objects
- ◆ Create a new release of your application
- ◆ Assign version labels to objects and filter version lists by this label
- ◆ View an object's version history
- ◆ View or print reports on objects and versions
- ◆ Restore earlier versions of objects
- ◆ Restore earlier versions of application libraries
- ◆ Synchronize objects with the ObjectCycle archive

## Installing the required software

The ObjectCycle client interface is built into PowerBuilder 6.0 Enterprise and Professional editions; but before you can use it, you must enable any Windows 3.11 client machines to use Microsoft RPC. You must install and configure the ObjectCycle Server. Windows 95 and NT (Intel) client machines are preconfigured to use Microsoft RPC services and do not need to be enabled.

Now you must do the following:

- ◆ Enable your Windows 3.11 client machines to use Microsoft RPC services so that they can connect to the ObjectCycle Server.
- ◆ Prepare for and install the Object Cycle Server (and optionally the ObjectCycle Client Utilities) from the ObjectCycle CD that is shipped with PowerBuilder 6.0. (PowerBuilder 6.0 Professional users will need to order this product from Powersoft.)
- ◆ Optionally install the ObjectCycle Manager on each client machine. The ObjectCycle Manager (a GUI interface to the ObjectCycle Server) is a PowerBuilder 6.0 application that may be used to perform general-purpose version control functions on non-PowerBuilder objects as well as administration of users, projects, folders, and objects in the ObjectCycle Server.

### ❖ To enable your Windows 3.11 client machines:

- 1 Ensure that your machine is running NetBEUI and that it is set as the default network protocol.
  - 1 Go to the Main program group and run the Windows Setup program.  
Select Options>Change Network Settings from the Windows Setup dialog box.  
The Network Setup dialog box displays. (If you do not find Change Network Settings, then you will need to install a network card.)
  - 2 Verify that Microsoft NetBEUI is listed in the Network Drivers listbox for your ethernet adapter.  
Select the ethernet adapter and click the Drivers button.  
The Network Drivers dialog box displays. (If NetBEUI is not listed, then you will need to install this network protocol; click the Add Protocol button to install it.)
  - 3 In the Network Drivers dialog box, select the Default Protocol, which is listed near the bottom.  
If the default is something other than Microsoft NetBEUI, then select NetBEUI and click the Set As Default Protocol button.

You will need to reboot for this change to take effect.

- 2 Install the Microsoft RPC software.

To do this, run the Microsoft RPC Setup program (SETUP.EXE) from the <CD>:\OCI\INTELL6\MSRPC\DISK1 directory on the ObjectCycle CD-ROM.

While installing the Microsoft RPC, it is recommended that you accept the defaults. During the installation of this software, you will be asked for your MS-DOS Runtime DLL Path (C:\DOS IS THE DEFAULT). If the directory that you choose is not currently in your path, you will need to accept the offer to update your autoexec.bat. You will need to reboot for this software to take effect.

❖ **To prepare for the ObjectCycle Server installation:**

- 1 Decide which machine in your Microsoft Network will host your ObjectCycle Server.

Select an Intel machine installed with NT Workstation (or Server) 3.51 or greater. It should have at least 16 MB of RAM (32 MB is recommended for best NT performance).

- 2 Select a disk and directory with enough disk space to register and keep several versions of your team development objects.

A disk with at least 50 MB of free space is recommended. If security is an issue, select a disk that is not shared.

❖ **To install the ObjectCycle Server software:**

- 1 Log in to the NT machine as administrator or as someone who has administrative privileges.
- 2 Locate the ObjectCycle CD and run setup.exe. It is recommended that you choose to install both the ObjectCycle Server and Client Utilities on this machine.
- 3 After installation, decide which database you want to use. The ObjectCycle Server comes with two preconfigured SQL Anywhere 5.0 database files:
  - ◆ *A regular database* (uses less disk space) named sa50oc.db that is accessed by the ODBC profile name ObjectCycle SQLA5. For this database, it is recommended that you have at least 25 MB of free disk space available for your project to grow. This is the default.

- ◆ A *transaction log database* (twice as fast but using more than twice the disk space) named sa50oct.db that is accessed by the ODBC profile name ObjectCycle SQLA5T. For this database, it is recommended that you have at least 50 MB of free disk space available for your project to grow.

To use this database, double-click the ObjectCycle server Startup Settings in your program group or choose this option from the ObjectCycle 1.0 folder on the Start menu and change the Name=ObjectCycle SQLA5 entry to **Name=ObjectCycleSQLA5T**. This causes the ObjectCycle Server to look for a different ODBC data source when it starts up.

- 4 Use the Create ObjectCycle Server NT Service program in the ObjectCycle folder to install the ObjectCycle server as an NT service. This will cause the ObjectCycle Server to startup each time you reboot the machine.

FOR INFO For more information, see the ObjectCycle Server Help.

- 5 Start the ObjectCycle Server by running the Object Cycle server program in the ObjectCycle folder. Check the ObjectCycle Server Log File after 30 seconds or so to verify that it is listening to clients.

FOR INFO For more information, see the ObjectCycle Server Help.

- 6 Use the ObjectCycle Manager to create projects, folders, and user names for PowerBuilder 6.0 users.

FOR INFO For more information on creating projects, folders, and user names, and on maintaining your ObjectCycle Server, see the ObjectCycle Manager Help.

❖ **To install the ObjectCycle Manager:**

- 1 From the PowerBuilder 6.0 Enterprise CD, choose the ObjectCycle Manager component.
- 2 From the ObjectCycle CD, run setup.exe and install the ObjectCycle Client Utilities.

## Setting up projects

After installing the ObjectCycle Server, you need to create user names, password(s), project(s), and node(s) for your development team.

❖ **To set up projects and user access:**

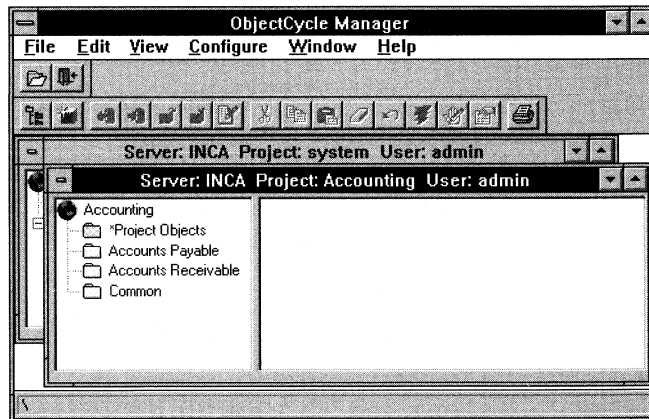
- 1 Double-click the ObjectCycle Server icon in the ObjectCycle program group to start the ObjectCycle Server.

The ObjectCycle Server has a preconfigured administrative user and default project. The user name is *admin*, the password is *camus*, and the project is *system*.

- 2 Use the ObjectCycle Manager to log on to the ObjectCycle Server and set up the user name(s), optional password(s), project(s), and nodes that will be used to store PowerBuilder objects.

**FOR INFO** For details on setting up projects and users, see the "Administrator setup" section in the ObjectCycle Manager Help.

Here is an example of a project set up from ObjectCycle Manager:



The project is an accounting project with three nodes (folders) to contain the project objects: Accounts Receivable, Accounts Payable, and Common.

- 3 Give PowerBuilder users the user name, optional password, project name, and ObjectCycle Server name to be used when they connect to ObjectCycle. They will also need to know the project node list(s) to use when they connect to ObjectCycle, as described in "Defining project nodes" on page 30.



## Connecting to ObjectCycle for the first time

The first time you open the Library painter in PowerBuilder, you must specify that your source control system is ObjectCycle.

❖ **To connect to ObjectCycle in PowerBuilder for the first time:**

- 1 Open your application in the Application painter.

Your application is now the current application.

**FOR INFO** For instructions on using the Application painter, see the *PowerBuilder User's Guide*.

- 2 In the Library painter, select Source>Connect from the menu bar.

The Connect dialog box displays.

- 3 Select ObjectCycle from the Vendors dropdown listbox. Click OK.

The ObjectCycle Connect dialog displays:

- 4 Type the values supplied by your ObjectCycle administrator for ObjectCycle User ID, ObjectCycle Password, Project, and ObjectCycle Server name.
- 5 Click OK.

PowerBuilder connects to ObjectCycle. Since this is the first time you are connecting to ObjectCycle, you see a message box prompting you to create a configuration file for this application.

**FOR INFO** For instructions on creating the configuration file, see "Creating and saving the configuration file" next.

**If PowerBuilder cannot connect**

If PowerBuilder is unable to connect to the ObjectCycle Server, make sure the ObjectCycle Server is running and the network is up, and then try to reconnect.

**FOR INFO** If you are still having problems, see "Installing the required software" on page 23.

## Creating and saving the configuration file

Before you can start checking out and checking in PowerBuilder objects with the ObjectCycle interface, you must set up an ObjectCycle configuration file for your PowerBuilder application. Configuration files are associated with each application whose objects you have put under ObjectCycle's control.

What the configuration file does

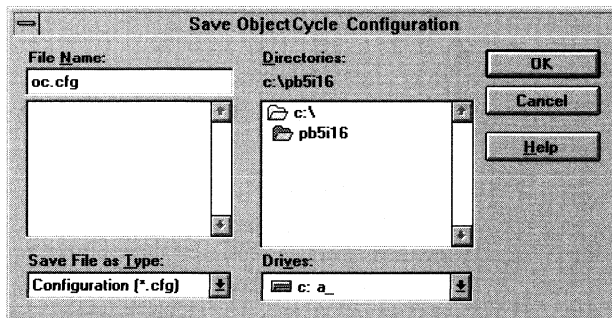
The ObjectCycle configuration file specifies the connection to the ObjectCycle server on which your PowerBuilder objects are stored. Connection information includes the ObjectCycle user ID, ObjectCycle password, ObjectCycle project, and ObjectCycle Server's machine name. The configuration filename must have a CFG extension. For example, here is the contents of a sample ObjectCycle configuration file:

```
[pb]
USERID=carol
PASSWORD=hiThair
PROJECT=system
SERVER=ILLUSION
NODELIST=
```

### ❖ To create and save an ObjectCycle configuration file for the first time:

- 1 Click Yes in the message box that prompts you to create the configuration file (as in the preceding section).

The Save ObjectCycle Configuration dialog box displays:



- 2 Select a directory in the Directories box, browsing other local or networked drives if necessary.
- 3 In the File Name box, type a configuration filename with the CFG extension.

If you omit the CFG extension, PowerBuilder automatically adds it to the configuration filename.

- 4 Click OK.

PowerBuilder creates the configuration file and saves it in the specified directory.

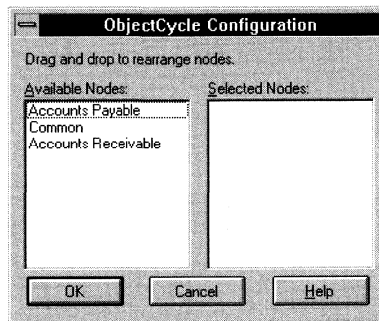
## Defining project nodes

After connecting to ObjectCycle, you need to select the nodes (folders) you will use in your project. The node list is created for you by an ObjectCycle user with administrator status, as explained in "Setting up projects" on page 26.

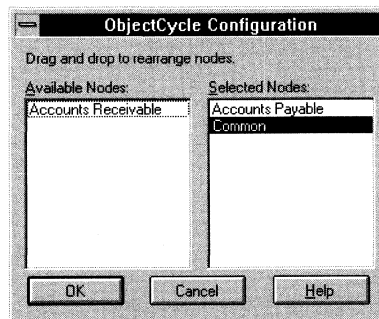
❖ **To select project nodes:**

- 1 In the Library painter, select Source>Configuration from the menu bar.

The ObjectCycle Configuration dialog displays. The Available nodes (folders) for the objects you will check in and out display in the left column:



- 2 Drag and drop the Available nodes you intend to use to the right column (Selected Nodes).



## Creating work libraries

When you use the ObjectCycle interface to check out PowerBuilder objects for modification, you put the objects in a **work library**, a private library where you keep working copies of objects.

Your archive directories and public libraries often reside on a network server, accessible to everyone in your development group. The work library typically resides on your own computer and is placed first in the Library Search path.

You create the work library in the Library painter and adjust the Library Search path in the Application painter.

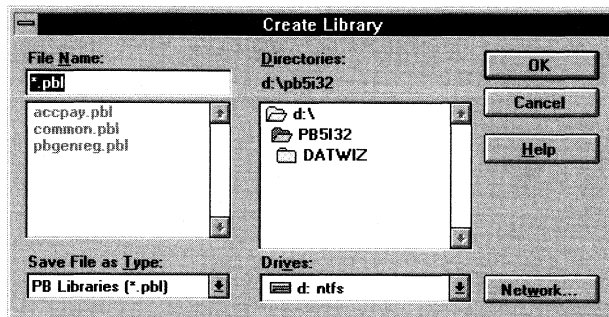
❖ **To create a work library:**

- 1 In the Library painter, click the Create Library button.

*or*

Select Library>Create from the menu bar.

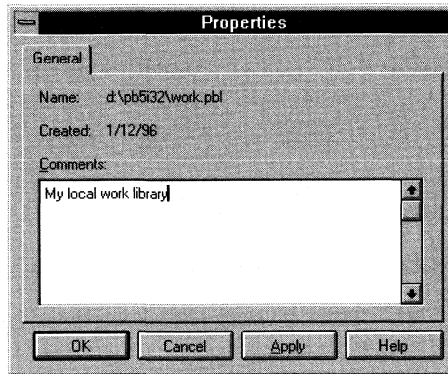
The Create Library dialog box displays:



- 2 Select a directory from the Directories box where you want to locate the work library.

- 3 Specify a filename for the work library with the PBL extension and click OK.

The Properties dialog box displays:



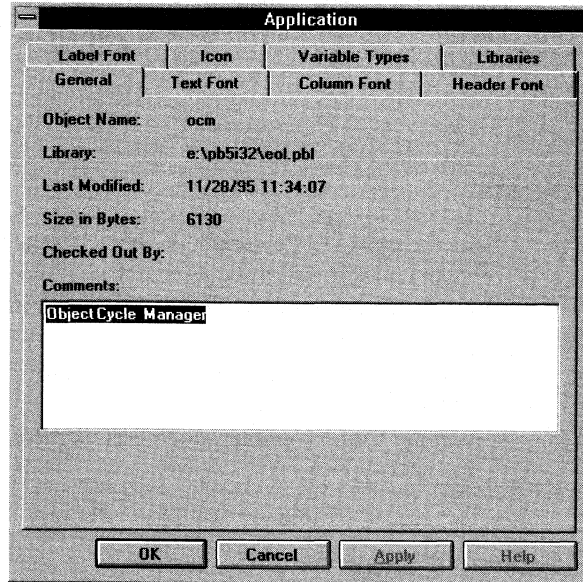
- 4 Type a comment describing the purpose of your work library and click OK.

PowerBuilder creates the work library in the directory you specified.

❖ **To add the work library to the library search path:**

- 1 In the Application painter, select Entry>Properties from the menu bar.

The Application dialog box displays:



- 2 Select the Libraries tab and click Browse.
- 3 Locate the Library, add it to the Library Search Path, and click OK.

## Registering PowerBuilder objects

After you connect to ObjectCycle, you can register the objects in your PowerBuilder application. **Registering** an object places it under ObjectCycle's control. When you register objects, the ObjectCycle interface creates a separate ObjectCycle object on the ObjectCycle server for each PowerBuilder object you specify.

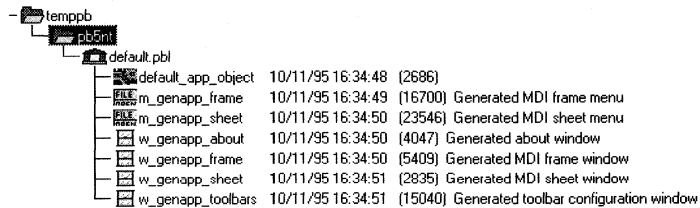
### Registering objects

You can register a PowerBuilder object anytime. After you register an object you can check it out of the public library into your work library, modify the object as needed, and check the object back into the public library for safekeeping.

❖ **To register objects:**

- 1 In the Library painter, double-click the public library that contains the objects you want to register.

A list of the objects displays in the Library painter window:



- 2 Select one or more objects that you want to register.

---

#### Selecting a group of objects

If you select a group of objects to register, a separate ObjectCycle Archive Registration dialog box displays for *each* object you select.

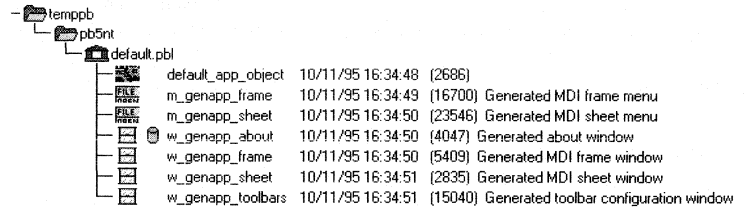
---

- 3 Select Source>Register from the menu bar.
- 4 Select the Target Node and Starting Revision.



## 5 (Optional) Type any comments you want.

PowerBuilder registers the objects you select into the project (archive) node specified and assigns a registration icon to each object in the Library painter:



## Listing registered objects

You can view a list of registered objects based on your current configuration at any time.

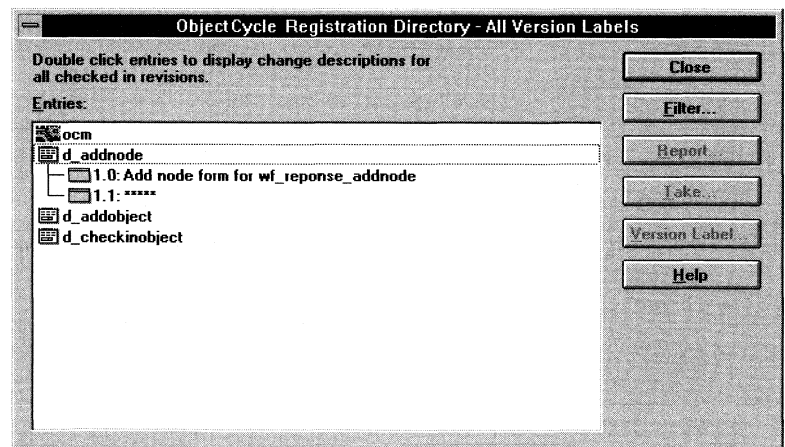
### ❖ To view a list of registered objects:

- ◆ In the Library painter, click the Directory button.

or

Select Source>Registration Directory from the menu bar.

The ObjectCycle Registration Directory dialog box displays listing all registered objects in your current application. *All Version Labels* displays in the title bar to indicate that all objects are displayed even if they have different version labels assigned:



---

### Viewing an object's version history

You can double-click any registered object listed in the ObjectCycle Registration Directory dialog box to see that object's version history.

FOR INFO For how, see "Viewing an object's version history" on page 53.

---

## Clearing an object's registration

You can remove an object from ObjectCycle's control by clearing its registration (the object cannot be currently checked out).

❖ **To clear an object's registration:**

- 1 In the Library painter, select one or more objects to clear.

---

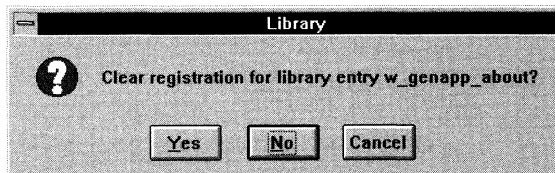
### Selecting a group of objects

If you select a group of objects, a message box displays for *each* object you select.

---

- 2 Select Source>Clear Registration from the menu bar.

A message box similar to the following displays:



- 3 Click Yes to clear the object's registration.

### What happens

PowerBuilder does the following for each object:

- ◆ Clears the object's registration
- ◆ Automatically deletes the object from the ObjectCycle Server (archive or project)
- ◆ Removes the registration icon for the object in the Library painter

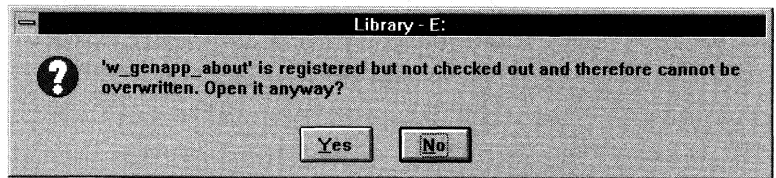
## Opening read-only versions of registered objects

If you want to view a registered object *without* making changes to it, you can open a read-only version of the object without checking it out.

❖ **To open a read-only version of a registered object:**

- 1 In the Library painter, double-click the registered object you want to open.

A message box similar to the following displays:



- 2 Click Yes to open a read-only version of the registered object.

The associated PowerBuilder painter opens and displays the selected object. You will be able to view the object but you will *not* be able to save changes to it.

## Checking out objects from ObjectCycle

After you register an object with the ObjectCycle interface (see "Registering objects" on page 34), you must check it out of the public library in order to modify it. When you check out an object, PowerBuilder locks this version of the object so no one else can check it out while you are working on it.

Before you check out objects

Typically, you check out objects to work libraries. Before you check out objects, make sure you have:

- ◆ Created the work library (see "Creating work libraries" on page 31)
- ◆ Registered the objects you want to check out (see "Registering objects" on page 34)

## Checking out objects

You can check out a registered object at any time by selecting it in the Library painter.

### ❖ To check out registered objects:

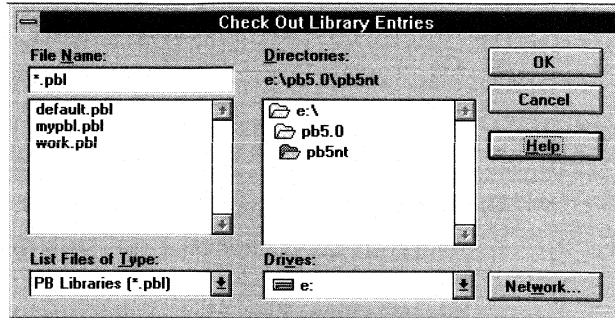
- 1 In the Library painter, select one or more registered objects to check out.
- 2 Check out the object or objects you want.

**To check out a group of objects** Click the Check Out button on the PainterBar *or* select Source>Check Out from the menu bar.

**To check out a single object** Right-click the object and select Check Out from the popup menu.

- 3 Type your User ID (the one you entered when you connected) and click OK.

The Check Out Library Entries dialog box displays showing the current directory and libraries (PBLs) in that directory:

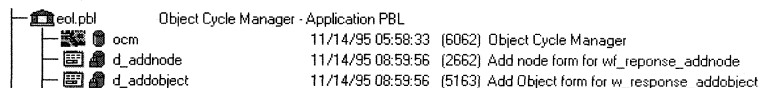


- 4 In the Directories box, select the directory where your work library resides.
- 5 Type the name of the work library (with a PBL extension) in the File Name box.  
*or*  
Select the name of the work library from the listbox.
- 6 Click OK to check out the objects to the work library.

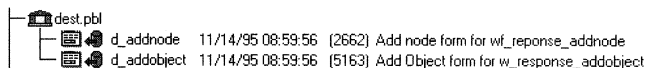
#### What happens

PowerBuilder does the following:

- ◆ Makes a working copy of each selected object and stores it in the work library you specified
- ◆ Locks this version of the object so no one else can check it out
- ◆ In the public library, assigns a lock icon to the object showing that the object is locked:



- ◆ In the work library, assigns a check-out icon to the object indicating that it is the working copy:



## Viewing the status of checked-out objects

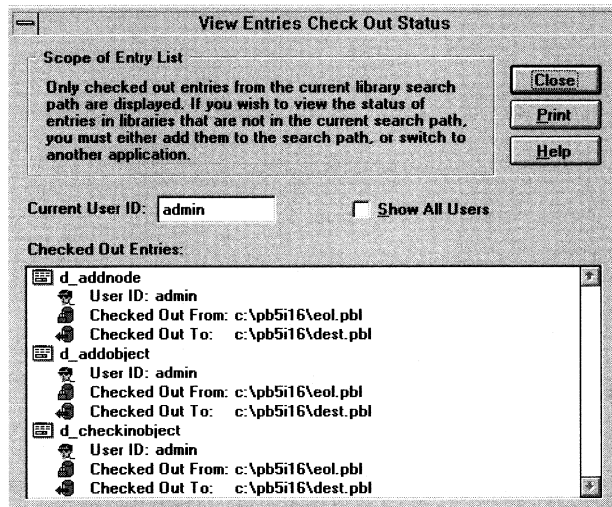
You can view the status of objects checked out from libraries in the search path of the current application anytime. The status information displayed for each checked-out object includes:

- ◆ The ID of the user who checked out the object
- ◆ The location of the public library from which the object was checked out
- ◆ The location of the work library to which the object was checked out

### ❖ To view the status of checked-out objects:

- 1 In the Library painter, click the Check Status button  
*or*  
Select Source>View Check Out Status from the menu bar.

The View Entries Check Out Status dialog box displays showing the status of objects you checked out:



- 2 To see the status of objects checked out by *all* users, select the Show All Users checkbox.
- 3 To print the status, click the Print button.  
The printed version contains the contents of the Checked Out Entries listbox.
- 4 Click Close to close the dialog box.

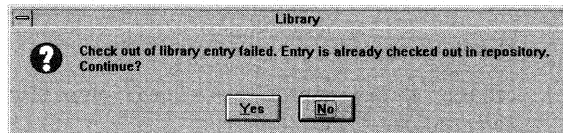
## Modifying checked-out objects

When you need to modify an object, you should check out the object to your work library and make changes only to the checked-out version. This ensures that no one else can modify the same object while you are working on it.

---

### You cannot overwrite objects checked out by someone else

If you try to open an object checked out by someone else, a message box similar to the following displays:



Click Yes if you want to open the object. You will be able to view the object, but you will *not* be able to save changes to it.

---

### ❖ To open a checked-out object in order to modify it:

- ◆ In the Library painter, double-click the checked-out object in your work library that you want to modify.

The associated PowerBuilder painter opens and displays the selected object.

## Checking in objects to ObjectCycle

When using the ObjectCycle interface, you should check in objects as follows:

- ◆ **Your objects** Check your objects into the public library *as soon as you finish modifying them*. This ensures that all developers will have access to the updated objects.
  - ◆ **All objects** Check all objects into the public library *before a major build of the application executable*. This ensures that PowerBuilder uses the most recent versions of the objects to build the new executable.
- ❖ **To check in objects:**

- 1 In the Library painter, select one or more registered objects that you want to check in.

---

### Selecting a group of objects

If you select a group of objects to check in, a separate ObjectCycle Check In dialog box displays for *each* object you select.

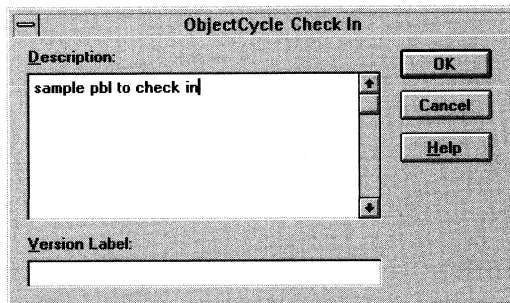
---

- 2 Check in the object or objects you want.

**To check in a group of objects** Click the Check In button *or* select Source>Check In from the menu bar.

**To check in a single object** Right-click the object and select Check In from the popup menu.

The ObjectCycle Check In dialog box displays:





- 3 Type a description in the Description box.

The description will appear in the registration directory listing for this object when you display its change history. It will also appear in the version report. For examples, see "Viewing an object's version history" on page 53.

- 4 (Optional) Type a version label in the Version Label box.

This associates the version label with the new version in the archive.

FOR INFO For version label considerations see "Using version labels" on page 47.

- 5 Click OK to check in the objects.

#### What happens

PowerBuilder does the following:

- ◆ Copies the object in the work library to the public library
- ◆ Stores a revised copy of the object in the ObjectCycle database
- ◆ Deletes the working copy of the object from your work library
- ◆ Increments the object's version number. For example, if the version number of the object is 1.0 when you check it out, it becomes 1.1 when you check the object back in
- ◆ Unlocks the object so other developers can check it out (PowerBuilder removes the lock icon for this object in the Library painter)

## Creating a new release

You can create a new release at any point in the development of your PowerBuilder application.

### When to do this

Typically you create a new release when you reach a development milestone and want to copy your libraries and objects to a separate place while preserving your version history.

### Your choices

PowerBuilder gives you some choices when you build a new release. You can:

- ◆ Set options for the new archive
- ◆ Specify a starting version number for the release

## About setting options for the new archive

When you create a new archive, you can start with a fresh object or you can preserve the existing version history. If you start with a fresh object, the new archive consists of only the most current version level. If you maintain the existing versions, the new archive consists of the entire version history.

## About specifying a starting version number

You can also specify a starting version number for the new archive if you want. When you create a new release, PowerBuilder will maintain the most current version number of each object unless you specify a new starting version number.

For example, if you specify 2.0 as the starting version number, your new release will have the most current versions of the objects begin with version number 2.0. ObjectCycle will then assign version numbers beginning with 2.0, such as 2.1, 2.2, and so on.

## Creating the release

You create a new release of your application by completing the ObjectCycle New Release dialog box.

**Before you start**

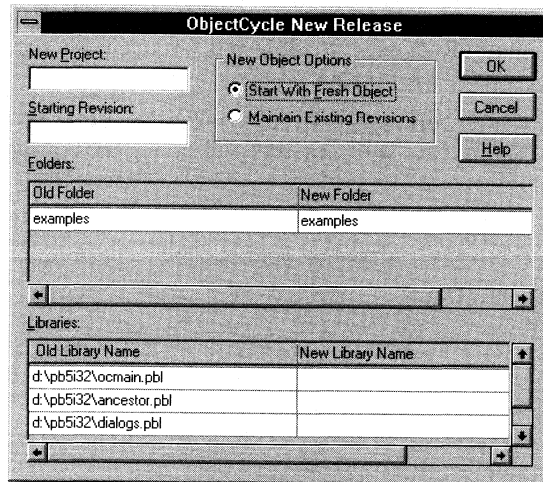
Make sure that all objects have been checked in to ensure that you start with an accurate version history.

FOR INFO For instructions on checking in objects, see "Checking in objects to ObjectCycle" on page 42.

❖ **To create the new release:**

- 1 In the Library painter, select Source>Create New Release from the menu bar.

The ObjectCycle New Release dialog box displays:



- 2 Type a new project name, such as **ProjectVersion2.0**. You will use this name to create a new project in the ObjectCycle Server.
- 3 Click one of the radio buttons in the New Object Options groupbox:

**Start With Fresh Object** (Default) Copies only the most current version level to the new project

**Maintain Existing Versions** Copies the entire version history in its current state to the new project

- 4 (*Optional*) Specify a starting version number in the Starting Revision box.
- 5 Type a new project name.

- 6 Type a new library name for each old library name listed.

If you specify the names of existing libraries, PowerBuilder prompts you before overwriting them.

- 7 Click OK.

PowerBuilder creates the new release in the project and libraries you specified.

After you create the new release

To make changes to the objects in your new release, you must:

- ◆ Make the new application the current application by opening it in the Application painter. (For how, see the *PowerBuilder User's Guide*.)
- ◆ Add the new libraries to the new application's library search path. (For how, see the *PowerBuilder User's Guide*.)
- ◆ Reconnect to ObjectCycle (using Source>Connect) and specify the new project name
- ◆ Define an ObjectCycle configuration file for the new application. (For how, see "Creating and saving the configuration file" on page 28.)

## Using version labels

A **version label** is a descriptive name that identifies versions in an archive and that stays associated with its version even after you check in new versions.

### Why use

You can use version labels to group the versions that comprise a particular version of an application so you can select them quickly and easily. For example, you might assign the version label *Release\_1* to all objects belonging to this release.

When you use the Project painter to build a project, the ObjectCycle interface lets you assign version labels as part of the build process (see "Assigning version labels to a group of objects" next). In this way, you can use version labels as an alternative to creating a new release by copying libraries (as described in "Creating a new release" on page 44).

### Rules

The following rules apply when you use version labels with the ObjectCycle interface:

| When you do this                                      | Follow these rules  |
|---|---|
| Assign version labels to objects                      | <p>You <i>can</i> assign the same version label to any number of objects stored in the archive</p> <p>You <i>can</i> assign multiple version labels to a single version of an object</p> <p>You <i>cannot</i> assign the same version label to more than one version of an object in the same archive</p>   |
| Specify a version label for the ObjectCycle interface | <p>The version label must be a contiguous string of letters, digits, or underscores ( <code>_</code> ) with no spaces or periods</p> <p>The first character of the version label must be a letter or underscore ( <code>_</code> )</p> <p>The version label is case sensitive. For example, <i>Release_1</i> and <i>release_1</i> are considered different version labels</p> |

## Assigning version labels to a group of objects

Typically you assign a version label to a group of objects.

❖ **To assign a version label to a group of objects:**

- 1 In the Library painter, click the Directory button.

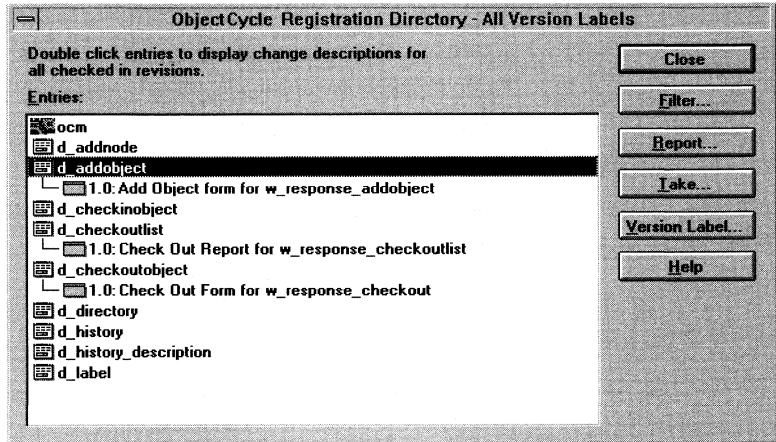
*or*

Select Source>Registration Directory from the menu bar.

The ObjectCycle Registration Directory dialog box displays.

- 2 Double-click the objects for which you want to display versions.

The version history displays for each selected object:



- 3 Select the versions to which you want to assign the same version label.

Press CTRL+click to select noncontiguous versions.

- 4 Click the Version Label button.

The ObjectCycle Assign Version Label dialog box displays.

- 5 Type a version label in the Version Label box and click OK.

FOR INFO For version label rules see "Rules" on page 47.

PowerBuilder assigns the version label to the selected versions.

You can check the results of the version label assignment in either of the following ways:

- ◆ Display the archive report for any of the objects containing versions with the version label you specified. The archive report includes the version label.

FOR INFO See "Displaying an archive report" on page 54.

- ◆ Filter the list of versions using the version label in the Registration Directory dialog box.

FOR INFO See "Filtering version lists by version labels" on page 49.

## **Filtering version lists by version labels**

You can filter the list of versions in the Registration Directory dialog box by a particular version label. This lets you display the change history for only those objects containing versions with this version label.

❖ **To filter a list of versions by version label:**

- 1 In the Library painter, click the Directory button.

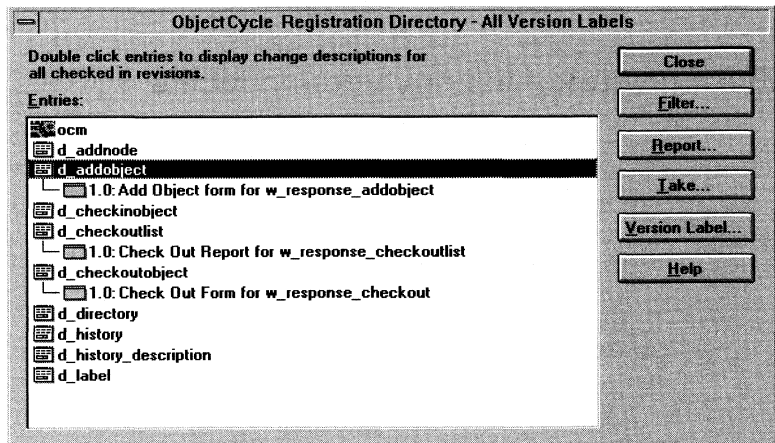
*or*

Select Source>Registration Directory from the menu bar.

The ObjectCycle Registration Directory dialog box displays listing all the registered objects in your current application.

Initially *All Version Labels* displays in the title bar of the dialog box.

This indicates that all objects are displayed even if they have different version labels assigned. After you filter the list, the title bar changes to show the version label you are filtering by:



- 2 Click the Filter button.

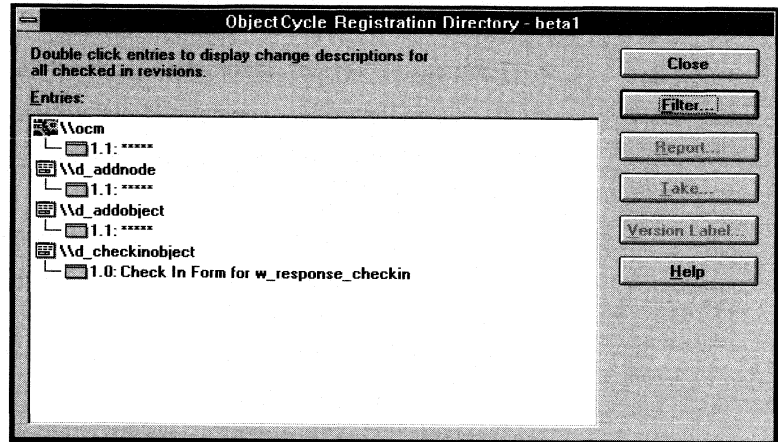
The ObjectCycle Filter On Version Label dialog box displays.

- 3 Type the version label you want to filter by in the Version Label box.



- 4 Click OK.

The Registration Directory dialog box displays only those objects containing versions with the specified version label. Notice that *beta1* displays in the title bar to indicate the version label you are filtering by:



## Clearing the filter

When you close the Registration Directory dialog box, PowerBuilder clears the filter automatically so that the next time you open the Registration Directory dialog box, all objects display. However, you may want to clear the filter *without* closing the Registration Directory dialog box.

### ❖ To clear the filter without closing the Registration directory dialog box:

- 1 Click the Filter button in the Registration Directory dialog box.

The ObjectCycle Filter On Version Label dialog box displays with the version label filter you specified.

- 2 Do either of the following:

- ◆ Click the Clear button.
- ◆ Select the version label text (if it is not already selected), press the DELETE key to clear it, then click OK.

The Filter On Version Label dialog box closes and all objects display in the Registration Directory dialog box.

## Assigning version labels when you build a project

When you use the Project painter to build a project that includes objects under ObjectCycle's control, you can specify a version label for all of the objects in the project. This is useful if you want to assign a version label as part of the build process.

❖ **To assign a version label when you build a project:**

- 1 Open the project you defined in the Project painter.

FOR INFO For instructions on using the Project painter, see the *PowerBuilder User's Guide*.

- 2 Click the Build button in the Project painter bar.

After building the executable, the ObjectCycle Assign Version Label dialog box displays.

- 3 Type the version label in the Version Label box, and click OK.

PowerBuilder builds the executable and assigns the label to each registered object that was used to build the executable. This information is stored in the project object.

FOR INFO For version label rules see "Using version labels" on page 47.

## Viewing an object's version history

You can view a registered object's version history at any time by displaying the ObjectCycle Registration Directory dialog box. A **version** is a saved *version* of an object under ObjectCycle's control.

Each time you check an object into its archive, it is saved as a new version and ObjectCycle increments the version number. For example, if you check out an object with a version number of 1.0, its version number becomes 1.1 when you check it back in.

### ❖ To view an object's version history:

- 1 In the Library painter, click the Directory button.

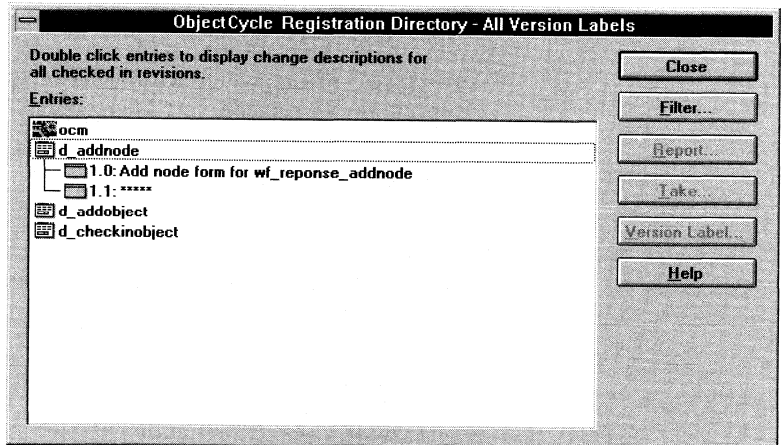
*or*

Select Source>Registration Directory from the menu bar.

The ObjectCycle Registration Directory dialog box displays listing all the registered objects in your current application.

- 2 Double-click any object to view its version history.

All versions for the selected object display in the dialog box:



## Displaying reports

You can display two types of reports using the ObjectCycle interface. These reports give you more information about the archive and versions in your application.

- ◆ **Archive report** Describes the entire version history of an object's archive since its registration.
- ◆ **Version report** Describes one specific version of an object.

## Displaying an archive report

You can display an object's archive report by selecting the object in the Library painter or in the ObjectCycle Registration Directory dialog box. You can view the report, print it, or copy it to a file.

### ❖ To display an archive report from the Library painter:

- 1 Right-click the name of the registered object for which you want a report and select Registration Report from the popup menu.

The ObjectCycle Archive Report dialog box displays.

**FOR INFO** For more about the types of information in the archive report, see the ObjectCycle Manager Help.

- 2 Click the Print button if you want to print the archive report on your default printer.
- 3 Click OK to close the dialog box.

### ❖ To display an archive report from the Registration Directory dialog:

- 1 In the Library painter, click the Directory button.  
*or*  
Select Source>Registration Directory from the menu bar.

The ObjectCycle Registration Directory dialog box displays listing all the registered objects in your current application.

- 2 Select the name of an object.
- 3 Click the Report button.

The ObjectCycle Archive Report dialog box displays.

**FOR INFO** For more about the types of information in the archive report, see the ObjectCycle Manager Help.

- 4 Click the Print button if you want to print the archive report on your default printer.
- 5 Click OK to close the dialog box.

## Displaying a revision report

You can display a revision report at any time by selecting the version in the ObjectCycle Registration Directory dialog box. You can view the report, print it, or copy it to a file.

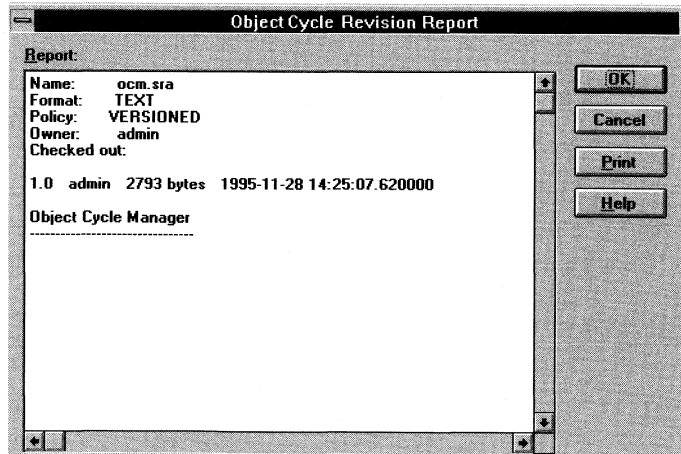
### ❖ To display a version report:

- 1 In the Library painter, click the Directory button.  
*or*  
Select Source>Registration Directory from the menu bar.

The ObjectCycle Registration Directory dialog box displays listing all the registered objects in your current application.

- 2 Double-click an object to view a list of its versions.
- 3 Select the version for which you want a report.
- 4 Click the Report button.

The ObjectCycle Revision Report dialog box displays:



**FOR INFO** For more about the types of information in the revision report, see the ObjectCycle Manager Help.

- 5 Click the Print button if you want to print the archive report on your default printer.
- 6 Click OK to close the dialog box.

## Copying a report to a file

You can copy the text of an archive or revision report to a file.

❖ **To copy the text of an archive or revision report to a file:**

- 1 In the Archive Report or Revision Report dialog box, select the text you want to copy from the Report box.
- 2 Press CTRL+C to copy the selected text to the clipboard.
- 3 Open a file in a text editor of your choice.
- 4 Press CTRL+V to paste the text from the clipboard into the file.
- 5 (*Optional*) Save the file for future reference.

## Restoring earlier versions of an object

One of the main reasons for putting your PowerBuilder objects under version control is to enable you to restore earlier versions of an object. The ObjectCycle interface makes this possible.

### Deciding which version to restore

The first step in restoring an earlier version of an object is to decide which version you want to restore. There are several items you can display in the ObjectCycle interface to help you decide which version you want.

#### Use descriptive comments

These methods work best when you carefully comment each version as you check it back into the archive.

| To see this  | Display this  | For how, see                                     |
|--|---|--|
| A list of all versions for this object                                   | The object's version history in the ObjectCycle Registration Directory dialog box | "Viewing an object's version history" on page 53 |
| The entire version history of an object's archive since its registration | An archive report for an object   | "Displaying an archive report" on page 54        |
| Information about one version of an object                               | A version report for a specific version   | "Displaying a revision report" on page 55        |

### Restoring an earlier version

Once you decide which version of an object you want to restore, you can perform the restoration from the ObjectCycle Registration Directory dialog box.

❖ **To restore an earlier version of an object:**

- 1 Check out the current version of the object you want to restore to your work library.

FOR INFO For instructions, see "Checking out objects from ObjectCycle" on page 38.

- 2 In the Library painter, click the Directory button.

*or*

Select Source>Registration Directory from the menu bar.

The ObjectCycle Registration Directory dialog box displays listing all the registered objects in your current application.

- 3 Double-click an object to view a list of its versions.

- 4 Select the version that you want to restore and click the Take button.

The Select Destination Library dialog box displays.

- 5 Select a destination library in the Libraries box. (Typically, this is your work library.)

- 6 Click OK.

PowerBuilder copies the selected version of the object over the current version.

- 7 Check the restored version back into the archive.

FOR INFO For instructions, see "Checking in objects to ObjectCycle" on page 42.

The restored version becomes the current version of the object.

## Restoring versions by version labels

To restore only those versions having a particular version label, you filter the list of objects by the version label and then select the objects you want from the resulting list.

❖ **To restore versions by version label:**

- 1 In the ObjectCycle Registration Directory dialog box, click the Filter button to filter by the version label you want.

FOR INFO For instructions, see "Filtering version lists by version labels" on page 49.



- 2 Select the version you want from each object and click the Take button.  
The Select Destination Library dialog box displays.
- 3 Select a destination library in the Libraries box and click OK.  
PowerBuilder puts the selected versions into the library you specified.

## Restoring libraries

When you use the ObjectCycle interface, you can restore earlier versions of PowerBuilder libraries (PBLs).

What PowerBuilder does

When you restore libraries, PowerBuilder puts your objects into specified libraries. After you restore your libraries to their new location, you can rebuild the application executable in the Project painter.

Project object

Restoring libraries is possible because of the **project object**. Each time you build an executable in the Project painter, PowerBuilder saves information about the application in the project object. You can use this information later to restore your libraries. PowerBuilder stores the following information about objects in each project object:

- ◆ PowerBuilder object names with PBL name
- ◆ Archive names and version numbers
- ◆ Version labels

Each time you build an executable, PowerBuilder updates the project object with the most recent version of this information.

## Listing the objects in a project

After you build your project in the Project painter, you can display a list of the objects in the project.

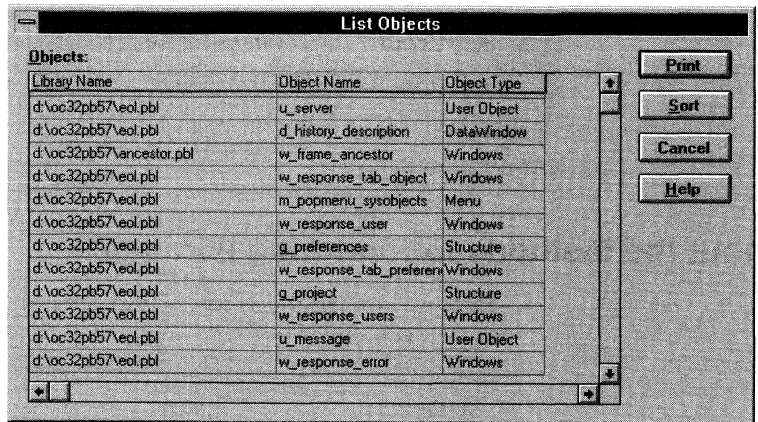
❖ **To list the objects in a project:**

- 1 Build your project in the Project painter.

FOR INFO For instructions on using the Project painter, see the *PowerBuilder User's Guide*.

- 2 Select Design>List Objects from the menu bar.

The List Objects dialog box appears displaying a report of the objects in your project and how they map to objects in your archive:



- 3 (Optional) Click the Print button to send the report to your default printer, or click the Sort button to specify the order in which the objects are listed.
- 4 Click Cancel to close the dialog box.

What's in the report

The report in the List Objects dialog box includes the following columns:

| This column    | Shows this information  |
|----------------|---|
| Library Name   | Source library containing the object  |
| Object Name    | Name of the object  |
| Object Type    | Type of object  |
| Version Number | Unique number (such as 1.0 or 1.1) that identifies the version in the archive |
| Version Label  | Descriptive name that identifies the version in the archive                   |

What you can do                      You can do the following when viewing the List Objects report:

- ◆ Scroll horizontally to see the columns
- ◆ Resize the columns
- ◆ Reorder the columns (by selecting and dragging)
- ◆ Print the report
- ◆ Sort the rows in the report

## About the methods for restoring libraries

Because PowerBuilder stores information about the objects in the project object, you can restore your libraries by using either of the following methods:

| Method   | What you do   | Uses ObjectCycle? |
|--|---|-------------------|
| Retrieve an earlier version of the project object from ObjectCycle | Use ObjectCycle to track changes to the project object  | Yes               |
| Save the project object with a new name                            | Each time you build an executable or change the project object definition, select File>Save As from the Project painter menu bar to save the object to a new name in current library list | No                |

## Retrieving the project object from ObjectCycle

When to use                      Use this method if:

- ◆ You need to be able to restore any version level of your project object at any time
- ◆ You expect to have many version levels of your project object

Benefits                              Putting your project object under ObjectCycle's control is very secure. In addition, this method creates only a single project object in your library listing since ObjectCycle takes care of tracking changes to it.

Strategy                              Use descriptive comments and version labels when handling project objects under ObjectCycle's control to help you determine which project object you want to retrieve.

## Saving the project object with a new name

|                   |   |
|-------------------|---|
| When to use       | Use this method if you plan to save a small number of project objects that you may want to restore from.  |
| Benefit           | Saving a project object with a new name is a very simple method to use. To open an earlier version of a project object, you simply double-click it in the Library painter listing. In this way, you have access to the project objects you create each time you build an executable or change the project object definition.  |
| Possible drawback | However, for large, complex development projects with many intermittent builds, this method may not meet your needs. Using this method could create more project objects in your library than you can easily manage.  |
| Strategies        | <p><b>Create a separate library for your project objects</b> If you use this method, you might want to create a separate library to hold project objects that you create each time you build an application executable or change the definition of a project object. This makes it easier to find a project object when you want to open it and restore libraries.</p> <p><b>Use descriptive comments</b> Be sure to provide descriptive comments for your project objects so you can easily identify them.</p> |

## Retrieving the project object from ObjectCycle

The project object stores information about the objects, version numbers, and version labels in your application. Therefore, when you put the project object under ObjectCycle's control, you can use its version history to restore libraries used to build earlier versions of your application.

- ❖ **To restore libraries by retrieving the project object from ObjectCycle:**
  - 1 Create and save the initial version of the project object in the Project painter.

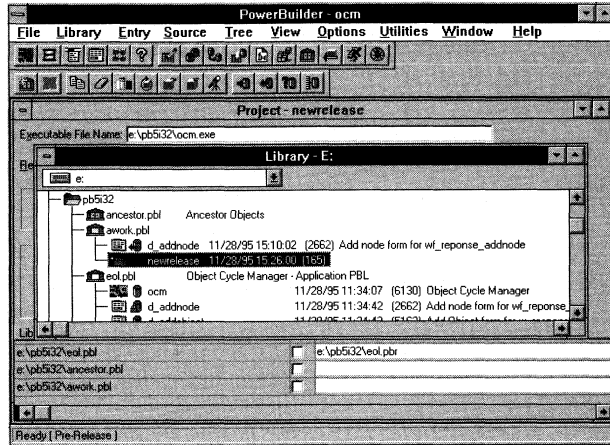
FOR INFO For instructions on using the Project painter, see the *PowerBuilder User's Guide*.
  - 2 Register the project object with ObjectCycle.

FOR INFO For instructions, see "Registering PowerBuilder objects" on page 34.

- 3 Click the Take button in the ObjectCycle Registration Directory dialog box to retrieve an earlier version level of the project object.

FOR INFO For instructions, see "Restoring earlier versions of an object" on page 57.

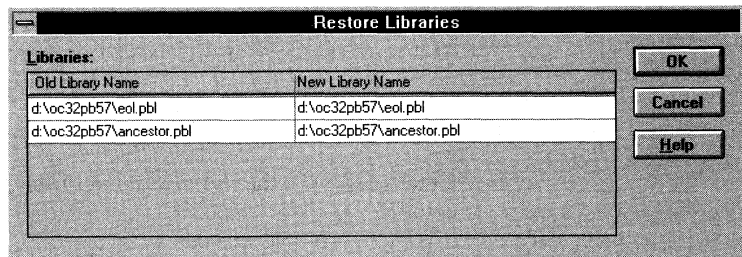
- 4 In the Library painter, double-click the project object you retrieved.



The Project painter opens and displays the selected project object definition.

- 5 Select Design>Restore Libraries from the Project painter menu bar.

The Restore Libraries dialog box displays:



- 6 Type a new library (PBL) name for each old library name listed.

If you specify the names of existing libraries, PowerBuilder prompts you before overwriting them.

- 7 Click OK.

PowerBuilder restores the libraries to the state they were in when you built the application executable.

## Saving the project object with a new name

If you decide *not* to put your project objects under ObjectCycle's control, you can save the project object with a new name each time you build an executable or change the project object definition. You then open an earlier version of the project object to restore your libraries.

❖ **To restore libraries by saving the project object with a new name:**

- 1 Build your project in the Project painter.  
FOR INFO For instructions on using the Project painter, see the *PowerBuilder User's Guide*.
- 2 Select File>Save As from the Project painter menu bar.  
The Save Project dialog box displays.
- 3 Specify the new name for the project object and a comment describing it, and click OK.  
PowerBuilder saves the project object with the new name.

## Synchronizing objects

PowerBuilder maintains information about your libraries separately from ObjectCycle. An ObjectCycle archive is updated only when you perform an operation that affects the archive such as registering, checking out, or checking in an object. Therefore, you may need to resynchronize your PowerBuilder libraries periodically with the ObjectCycle archive.

When to synchronize

You need to synchronize if one of the following events desynchronizes objects in your PowerBuilder libraries:

- ◆ Unexpected system failures
- ◆ System crashes that occur as you register, check out, or check in an object
- ◆ Changes made to an archive outside PowerBuilder (such as in the ObjectCycle Manager)

What synchronizing does

The ObjectCycle interface lets you synchronize registered objects to ensure that the objects in your PowerBuilder library are the same as the latest version of the objects stored in the ObjectCycle archive.

❖ **To synchronize registered objects:**

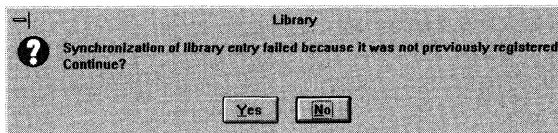
- 1 In the Library painter, select one or more registered objects that you want to synchronize.
- 2 Select Source>Synchronize from the menu bar.

PowerBuilder regenerates the objects from their definitions in the ObjectCycle archive.

---

### **Objects that you synchronize must be registered**

The objects that you select to synchronize must be registered with ObjectCycle. If you select an unregistered object, the following message box displays and PowerBuilder will *not* synchronize the object:





# Using the PowerBuilder PVCS Interface

## About this chapter

This chapter describes how to configure and use the PowerBuilder PVCS interface to manage the objects in your PowerBuilder applications.

## Contents

| <b>Topic</b>                                     | <b>Page</b> |
|--|-------------|
| About the PVCS interface                         | 68          |
| Configuring the PVCS interface                   | 69          |
| Checking objects out of PVCS                     | 79          |
| Modifying objects                                | 82          |
| Checking objects in to PVCS                      | 83          |
| Using version labels                             | 85          |
| Creating a new release                           | 88          |
| Viewing an object's change history               | 90          |
| Reporting  | 91          |
| Restoring an earlier revision level of an object | 94          |
| Restoring libraries                              | 97          |
| Synchronizing objects                            | 102         |

## Before you begin

Make sure you've installed PVCS Version Manager version 5.2 before you try to use the PowerBuilder PVCS interface.

## About the PVCS interface

The PowerBuilder PVCS interface lets you manage source without having to run PVCS Version Manager. The main PVCS source control features display in the Library painter, where your principal file management activities take place, and in the Project painter, where you can build your application.

FOR INFO See Chapter 1, "Preparing to Use Version Control with PowerBuilder" for general version control information.

### Configuring the PVCS interface

After you have installed the version control system, PowerBuilder does not automatically put your libraries under the control of PVCS. You must perform a series of configuration tasks to prepare your environment for source control. You must:

- ◆ Define a configuration file
- ◆ Create a work library and add it to your application library search path
- ◆ Register your objects

### Source control activities

Once you have configured your environment to use the PVCS interface, you can start managing your objects. Source control activities available with the PVCS interface include:

- ◆ Clearing object registration status
- ◆ Checking objects in and out
- ◆ Modifying objects
- ◆ Assigning version labels to your objects
- ◆ Creating a new release of your application
- ◆ Running reports
- ◆ Retrieving earlier revisions of objects
- ◆ Rebuilding earlier releases of your application
- ◆ Viewing a list of objects that are checked out
- ◆ Synchronizing your objects with PVCS archives

## Configuring the PVCS interface

Before you can start checking out and checking in your objects, you need to configure your environment by:

- ◆ Confirming your installation
- ◆ Connecting to PVCS
- ◆ Specifying a configuration file, which includes defining a user ID to PVCS and specifying archive directories
- ◆ Creating work libraries and adding them to your library search path
- ◆ Registering your objects with PVCS

### Confirming your installation

Before using the PowerBuilder interface to PVCS Version Manager, make sure that PVCS Version Manager has been installed on your system and that you are configured properly to use it.

Specifically, make sure that:

- ◆ Your VCS.CFG file specifies your user ID in the VCSID directive and the target archive directories in the VCSDir directive (and any other information required by PVCS Version Manager). For example:

```
VCSID=bobh  
VCSDir=C:\PVCSARCH
```

- ◆ You have defined an environment variable named VCSCFG that points to the directory containing your VCS.CFG file. Typically this is done in your AUTOEXEC.BAT file. For example:

```
SET VCSCFG=C:\PVCS
```

**FOR INFO** For more information, see the documentation that comes with PVCS Version Manager.

### Connecting to PVCS the first time

When you open the Library painter the first time, you will need to connect to PVCS. After that, PowerBuilder connects to PVCS automatically each time you open the Library painter.

❖ **To connect to PVCS:**

- 1 Open the Library painter and select Source>Connect from the menu bar.  
The Connect dialog box displays.
- 2 Choose PVCS from the dropdown listbox and click OK.  
PowerBuilder connects to PVCS.

If you have problems

If PowerBuilder fails to connect to PVCS, check that:

- ◆ PVCS Version Manager version 5.2 has been installed and is on your path
- ◆ The PBPVC060.DLL file has been installed in your PowerBuilder directory

## Specifying a configuration file

Configuration files are associated with each application whose objects you've put under PVCS's control. If you use configuration files, PowerBuilder automatically points to the correct archives when switching between applications. The PVCS configuration file contains a list of archive directories and your user ID.

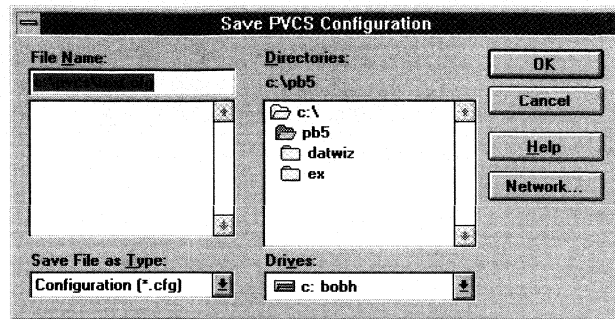
Although you may have defined an archive directory during your PVCS installation, you must define a configuration file for each application. The configuration file defines the location of your archive directories.

❖ **To define the PVCS configuration filename and location:**

- 1 In the Library painter, select Source>Configuration from the menu bar.  
The PVCS Configuration dialog box displays.

- 2 Click the File button.

The Save PVCS Configuration dialog box displays:



- 3 Select a directory in the Directories box, browsing other local or networked drives if necessary.
- 4 Type a filename with the CFG extension and click OK.

PowerBuilder locates the configuration file you defined in the specified directory.

## Defining archive directories

PVCS maintains copies of your objects in archive directories of your choice. An individual archive is a file that contains information about one PowerBuilder object such as its change history, descriptions of the changes and who made them, and dates and times of the changes. Every time you modify an object and check it into the archive, it becomes a new **revision**.

### Creating the archive directory

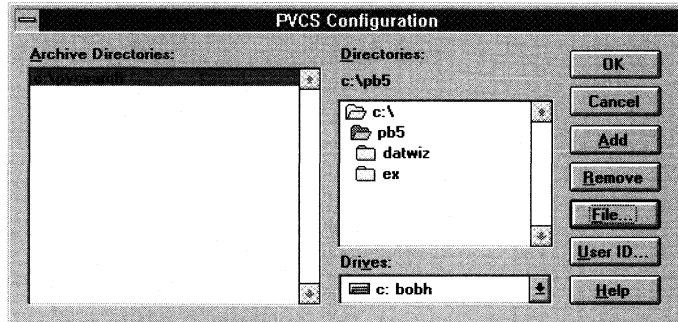
You choose existing directories as repositories for your archive files. You may want to create archive directories specifically for this purpose. When defining which directories are going to serve as archive directories, you can only select existing directories; so if you need to create a new directory for this purpose, you should do it before defining the directory as an archive directory in the Library painter.

### Defining multiple archive directories

A single archive directory can be defined for your application, but you may find it useful to define more than one archive directory. For example, you may want to define an archive directory for each PBL in your application, or one for each type of object: one for window objects, one for DataWindows, one for menus, and so on.

❖ **To add archive directories to your configuration:**

- 1 In the Library painter, select Source>Configuration from the menu bar.  
The PVCS Configuration dialog box displays:



- 2 Select a directory in the Directories box and click Add.  
The directory is added to the Archive Directories box.
- 3 Repeat this process for each archive directory you want to define.
- 4 Click OK to accept the archive directories you've chosen.

❖ **To remove archive directories from your configuration:**

- 1 In the Library painter, select Source>Configuration from the menu bar.  
The PVCS Configuration dialog box displays.
- 2 Select the directory you want to remove in the Archive Directories box and click the Remove button.  
PowerBuilder removes the directory as an archive directory.
- 3 Click OK to close the PVCS Configuration dialog box.

## Defining a user ID

Although you may have provided a user ID while installing PVCS, you must supply a PVCS user ID for each configuration file.

❖ **To define a PVCS user ID:**

- 1 In the Library painter, select Source>Configuration from the menu bar.  
The PVCS Configuration dialog box displays.
- 2 Click the User ID button.  
The Set Current User ID dialog box displays.

- 3 Type your PVCS user ID and click OK.

## Creating work libraries

When you check out objects for modifying, you need to put them into a work library, a place where you can keep your working copies of objects. Often your archive directories or public libraries reside on a network drive, accessible to everyone working in your development group. The work library is typically located on your own machine.

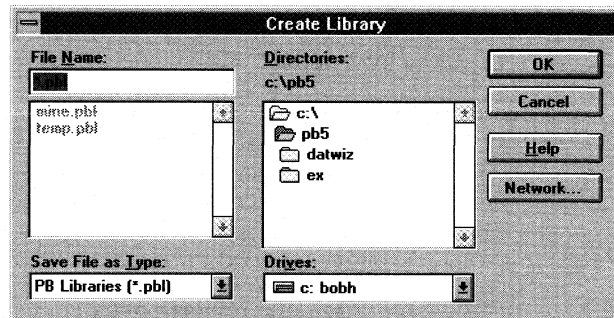
### ❖ To create a work library:

- 1 In the Library painter, click the Create button in the PainterBar.

*or*

Select Library>Create from the menu bar.

The Create Library dialog box displays:



- 2 Select a directory from the Directories box where you want to locate your work library.
- 3 Specify a filename for your work library with the PBL extension and click OK.

The property sheet displays.

- 4 Type a comment describing the purpose of your work library and click OK.

When you create a work library, you also need to add that library to your library search path.

Compiling with objects from your work library

If you want to build a new application executable using objects that you have checked out to your work library, make sure you list the work library in the library search path dialog box *before* your archive libraries. During the compile process, PowerBuilder looks for the first instance of the objects referenced in the application. If the archive library is before the work library in your library search path, the archive library versions of the objects will be used to compile the new executable.

---

**To move your work library to the top of the list**

To move your work library to the top of the list: In the Application painter, click the Properties button in the PainterBar and select the Libraries tab. Then select the library name in the Library Search Path box. Use CTRL+X to cut the library name and CTRL+V to paste it at the top of the library list.

---

## Registering your objects

Once you have defined archive directories, you can register your objects.

---

**Register all of your application's objects** In order to take full advantage of PVCS and to avoid problems, you should register *all* the objects in your application libraries. For example, if your application consists of five libraries and 100 objects, you should register them all in order to be assured of the protection that PVCS offers.

**If you cannot register multiple objects** If you get an error that an archive file already exists when you try to register more than one object, make sure the Make Selection Secure checkbox is clear in the Master Project in PVCS.

**FOR INFO** For more information on the Master Project, see the PVCS documentation.

---

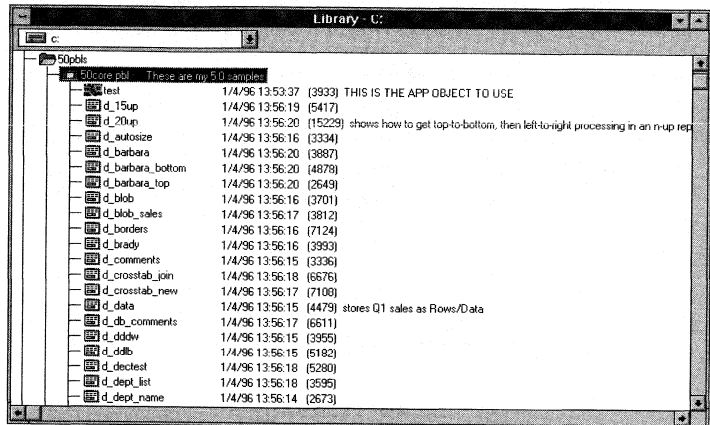
When you register objects, PVCS creates an archive in the archive directory for each object you specify. You can choose an archive directory in which to register your objects so that all objects from a single library could be located in a single archive directory.



❖ **To register objects:**

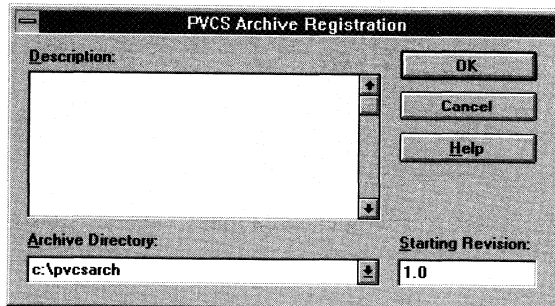
- 1 In the Library painter, double-click the library that has the objects you want to register.

A list of the objects in the library displays:



- 2 Select the objects you want to register.
- 3 Select Source>Register from the menu bar.

The PVCS Archive Registration dialog box displays:



- 4 Type a comment in the Description box.

The Archive Registration dialog box displays for each object you select. If the object has a PowerBuilder object comment associated with it, the comment displays in the Description box. You can accept the default comment if one displays, or edit the comment.

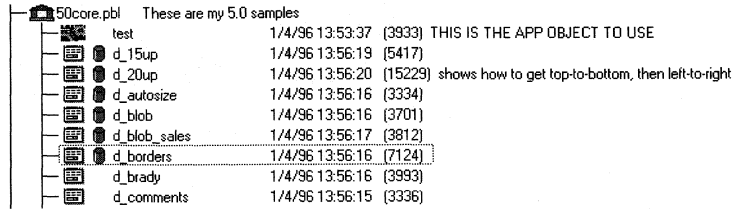
- 5 Select an archive directory, if necessary, from the Archive Directory dropdown listbox where you want to locate the object's archive.

- 6 (Optional) Type a starting revision number.

If you don't specify a starting revision number, the revision numbering will start at 1.0.

- 7 Click OK.

PowerBuilder assigns a registration icon to the object:



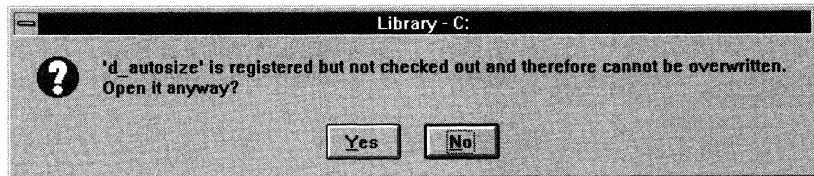
Once you have registered your objects, you can check out objects to make changes, and check them back in to the archive for safekeeping.

---

### Opening for read-only

If you want to simply view an object without making any changes, you can open the object without checking it out.

PowerBuilder displays a message box when you try to open a registered object that has not been checked out:



Click Yes to open a read-only version of the object.

---

## Viewing a list of registered objects

At any time you can see a list of all registered objects in your archive directories.

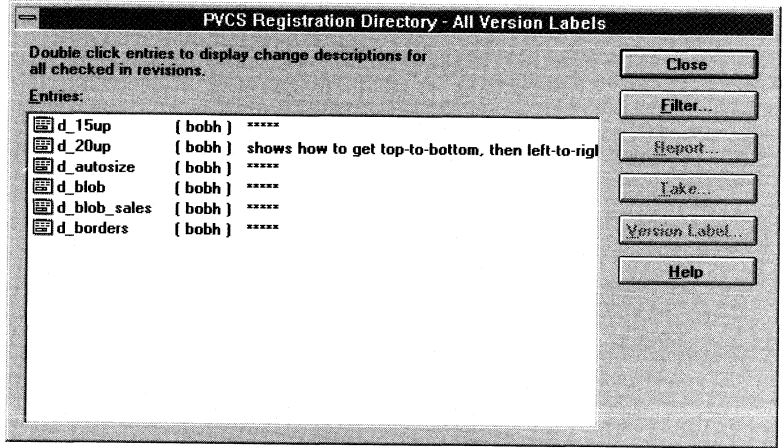
❖ **To view a list of registered objects:**

- ◆ Click the Directory button in the PainterBar.

*or*

Select Source>Registration Directory from the menu bar.

The Registration Directory dialog box displays showing all registered objects:



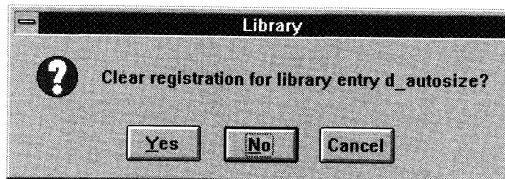
## Clearing an object's registered status

To remove an object from the control of PVCS, you clear its registration.

❖ **To clear an object's registration:**

- 1 In the Library painter, select the object.
- 2 Select Source>Clear Registration from the menu bar.

You are asked to confirm the action:



3 Click Yes.

PowerBuilder clears registration for the object and deletes the archive file automatically.

## Checking objects out of PVCS

Once you have registered an object in the version control system, you must check it out in order to modify it. When you check out an object, the archive version is locked so that no one else can check out the same object.

You can check objects out to work libraries that you have defined and that have been added to your library search path. Make sure you have defined a work library and that it has been added to your library search path before you check out any objects.

**FOR INFO** For complete information about check-out and check-in and how to set up libraries to support it, see the discussion of library management in the *PowerBuilder User's Guide*.

## Checking out objects

Before you check out objects from PVCS, they must be registered.

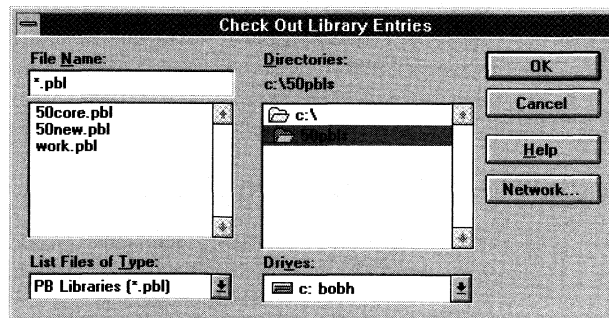
### ❖ To check out objects:

- 1 In the Library painter, select the objects you want to check out.
- 2 Check out the object or objects you want:

**To check out a group of objects** Click the Check Out button in the PainterBar *or* select Source>Check Out from the menu bar.

**To check out one object at a time** Right-click the object you want and then select Check Out from the popup menu.

The Check Out Library Entries dialog box displays showing the current directory and the libraries in that directory:



- 3 Type the name of the destination library in the File Name box.  
*or*  
Select the destination library from the library list.

The destination library is the one in which you want to save the working copy of the object or objects you are checking out. You must specify a library in the current application's library search path or you won't be able to save the working copy later.

- 4 Click OK.

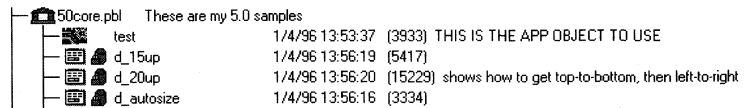
The Check Out Library Entries dialog box closes and the Library painter workspace displays.

What happens

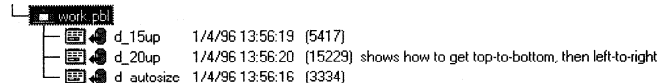
PowerBuilder does the following:

- ◆ Makes a working copy of each selected object and stores it in the destination library you specified
- ◆ Locks the object in the archive so no one else can check it out

In the object's archive, a lock icon shows that the object is locked:



In the work library the object has been checked out to, a check-out icon shows that it is the working copy:




---

### About your user ID

If you checked out files in PowerBuilder before using the PVCS interface, you specified a PowerBuilder-specific user ID. When you use the PVCS interface, PowerBuilder ignores this ID and uses the PVCS user ID.

---

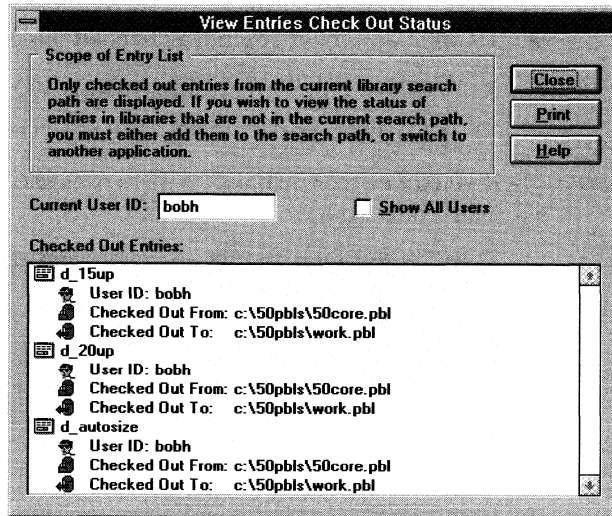
## Viewing the check-out status of objects

You can always see which objects have been checked out—by you or anyone else—from libraries in the library search path of the current application.

## ❖ To view a list of the objects that are checked out:

- 1 Select Source>View Check Out Status from the menu bar.

The View Entries Check Out Status dialog box displays showing the objects you have checked out:



- 2 To see a list of objects checked out by all users, select the Show All Users checkbox.
- 3 To print the list, click Print.

## Modifying objects

You should always use the checked-out versions of objects to make your changes. That way you can be sure that no one else is modifying the same objects.

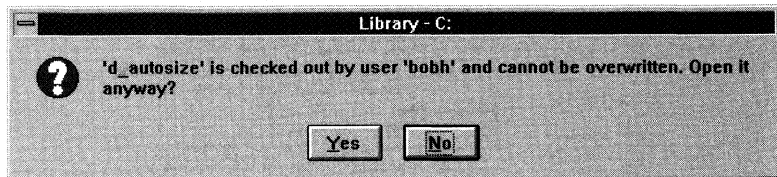
❖ **To open an object you have checked out for modification:**

- ◆ Double-click the checked-out object in your work library.

The painter associated with that object opens.

Security of checked-out objects

If you try to open an object that has been checked out by someone else, this dialog box displays:





## Checking objects in to PVCS

It is best to check in objects as follows:

- ◆ Check in *your objects* as soon as you have finished modifying them. This gives other developers access to them.
- ◆ Check in *all objects* before a major build of the application executable. This will ensure that the most recent versions are used to build the new executable.

---

### Building a new executable

You may want to be able to build a new executable on your own machine comprised of the objects you have been working on in your work library. In this case, be sure that your work library is *before* your archive in your library search path. Otherwise, PowerBuilder will use the objects it finds in your archive instead of the objects in your work library.

---

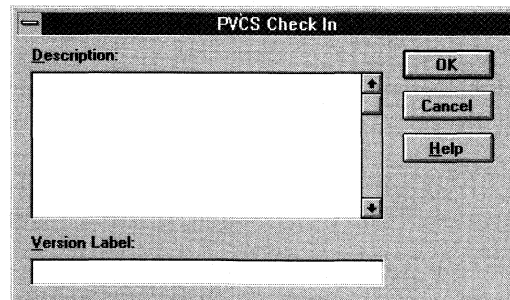
#### ❖ To check in an object to the archive:

- 1 In your work library list, select the object you want to check in.
- 2 Check in the object:

**To check in one object at a time** Right-click the object you want and then select Check In on the popup menu.

**To check in a group of objects** Click the Check In button on the PainterBar or select Source>Check In from the menu bar.

The PVCS Check In dialog box displays:



- 3 Type a description in the Description box. This description will appear in the PVCS archive report and the PVCS revision report.

- 4 (Optional) Type a version label.

This associates the version label with the revision level your object receives when being checked back into the archive. You can see this in the PVCS Archive Report dialog box.

You cannot assign the same version label to more than one revision of the same object.

- 5 Click OK.

#### What happens

PowerBuilder does the following:

- ◆ Replaces the object in the original library with the working copy and deletes the working copy from your work library.
- ◆ Increments the object's revision level. For example, if the object's revision level is 1.5 when you check it out, when you check it back in it receives a revision level of 1.6.
- ◆ Unlocks the archive version of the object so that other developers can check it out.

## Using version labels

A version label is a symbolic name that identifies a group of revisions of objects stored in separate archives. You can use version labels to associate the revisions that make up a version of an application so that you can select them quickly and easily. A version label remains associated with a revision even after you check in new revisions.

Keep in mind that:

- ◆ You can assign the same version label to any number of revisions of objects stored in different archives.
- ◆ You can assign multiple version labels to a single revision.
- ◆ You cannot assign the same version label to more than one revision of an object in an archive.

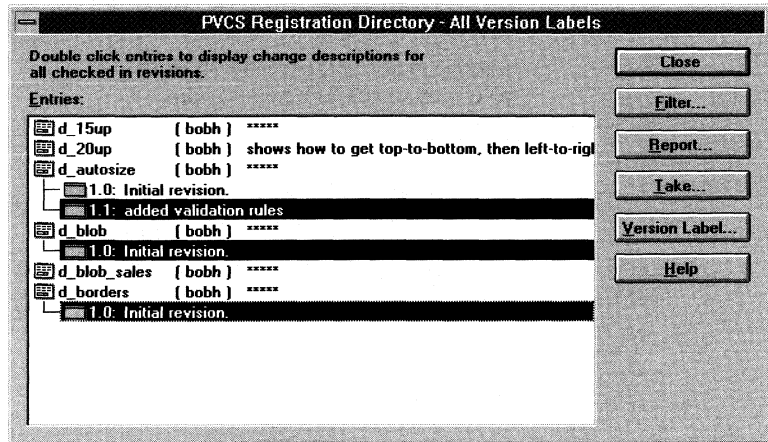
## Assigning a version label to a group of objects

Normally you will want to assign a version label to a group of objects as opposed to assigning a version label to an individual object.

❖ **To assign a version label to a group of objects:**

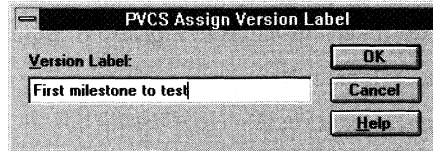
- 1 Select Source>Registration Directory from the menu bar.  
The Registration Directory dialog box displays.
- 2 Double-click the objects whose revision levels you want to display.  
The change history displays for each object you selected.

- 3 Select each of the revisions you want to assign the same version label.  
In the following example, three revisions were selected to receive the same version label:



- 4 Click the Version Label button.

The PVCS Assign Version Label dialog box displays:



- 5 Type a version label and click OK.

---

### Checking version label assignments

To check the results of the version label assignments, display an archive report for any of the archives that contain revisions you assigned a version label. You can also use the filter option in the Registration Directory dialog box.

---

## Using version labels to filter a list of revisions

You can filter the display of revisions by version label.

❖ **To filter by version label:**

- 1 Select Source>Registration Directory from the menu bar.  
The Registration Directory dialog box displays.
- 2 Click the Filter button.  
The PVCS Filter On Version Label dialog box displays.
- 3 Type the version label you want to filter the revision list by and click OK.  
The Registration Directory dialog box displays showing only the objects that contain revisions with the version label you specified. The version label displays in the dialog's title bar.
- 4 Double-click an object name to show the revision that has the version label you specified as the filtering argument.

## Retrieving revisions by version label

To retrieve objects using a version label, you filter a list of objects by version label and then select the objects you want from the resulting list.

❖ **To retrieve revisions by version label:**

- 1 Select Source>Registration Directory from the menu bar.  
The Registration Directory dialog box displays.
- 2 Click the Filter button.  
The PVCS Filter On Version Label dialog box displays.
- 3 Type the version label you want to filter the revision list by and click OK.  
The Registration Directory dialog box displays showing only the objects that contain revisions with the version label you specified.
- 4 Double-click an object name to show the revision that has the version label you specified as the filtering argument.
- 5 Select the revision you want from each object and click the Take button.  
The Select Destination Library dialog box displays.
- 6 Select a destination library in the Select Destination Library dialog box and click OK.  
PowerBuilder puts the objects you selected into the libraries you specified.

## Creating a new release

You can create a new release at any point in the development of your PowerBuilder application. Typically you create a new release when your application development has reached a milestone and you want to copy your libraries and their objects to a separate place while preserving your revision history.

PowerBuilder allows you some flexibility when building a new release. You can:

- ◆ Copy the entire revision history to the new set of archives  
*or*
- ◆ Start with a fresh set of archives by copying only the most current revision level to the new set of archives

When you start with a fresh set of archives, the most current revision number of each object will be maintained unless you specify a starting revision number.

## To create a new release

---

### Before you begin

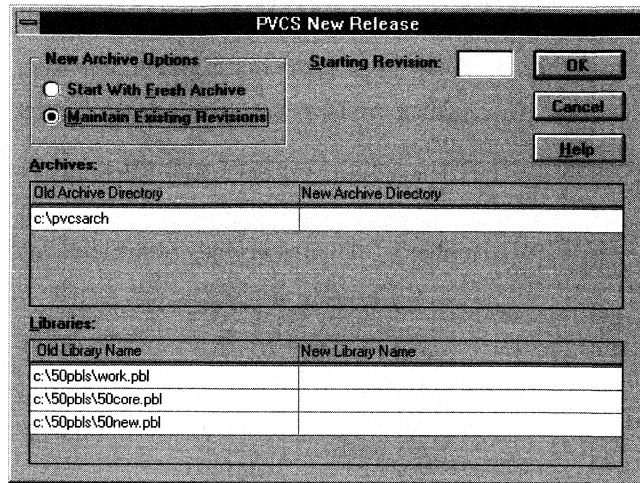
Make sure that all your objects have been checked in so that you start with an accurate revision history.

---

### ❖ To create a new release:

- 1 Select Source>Create New Release from the menu bar.  
The PVCS New Release dialog box displays.
- 2 Specify new archive options in the New Archive Options group box.

| To start a new archive with this                       | Do this   |
|--|---|
| The entire revision history                            | Select Maintain Existing Revision   |
| Fresh archives   | Select Start With Fresh Archives  |
| Fresh archives <i>and</i> at a specific revision level | Select Start With Fresh Archives<br>Specify a starting revision number in the Starting Revision box |



- 3 Type a new archive directory for each old archive directory specification in the New Archive Directory box.  
You can specify a new directory name. PowerBuilder prompts you before creating any new directories.
- 4 Type a new library name for each old library name in the New Library Name box.  
If you specify an existing library, PowerBuilder prompts you before overwriting library entries.
- 5 Click OK.

---

#### After you create the new release

In order to make changes to objects in your new release, you need to make the new application the current application by opening it in the Application painter.

You also need to define a configuration file for the new application.

---

## Viewing an object's change history

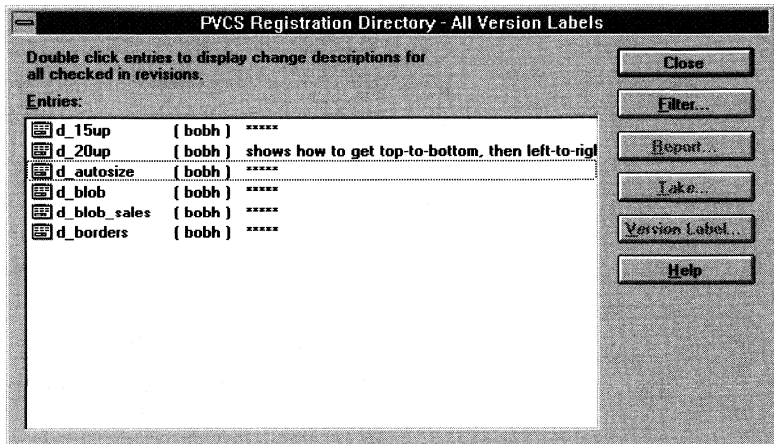
❖ To view the change history of an object:

- 1 Click the Directory button in the PainterBar.

*or*

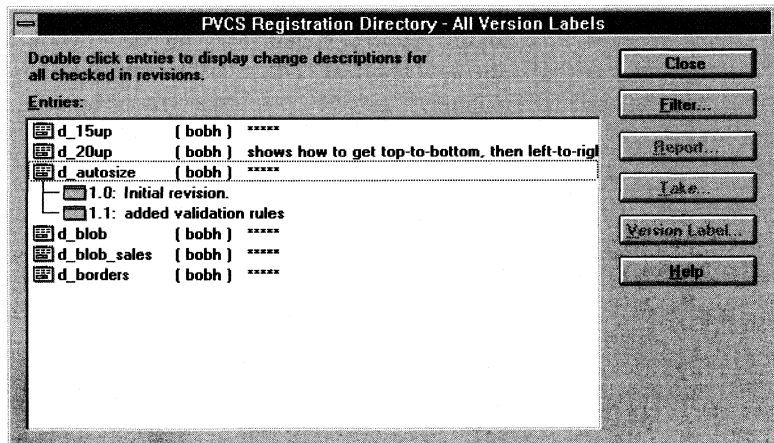
Select Source>Registration Directory from the menu bar.

The PVCS Registration Directory dialog box displays listing all the objects that are currently in the archive directories:



- 2 Double-click the object whose change history you want to see.

PowerBuilder expands the display to show all revision levels for the object:





## Reporting

There are two reports you can generate that provide information about archives and revisions. With these reports, you can:

- ◆ View or print a revision report of a specific revision level of an object
- ◆ View or print an archive report showing the entire change history of the object's archive

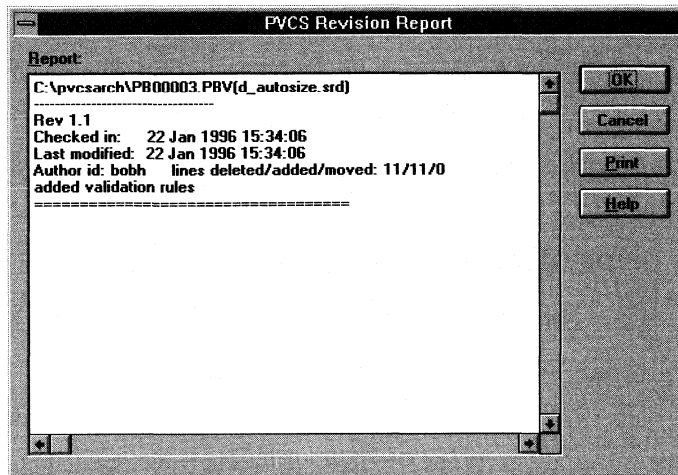
### Viewing or printing a revision report

The revision report details one revision level of one object.

- ❖ **To view or print a revision report:**
  - 1 Select Source>Registration Directory from the menu bar.  
The PVCS Registration Directory dialog box displays.
  - 2 Double-click an object name.  
The object's change history displays in a hierarchical format.
  - 3 Select the revision level you want to view or print.

- 4 Click the Report button.

The PVCS Revision Report dialog box displays:



FOR INFO For information about the categories presented in this report, see your PVCS Version Manager documentation.

- 5 Click Print to print the report.

## Viewing or printing an archive report

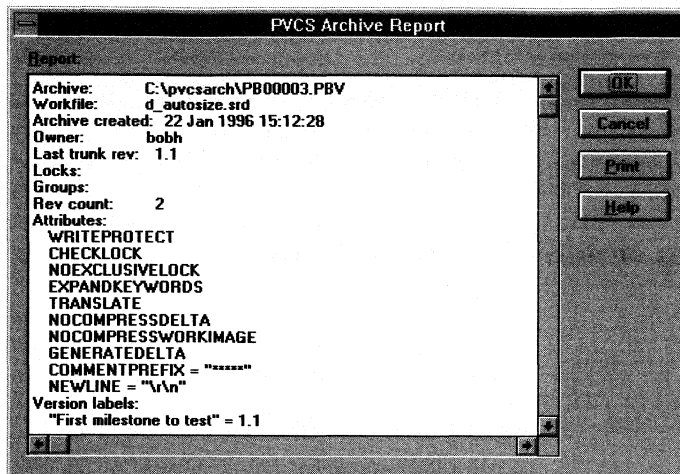
The archive report details the change history of an object's archive. This report gives you the entire history of the object since its registration.

❖ **To view or print an archive report:**

- 1 Select Source>Registration Directory from the menu bar.  
The PVCS Registration Directory dialog box displays.
- 2 Select an object name.

- 3 Click the Report button.

The PVCS Archive Report dialog box displays:



FOR INFO For information about the categories presented in this report, see your PVCS Version Manager documentation.

- 4 Click Print to print the report.

## Copying a revision or archive report to a file

You can copy the text of the revision or archive report to a file.

### ❖ To copy report text to a file:

- 1 Using your mouse, select the text you want to copy to a file.
- 2 Press CTRL+C to copy the selected text to the clipboard.
- 3 You can then open a text editor and paste the text from the clipboard into a file by pressing CTRL+V.

## Restoring an earlier revision level of an object

A principal reason for putting your objects under source control is to permit the restoration of earlier revision levels of your objects. The PVCS interface makes this easy.

Before you can restore an earlier revision level, you need to decide which revision level you want.

### Deciding which revision level to restore

There are several items you can view to help you decide which revision level of an object you want. All depend on the careful commenting of revision levels as they are checked back in to PVCS archives:

| What you can view  | Description  |
|--|--|
| An object's revision history in the PVCS Registration Directory dialog box | The first thing you may want to do is simply view the list of revisions for the object you want to restore<br><br>FOR INFO For instructions, see "Viewing an object's change history" on page 90                 |
| A revision report of a specific revision level for an object               | The revision report details one revision level of one object<br><br>FOR INFO For instructions, see "Reporting" on page 91  |
| An archive report showing the complete change history for an archive       | The archive report details the change history of an object's archive. This report gives you the entire history of the object since its registration<br><br>FOR INFO For instructions, see "Reporting" on page 91 |

### Performing the restoration

Before you can restore an object, you must determine the revision level to restore.

❖ **To choose a revision level to restore:**

- 1 Check out the current revision of the object you want to restore to your work library.

FOR INFO For instructions, see "Checking objects out of PVCS" on page 79.

- 2 Use the Registration Directory dialog box to choose the revision level of the object you want to restore.

FOR INFO For instructions, see "Restoring an earlier revision level of an object" on page 94.

Once you know which revision level of an object you want to restore, you select it from the revision level list.

❖ **To restore a previous revision level of an object:**

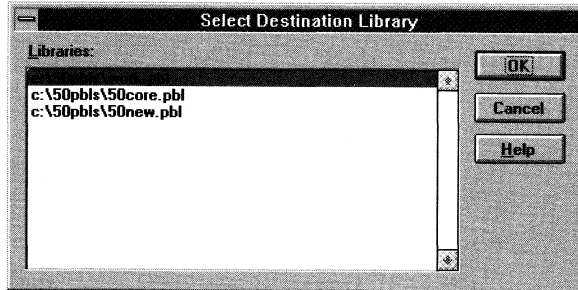
- 1 Select Source>Registration Directory from the menu bar.

The PVCS Registration Directory dialog box displays.

- 2 Select the revision you want to restore.

- 3 Click the Take button.

The Select Destination Library dialog box displays:



- 4 Select a destination library.

PowerBuilder lists the libraries in the current application's library search path.

- 5 Click OK.

PowerBuilder copies the revision level of the object over the current revision you checked out and regenerates (recompiles) the object.

- 6 Check the restored revision back in to the archive.

The restored revision becomes the current revision of the object.

---

**To restore the replaced version**

To restore the replaced version, repeat this procedure choosing the revision from the Registration Directory dialog box.

---

## Restoring libraries

When using PVCS with PowerBuilder applications, you can restore earlier versions of PowerBuilder libraries using the Restore Libraries option in the Project painter. When you restore libraries, PowerBuilder puts your objects into specified libraries. Once your libraries have been restored to their new location, you can rebuild the application executable.

Restoring libraries is enabled by the project object. Each time you build an executable in the Project painter, PowerBuilder saves information about the application in the project object that can be used later to restore your libraries. The following information is stored for each object:

- ◆ PowerBuilder object names with .PBL name
- ◆ Archive names and revision numbers
- ◆ Version labels

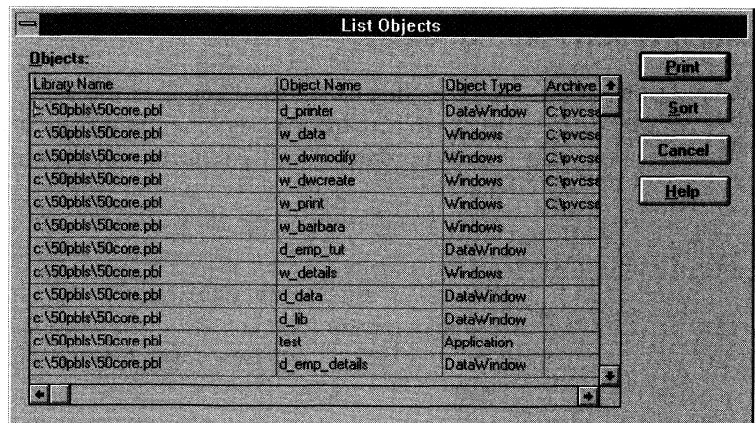
## Viewing a list of objects in a project

You can view the list of objects stored in your project object. This list shows what objects comprise your application.

❖ **To see the list of objects:**

- 1 In the Project painter, select Design>List Objects from the menu bar.

The List Objects dialog box displays:



- 2 (Optional) Click the Print button to Print the list, or click the Sort button to specify the order in which the objects are listed.

Each time you build an executable in the Project painter, PowerBuilder updates the information stored in the project object to include the correct list of objects and the information listed above.

## Two methods of restoring libraries

Because PowerBuilder stores information about the objects in the application in the project object, you can restore your libraries by retrieving a project object revision, or by opening an earlier version of a project object saved in your library list.

These two different methods are available for restoring libraries:

| To use this method  | You must  | And use  |
|---|---|--|
| Restoring by retrieving an earlier revision level of a project object from PVCS                   | Put the project object under PVCS's control     | PVCS to track each revision of the project object  |
| Restoring by opening a project object saved to a new name in the current application library list | Leave the project object outside PVCS's control | The Save As option on the Project painter File menu to save the object to a new name in the current library list |

## Deciding which method to use

Method 1: using PVCS for project objects

Use the first technique if you need the ability to restore any revision level of your project object at any time and if you expect to have many revision levels of your project object. This method has the advantage of being very secure. In addition, you'll always see only a single project object in your library listing, since PVCS is doing the work of tracking changes to it.

---

### Use comments and version labels

Using comments and version labels when handling project objects under the control of PVCS will help you determine which project object you want to retrieve.

---



**Method 2: saving project objects to new names**

Use the second technique if you'll be saving a moderate number of project objects that you may want to restore from. This method has the advantage of being very simple to institute. To open an earlier version of a project object, you simply double-click it in the library listing. But for large, complex development projects with many intermittent builds, this method may not be what you need. You could wind up with more project objects in your library than you like.

---

**Create a separate library for your project objects**

You might want to create a separate library just for keeping project objects that you create each time you build an application executable or change the definition of a project object. This way you won't have to go searching for a project object when you want to open it and restore libraries. Be sure to provide descriptive comments for your project objects.

---

**Method 1: retrieving project objects from PVCS**

The project object stores revision level information about your objects, archives, version labels, and so on. As a result, your project object revision history enables you to recreate libraries used to build earlier versions of your application. To restore an earlier version, you:

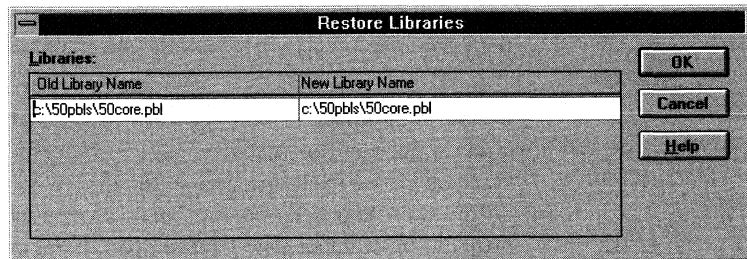
- 1 Retrieve the version of the project object you want using the PVCS Registration Directory dialog box.  
FOR INFO See "Restoring an earlier revision level of an object" on page 94.
- 2 Open the Project painter with the project object you retrieved.  
FOR INFO For more about the Project painter, see the *PowerBuilder User's Guide*.
- 3 Use the Project painter Restore Libraries feature to restore the versions of the libraries used for that project object to directories you specify, as follows.

Once you have retrieved the project object you want from PVCS and opened the Project painter, you can restore the project object's libraries.

### ❖ To restore libraries:

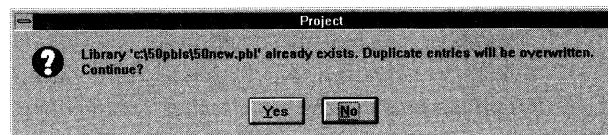
- 1 In the Project painter, select Design>Restore Libraries from the menu bar.

The Restore Libraries dialog box displays:



- 2 Specify a new library name for each Old Library Name listed. The New Library Name is where the new libraries will be placed.

If you specify existing libraries, PowerBuilder prompts you to overwrite the entries in the libraries you specify:



If you click Yes, the entries will be overwritten. If you click No, PowerBuilder returns you to the Restore Libraries dialog box.

PowerBuilder will create the libraries for you if they don't exist.

- 3 Click OK.

PowerBuilder restores the libraries to the state they were in when you built the application executable.

## Method 2: creating a new project object

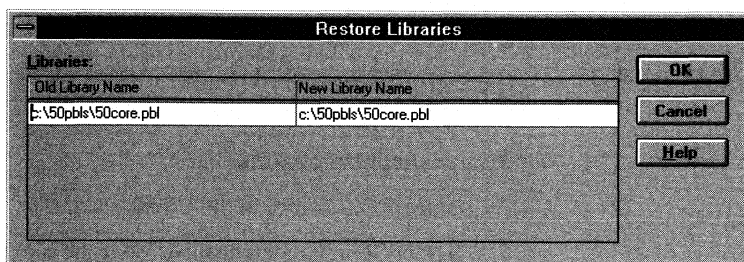
If you choose not to put your project object(s) under PVCS's control, you can still restore libraries by opening an older project object. This is a simple method that gives you access to versions of project objects you create each time you execute a build or make a change to the project object definition. This method requires you to:

- 1 Save each project object to a new name after every build you execute or after making a change to the project object definition.
- 2 Restore a project object in the Project painter.

❖ **To restore libraries:**

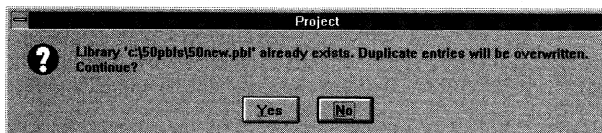
- 1 Select the project object you want in the library list.
- 2 Double-click the project object to open the Project painter. The project object definitions display in the painter workspace.
- 3 Select Design>Restore Libraries from the menu bar.

The Restore Libraries dialog box displays:



- 4 Specify a new library name for each Old Library Name listed. The New Library Name is where the new libraries will be placed.

If you specify existing libraries, PowerBuilder prompts you to overwrite the entries in the libraries you specify:



If you click Yes, the entries will be overwritten. If you click No, PowerBuilder returns you to the Restore Libraries dialog box.

PowerBuilder will create the libraries for you if they don't exist.

- 5 Click OK.

PowerBuilder restores the libraries to the state they were in when you built the application executable.

## Synchronizing objects

PowerBuilder maintains information about your libraries separately from PVCS. The PVCS archive is updated only when you perform an operation that affects the PVCS archive. You may need to resynchronize your libraries with the PVCS archives.

**When to synchronize** You need to synchronize if one of the following events desynchronizes your PowerBuilder library entries:

- ◆ Unexpected system failures
- ◆ System crashes during a registration, check-in, or check-out operation
- ◆ Changes made to an archive outside of PowerBuilder

**What synchronizing does** You synchronize objects to make sure that the object in your PowerBuilder library is the same as the latest revision stored in PVCS.

❖ **To synchronize objects:**

- 1 Select the objects in the Library painter.
- 2 Select Source>Synchronize from the menu bar.

PowerBuilder regenerates (recompiles) the objects from their definitions in PVCS.

# Using the PowerBuilder MKS Source Integrity Interface

## About this chapter

This chapter describes how to use the PowerBuilder MKS Source Integrity version control interface on your 16-bit or 32-bit Windows platform to manage the objects in your PowerBuilder application.

## Contents

| Topic  | Page |
|--|------|
| Basic steps for using the Source Integrity interface | 104  |
| About the Source Integrity interface                 | 105  |
| Installing the required software                     | 107  |
| Connecting to Source Integrity for the first time    | 108  |
| Defining the configuration file                      | 110  |
| Creating work libraries                              | 115  |
| Registering PowerBuilder objects                     | 118  |
| Checking out objects from Source Integrity           | 123  |
| Modifying checked-out objects                        | 126  |
| Checking in objects to Source Integrity              | 127  |
| Creating a new release                               | 129  |
| Using revision labels                                | 132  |
| Viewing an object's revision history                 | 139  |
| Displaying reports                                   | 141  |
| Restoring earlier revisions of an object             | 145  |
| Restoring libraries                                  | 149  |
| Synchronizing objects                                | 157  |

## Before you begin

Make sure you have installed MKS Source Integrity for Windows and the PowerBuilder Source Integrity interface.

## Basic steps for using the Source Integrity interface

The following table lists the basic steps for using the PowerBuilder Source Integrity interface and where you can find information about each step:

| <b>What you do</b>   | <b>For information, see</b>   |
|--|---|
| (Optional) Get an overview of the Source Integrity interface                           | "About the Source Integrity interface" on page 105  |
| Install the required MKS and PowerBuilder software                                     | "Installing the required software" on page 107  |
| Connect to MKS Source Integrity from PowerBuilder the first time you use the interface | "Connecting to Source Integrity for the first time" on page 108   |
| Define the configuration file  | "Defining the configuration file" on page 110   |
| Create work libraries for your objects   | "Creating work libraries" on page 115   |
| Register your objects  | "Registering PowerBuilder objects" on page 118  |
| Check out registered objects   | "Checking out objects from Source Integrity" on page 123  |
| Modify checked-out objects   | "Modifying checked-out objects" on page 126   |
| Check in objects   | "Checking in objects to Source Integrity" on page 127   |
| Create a new release   | "Creating a new release" on page 129  |
| Manage your objects as needed  | "Using revision labels" on page 132<br>"Viewing an object's revision history" on page 139<br>"Displaying reports" on page 141<br>"Restoring earlier revisions of an object" on page 145<br>"Restoring libraries" on page 149<br>"Synchronizing objects" on page 157 |

## About the Source Integrity interface

The PowerBuilder Source Integrity interface lets you manage source outside the MKS Source Integrity product. In the context of PowerBuilder development, your source is a collection of objects stored in PowerBuilder libraries.

**FOR INFO** For general version control information, see Chapter 1, "Preparing to Use Version Control with PowerBuilder".

### Libraries, projects, and sandboxes

If you are more familiar with Source Integrity than with PowerBuilder, the concepts of public and private libraries may be new to you. It may help to think of a public library as analogous to a Source Integrity project and a private library as analogous to a Source Integrity sandbox. (For more about projects and sandboxes, see your Source Integrity documentation.)

**FOR INFO** For complete information about organizing and managing PowerBuilder libraries, see the *Project Primer*.

### Setting up the Source Integrity interface

After you install the MKS Source Integrity software and connect to Source Integrity, you must set up your environment to put your objects under the control of Source Integrity. Setup tasks include:

- ◆ Creating and saving a configuration file
- ◆ Defining archive directories
- ◆ Specifying a user ID
- ◆ Creating work libraries
- ◆ Adding your work libraries to your application's library search path
- ◆ Registering your objects

What you can do

After you set it up, the Source Integrity interface makes it easy for you to manage your PowerBuilder objects. You can:

- ◆ Register objects and clear an object's registration
- ◆ Open read-only versions of registered objects
- ◆ List registered objects
- ◆ Check out, modify, and check in objects
- ◆ Display the status of checked-out objects
- ◆ Create a new release of your application
- ◆ Assign revision labels to objects and filter revision lists by this label
- ◆ View an object's revision history
- ◆ Display reports about archives and revisions
- ◆ Restore earlier revisions of objects
- ◆ Restore earlier versions of application libraries
- ◆ Synchronize objects with Source Integrity archives



## Installing the required software

Before you can start using the Source Integrity interface, you must install MKS Source Integrity for Windows and the PowerBuilder MKS Source Integrity interface.

You must obtain the MKS Source Integrity for Windows software from Mortice Kern Systems, Inc.

❖ **To install the required software on your computer:**

- 1 Install MKS Source Integrity for Windows.

FOR INFO For instructions, see your Source Integrity documentation.

- 2 Install the PowerBuilder MKS Source Integrity interface, PBRCS060.DLL.

FOR INFO For instructions, see the *PowerBuilder Installation Guide*.

- 3 Make sure your path statement includes both of the following:

- ◆ The directory where the MKS Source Integrity for Windows software resides
- ◆ The directory where PBRCS060.DLL resides

## Connecting to Source Integrity for the first time

The first time you open the Library painter in PowerBuilder, you must connect to Source Integrity. After that, PowerBuilder connects to Source Integrity automatically each time you open the Library painter.

❖ **To connect to Source Integrity in PowerBuilder for the first time:**

- 1 Open your application in the Application painter. This makes it the current application.

FOR INFO For instructions on using the Application painter, see the *PowerBuilder User's Guide*.

- 2 Open the Library painter.

FOR INFO For instructions, see the *PowerBuilder User's Guide*.

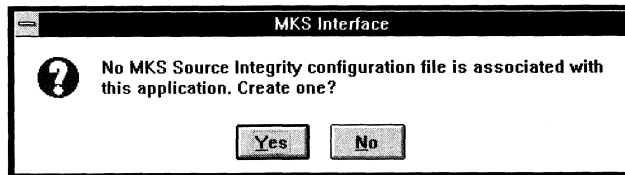
- 3 Select Source>Connect from the menu bar.

The Connect dialog box displays.

- 4 Select RCS from the Vendors dropdown listbox.

- 5 Click OK.

PowerBuilder connects to Source Integrity and displays the following message box prompting you to create a configuration file for this application:



What to do next

Next you need to define the configuration file.

FOR INFO For instructions on creating the configuration file, see "Creating and saving the configuration file" on page 110.

**If PowerBuilder  
cannot connect**

If PowerBuilder is unable to connect to Source Integrity, make sure you have done all of the following and then try to reconnect:

- ◆ Installed MKS Source Integrity for Windows
- ◆ Installed the PowerBuilder MKS Source Integrity interface (PBRCS060.DLL) on your computer
- ◆ Edited your path statement to include the directories where Source Integrity and PBRCS060.DLL reside, and rebooted your computer so the revised path takes effect

## Defining the configuration file

Before you can start checking out and checking in PowerBuilder objects with the Source Integrity interface, you must set up a Source Integrity configuration file for your PowerBuilder application. Configuration files are associated with each application whose objects you've put under Source Integrity's control. If you use configuration files, PowerBuilder automatically points to the correct archives when switching between applications.

What the configuration file does

The Source Integrity configuration file specifies your archive directories (RCSPATH) and user ID (LOGNAME). The configuration filename must have a CFG extension. For example, here is the contents of a sample Source Integrity configuration file named TESTAPP.CFG:

```
RCSPATH=c:\pb4\mkssourc;  
LOGNAME=frans
```

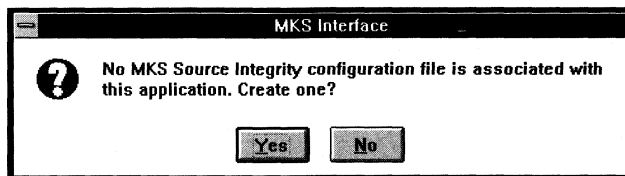
Tasks

Although you may have defined an archive directory (RCSPATH) while installing Source Integrity, you must define a new configuration file for each PowerBuilder application. Defining the configuration file includes these tasks:

- ◆ Creating and saving the configuration file
- ◆ Defining archive directories
- ◆ Specifying a user ID

## Creating and saving the configuration file

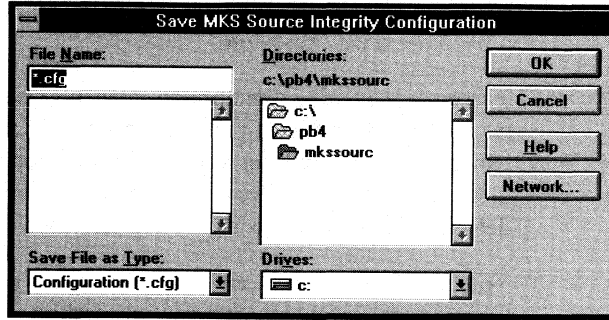
When you connect to Source Integrity for the first time, the following message box displays prompting you to create a configuration file for the current application:



❖ **To create and save a Source Integrity configuration file for the first time:**

- 1 Click Yes in the message box prompting you to create the configuration file.

The Save MKS Source Integrity Configuration dialog box displays:



---

**Another way to display the dialog box**

You can also display this dialog box by clicking the File button in the MKS Source Integrity Configuration dialog box (shown in the next section). This may be useful if you want to change the configuration filename later.

---

- 2 Select a directory in the Directories box, browsing other local or networked drives if necessary.
- 3 In the File Name box, type a configuration filename with the CFG extension.

If you omit the CFG extension, PowerBuilder automatically adds it to the configuration filename.

- 4 Click OK.

PowerBuilder creates the configuration file and saves it in the specified directory.

The MKS Source Integrity Configuration dialog box displays so you can define archive directories, as described in "Defining archive directories" next.

## Defining archive directories

The PowerBuilder Source Integrity interface maintains copies of your objects in archive directories that you specify. An individual **archive** is a file that contains information about one PowerBuilder object, such as its change history, descriptions of the changes and who made them, and dates and times of the changes. Every time you modify an object and check it into the archive, it becomes a new **revision**.

### Specifying existing directories

You choose existing directories as repositories for your archive files. You may want to create archive directories specifically for this purpose. When defining which directories will serve as archive directories, you can only select existing directories. If you need to create a new directory for this purpose, do so before defining the directory as an archive directory.

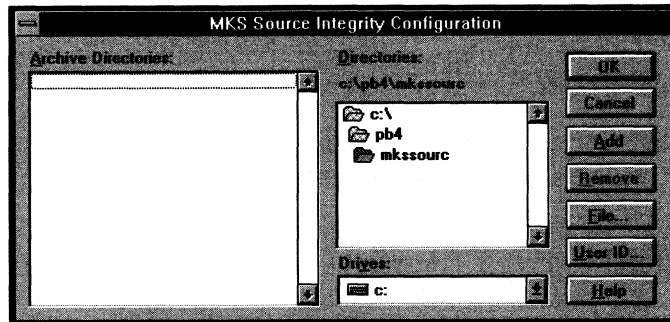
### Defining multiple archive directories

You can define a single archive directory for your application, but you may find it more useful to define multiple archive directories for your application. For example, you may want to define an archive directory for each PBL in your application, or one for each type of object: one for window objects, one for DataWindows, one for menus, and so on.

#### ❖ To add archive directories to your configuration:

- 1 In the Library painter, select Source>Configuration from the menu bar.

The MKS Source Integrity Configuration dialog box displays:



- 2 Select a directory in the Directories box and click the Add button.  
The directory is added to the Archive Directories box.
- 3 Repeat step 2 for each additional archive directory you want to add.
- 4 Click OK.

PowerBuilder adds the specified archive directories to your Source Integrity configuration file.

❖ **To remove archive directories from your configuration:**

- 1 In the Library painter, select Source>Configuration from the menu bar.  
The MKS Source Integrity Configuration dialog box displays.
- 2 Select the directory you want to remove from the Archive Directories box and click the Remove button.  
PowerBuilder removes the directory from the Archive Directories box.
- 3 Repeat step 2 for each additional archive directory you want to remove.
- 4 Click OK.

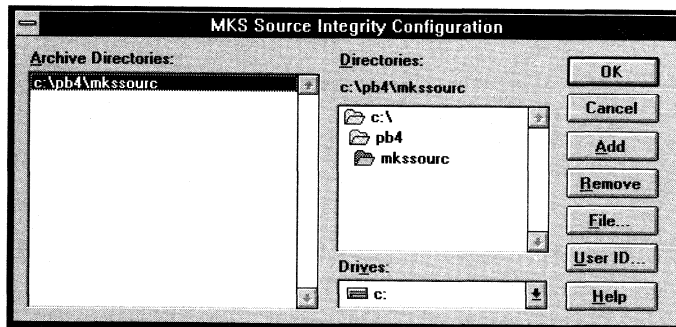
PowerBuilder removes the specified archive directories from your Source Integrity configuration file.

## Specifying a user ID

Although you may have supplied a user ID while installing Source Integrity, you must also supply a user ID for each configuration file.

❖ **To specify a user ID:**

- 1 In the Library painter, select Source>Configuration from the menu bar.  
The MKS Source Integrity Configuration dialog box displays:



- 2 Click the User ID button.  
The Set Current User ID dialog box displays.
- 3 Type your user ID and click OK.  
PowerBuilder adds the user ID to your Source Integrity configuration file.

- 4 Click OK in the MKS Source Integrity Configuration dialog box to close the dialog box.



## Creating work libraries

### About the work library

When you use the Source Integrity interface to check out PowerBuilder objects for modifying, you put the objects in a work library. A work library is a private library where you keep working copies of objects.

Often, your archive directories or public libraries reside on a network server, accessible to everyone in your development group. The work library typically resides on your own computer.

### Tasks

Creating a work library consists of two tasks:

- ◆ Creating the work library
- ◆ Adding the work library to your application's library search path

## Creating the work library

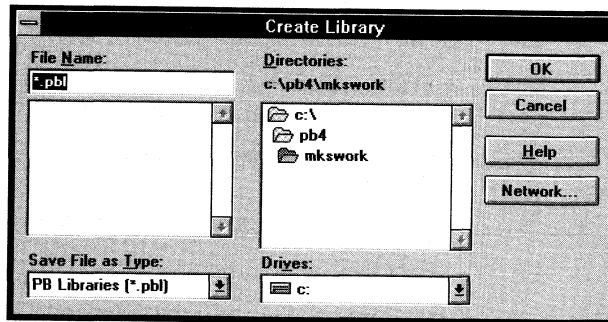
### ❖ To create a work library:

- 1 In the Library painter, click the Create button.

*or*

Select Library>Create from the menu bar.

The Create Library dialog box displays:

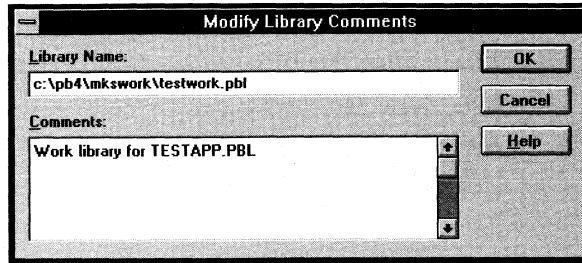


- 2 Select a directory from the Directories box where you want to locate the work library.
- 3 Specify a filename for the work library with the PBL extension.
- 4 Click OK.

The Modify Library Comments dialog box displays.

- 5 Type a comment describing the purpose of your work library.

For example:



- 6 Click OK.

PowerBuilder creates the work library in the directory you specified.

## Adding the work library to the library search path

After you create the work library, you must add it to the current application's library search path. This allows you to save changes you make to objects checked out to the work library.

### ❖ To add the work library to the library search path:

- 1 Open the Application painter and make sure your current application is open.  
*FOR INFO* For instructions on using the Application painter, see the *PowerBuilder User's Guide*.
- 2 Select Entry>Properties from the menu bar.  
*or*  
Click the Properties button in the PainterBar.
- 3 Select the Libraries tab.
- 4 Add the work library to the search path. You can select the library from a list by clicking the Browse button to display a dialog.
- 5 Click OK.

PowerBuilder adds the work library to your application's library search path.

## Using objects from your work library

If you want to run your application using objects that you checked out to your work library, your work library must appear *before* your public libraries in the application's library search path.

While running an application or building an executable, PowerBuilder looks for the *first instance* of the objects referenced in the application. If the public library appears before the work library in your library search path, PowerBuilder uses the public library versions of the objects in the application.

❖ **To move your work library to the beginning of the search path:**

- 1 In the Application painter, click the Properties button.  
*or*  
Select Entry>Properties from the menu bar.
- 2 Go to the Properties tab.
- 3 Select the name of the work library in the Library Search Path box and press CTRL+X to cut it.
- 4 Position the insertion point at the beginning of the list in the Library Search Path box and press CTRL+V to paste the work library name there.
- 5 Click OK.

PowerBuilder updates your application's library search path.

## Registering PowerBuilder objects

After you define archive directories, you can register the objects in your PowerBuilder application. **Registering** an object places it under Source Integrity's control. When you register objects, the Source Integrity interface creates a separate archive file in the archive directory for each object you specify.

Where to register objects

If you want, you can choose an archive directory in which to register your objects so that all objects from a single library are located in a single archive directory.

---

### Register all of your application's objects

To take full advantage of Source Integrity's version control features and to avoid problems, you should register *all* of the objects in *all* of your application libraries.

---

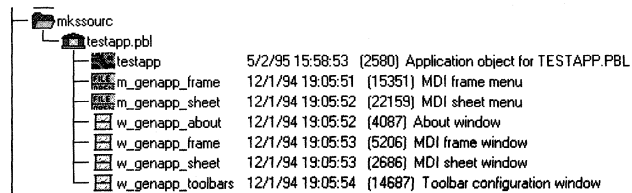
## Registering objects

You can register a PowerBuilder object at any time. After you register an object you can check it out of the public library into your work library, modify the object as needed, and check the object back into the public library for safekeeping.

### ❖ To register objects:

- 1 In the Library painter, double-click the library that contains the objects you want to register.

A list of the objects displays in the Library painter window:



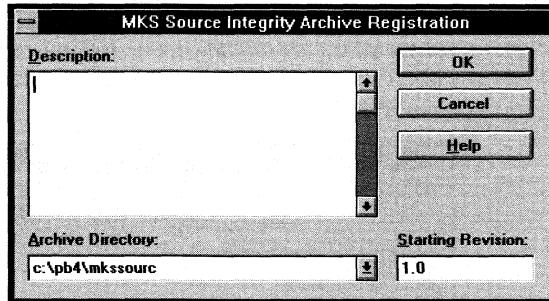
- 2 Select one or more objects that you want to register.

### Selecting a group of objects

If you select a group of objects to register, a separate MKS Source Integrity Archive Registration dialog box displays for *each* object you select.

- 3 Select Source>Register from the menu bar.

The MKS Source Integrity Archive Registration dialog box displays:



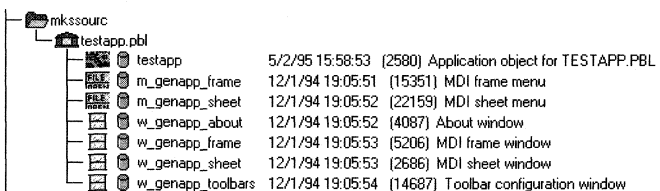
- 4 Type a comment identifying the object in the Description box.

If the object already has a comment associated with it, the comment displays in the Description box. You can accept this comment or edit it if you want.

The description will display in the MKS Source Integrity Registration Directory dialog box for this revision of the object when you display the object's change history. (For how, see "Viewing an object's revision history" on page 139.)

- 5 (Optional) Select an archive directory from the Archive Directory dropdown listbox. This is where you want to locate the object's archive.
- 6 (Optional) Specify a starting revision number in the Starting Revision box.
- 7 Click OK.

PowerBuilder registers the objects you selected and assigns a registration icon to each object in the Library painter:



## Listing registered objects

You can view a list of registered objects in your current application at any time.

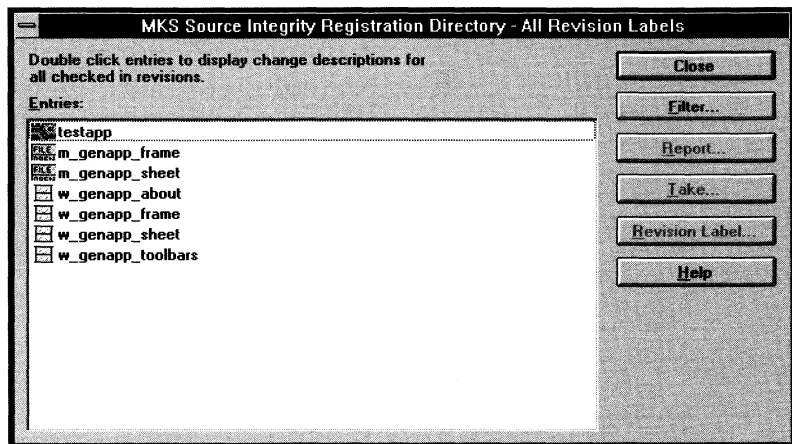
❖ **To view a list of registered objects:**

- ◆ In the Library painter, click the Directory button.

or

Select Source>Registration from the menu bar.

The MKS Source Integrity Registration Directory dialog box displays listing all registered objects in your current application. *All Revision Labels* displays in the title bar to indicate that all objects are displayed even if they have different revision labels assigned:



---

### Viewing an object's revision history

You can double-click any registered object listed in the MKS Source Integrity Registration Directory dialog box to see that object's revision history.

FOR INFO For how, see "Viewing an object's revision history" on page 139.

---

## Clearing an object's registration

You can remove an object from Source Integrity's control at any time by clearing its registration.

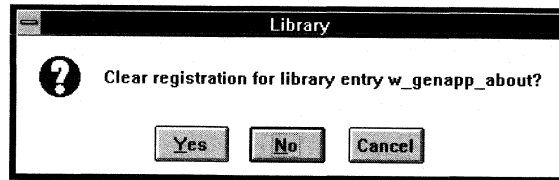
❖ **To clear an object's registration:**

- 1 In the Library painter, select one or more objects to clear.

If you select a group of objects to clear their registration, a separate message box displays for *each* object you select.

- 2 Select Source>Clear Registration from the menu bar.

A message box similar to the following displays:



- 3 Click Yes to clear the object's registration.

What happens

PowerBuilder does the following for each object:

- ◆ Clears the object's registration
- ◆ Deletes the archive file
- ◆ Removes the registration icon for the object in the Library painter

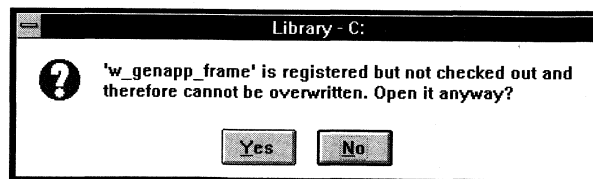
## Opening read-only versions of registered objects

If you want to view a registered object *without* making changes to it, you can open a read-only version of the object without checking it out.

❖ **To open a read-only version of a registered object:**

- 1 In the Library painter, double-click the registered object you want to open.

A message box similar to the following displays:



- 2 Click Yes to open a read-only version of the registered object.

The associated PowerBuilder painter opens and displays the selected object. You will be able to view the object but you will *not* be able to save changes to it.



## Checking out objects from Source Integrity

After you register an object with the Source Integrity interface, you must check it out of the public library in order to modify it. When you check out an object, PowerBuilder locks this version of the object so no one else can check it out while you are working on it.

Before you check out objects

Typically, you check out objects to work libraries. Before you check out objects, make sure you have:

- ◆ Created the work library
- ◆ Added the work library to your application's library search path
- ◆ Registered the objects you want to check out

## Checking out objects

You can check out a registered object at any time by selecting it in the Library painter.

❖ **To check out registered objects:**

- 1 In the Library painter, select one or more registered objects that you want to check out.
- 2 Check out the object or objects you want.

**To check out a group of objects** Click the Check Out button on the PainterBar *or* select Source>Check Out from the menu bar.

**To check out a single item** Right-click the object and select Check Out from the popup menu.

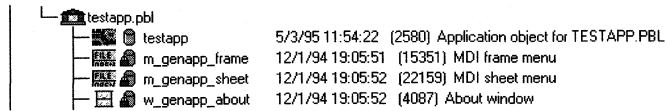
The Check Out Library Entries dialog box displays.

- 3 Select the work library to check out the objects to.

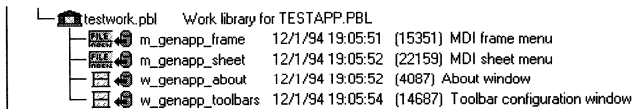
What happens

PowerBuilder does the following:

- ◆ Makes a working copy of each selected object and stores it in the work library you specified
- ◆ Locks this version of the object so no one else can check it out
- ◆ In the public library, assigns a lock icon to the object showing that the object is locked:



- ◆ In the work library, assigns a check-out icon to the object indicating that it is the working copy:



---

### Your user ID

If you checked out files in PowerBuilder before using the Source Integrity interface, you specified a PowerBuilder-specific ID. When you use the Source Integrity interface, PowerBuilder synchronizes this ID with the Source Integrity user ID (LOGNAME) stored in the configuration file.

---

## Viewing the status of checked-out objects

At any time, you can view the status of objects checked out from libraries in the search path of the current application. The status information displayed for each checked-out object includes:

- ◆ The ID of the user who checked out the object
- ◆ The location of the public library from which the object was checked out
- ◆ The location of the work library to which the object was checked out

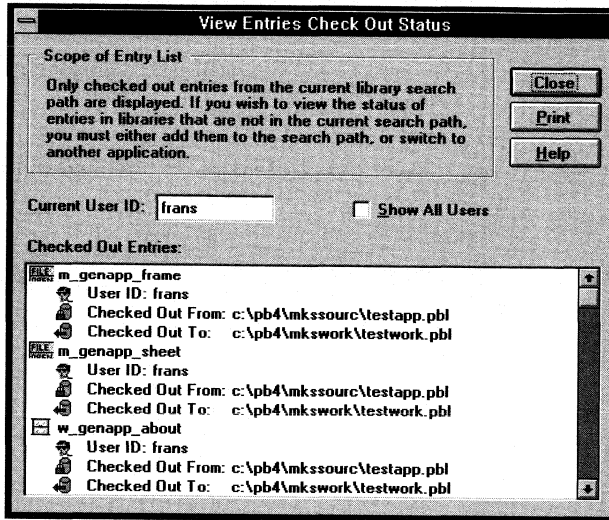
❖ **To view the status of checked-out objects:**

- 1 In the Library painter, click the Check Status button.

*or*

Select Source>View Check Out Status from the menu bar.

The View Entries Check Out Status dialog box displays showing the status of objects you checked out (in this example, the objects were checked out by frans):



- 2 To see the status of objects checked out by *all* users, select the Show All Users checkbox.
- 3 To print the status, click the Print button.  
The printed version contains the contents of the Checked Out Entries listbox.
- 4 Click Close to close the dialog box.

## Modifying checked-out objects

When you need to modify an object, you should check out the object to your work library and make changes only to the checked-out version. This ensures that no one else can modify the same object while you are working on it.

❖ **To open a checked-out object to modify it:**

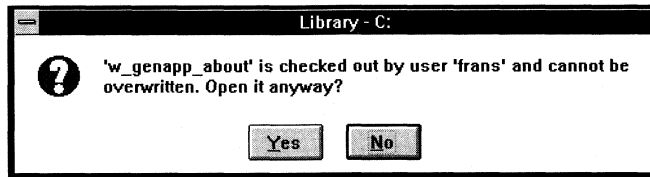
- ◆ In the Library painter, double-click the checked-out object in your work library that you want to modify.

The associated PowerBuilder painter opens and displays the selected object.

---

**You cannot overwrite objects checked out by someone else**

If you try to open an object checked out by someone else, a message box similar to the following displays:



Click Yes if you want to open the object. You will be able to view the object but you will *not* be able to save changes to it.

---

## Checking in objects to Source Integrity

When using the Source Integrity interface, you should check in objects as follows:

- ◆ **Your objects** Check your objects into the public library *as soon as you finish modifying them*. This ensures that all developers will have access to the updated objects.
- ◆ **All objects** Check all objects into the public library *before a major build of the application executable*. This ensures that PowerBuilder uses the most recent versions of the objects to build the new executable.

### ❖ To check in objects:

- 1 In the Library painter, select one or more registered objects that you want to check in.

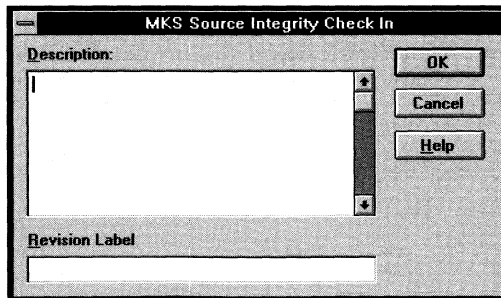
If you select a group of objects to check in, a separate MKS Source Integrity Check In dialog box displays for *each* object you select.

- 2 Check in the object or objects you want.

**To check in a group of objects** Click the Check In button on the PainterBar *or* select Source>Check In from the menu bar.

**To check in a single object** Right-click the object and select Check In from the popup menu.

The MKS Source Integrity Check In dialog box displays:



- 3 Type a description in the Description box.

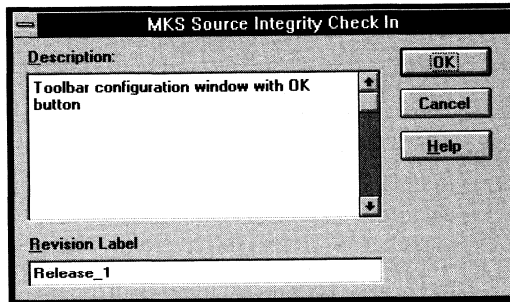
The description will appear in the registration directory listing for this object when you display its change history. It will also appear in the revision report.

- 4 (Optional) Type a revision label in the Revision Label box.

This associates the revision label with the new revision in the archive.

FOR INFO See "Using revision labels" on page 132 for considerations when assigning revision labels.

Here is a sample MKS Source Integrity Check In dialog box with a description and revision label specified:



- 5 Click OK to check in the objects.

What happens

PowerBuilder does the following:

- ◆ Copies the object in the work library to the public library.
- ◆ Deletes the working copy of the object from your work library.
- ◆ Increments the object's revision number. For example, if the revision number of the object is 1.0 when you check it out, it becomes 1.1 when you check the object back in.
- ◆ Unlocks the object so other developers can check it out. (PowerBuilder removes the lock icon for this object in the Library painter.)

## Creating a new release

You can create a new release at any point in the development of your PowerBuilder application.

### When to do this

Typically, you create a new release when you reach a development milestone and want to copy your libraries and objects to a separate place while preserving your revision history.

### Choices

PowerBuilder gives you some choices when you build a new release. You can:

- ◆ Set options for the new archive
- ◆ Specify a starting revision number for the release

## About setting options for the new archive

You can set either of the following options for the new archive when you create a release:

- ◆ **Start With Fresh Archive** (Default) This option copies only the most current revision level to the new set of archives. Use this option if you want to take a single revision level from which to create the new archive.
- ◆ **Maintain Existing Revisions** This option copies the entire revision history in its current state to the new archive. Use this option if you want to preserve the existing revision history of the current application and migrate it to the new archive.

## About specifying a starting revision number

You can also specify a starting revision number for the new archive if you want. When you create a new release, PowerBuilder will maintain the most current revision number of each object unless you specify a new starting revision number.

For example, if you specify 2.0 as the starting revision number, your new release will have the most current revisions of the objects begin with revision number 2.0. Source Integrity will then assign revision numbers beginning with 2.0, such as 2.1, 2.2, and so on.

## Creating the release

You create a new release of your application by completing the MKS Source Integrity New Release dialog box.

---

### Before you start

Make sure that all objects have been checked in to ensure that you start with an accurate revision history.

**FOR INFO** For instructions on checking in objects, see "Checking in objects to Source Integrity" on page 127.

---

### ❖ To create the new release:

- 1 In the Library painter, select Source>Create New Release from the menu bar.

The MKS Source Integrity New Release dialog box displays:

| Archives:             |                       |
|-----------------------|-----------------------|
| Old Archive Directory | New Archive Directory |
| c:\pb4\mkssourc       |                       |

| Libraries:                  |                  |
|-----------------------------|------------------|
| Old Library Name            | New Library Name |
| c:\pb4\mkssourc\testapp.pbl |                  |
| c:\pb4\mkswork\testwork.pbl |                  |

- 2 Click one of the following radio buttons in the New Archive Options groupbox:
  - ◆ **Start With Fresh Archive** (Default) Copies only the most current revision level to the new set of archives.
  - ◆ **Maintain Existing Revisions** Copies the entire revision history in its current state to the new archive.
- 3 (Optional) Specify a starting revision number in the Starting Revision box.



- 4 Type a new archive directory for each old archive directory listed.  
If you specify the name of a new directory, PowerBuilder prompts you before creating it.
- 5 Type a new library name for each old library name listed.  
If you specify the names of existing libraries, PowerBuilder prompts you before overwriting them.
- 6 Click OK.  
PowerBuilder creates the new release in the archives and libraries you specified.

After you create the new release

In order to make changes to the objects in your new release, you must:

- ◆ Make the new application the current application by opening it in the Application painter. (For how, see the *PowerBuilder User's Guide*)
- ◆ Add the new libraries to the new application's library search path. (For how, see the *PowerBuilder User's Guide*)
- ◆ Define a Source Integrity configuration file for the new application. (For how, see "Defining the configuration file" on page 110)

## Using revision labels

A **revision label** is a descriptive name that identifies revisions stored in an archive. A revision label stays associated with its revision even after you check in new revisions.

### Why use

You can use revision labels to group the revisions that comprise a particular version of an application so you can select them quickly and easily. For example, you might assign the revision label *Release\_1* to all objects belonging to this release.

When you use the Project painter to build a project, the Source Integrity interface lets you assign revision labels as part of the build process. In this way, you can use revision labels as an alternative to creating a new release by copying libraries, as described in "Creating a new release" on page 129.

### Rules

The following rules apply when you use revision labels with the Source Integrity interface:

| Rule  | Comment   |
|---|---|
| Assign revision labels to objects                           | <p><i>You can</i> assign the same revision label to any number of objects stored in different archives</p> <p><i>You can</i> assign multiple revision labels to a single revision of an object</p> <p><i>You cannot</i> assign the same revision label to more than one revision of an object in the same archive</p>                                   |
| Specify a revision label for the Source Integrity interface | <p>The revision label must be a contiguous string of letters, digits, or underscores ( _ ) with no spaces or periods</p> <p>The first character of the revision label must be a letter or underscore ( _ )</p> <p>The revision label is case sensitive. For example, <i>Release_1</i> and <i>release_1</i> are considered different revision labels</p> |

## Assigning revision labels to a group of objects

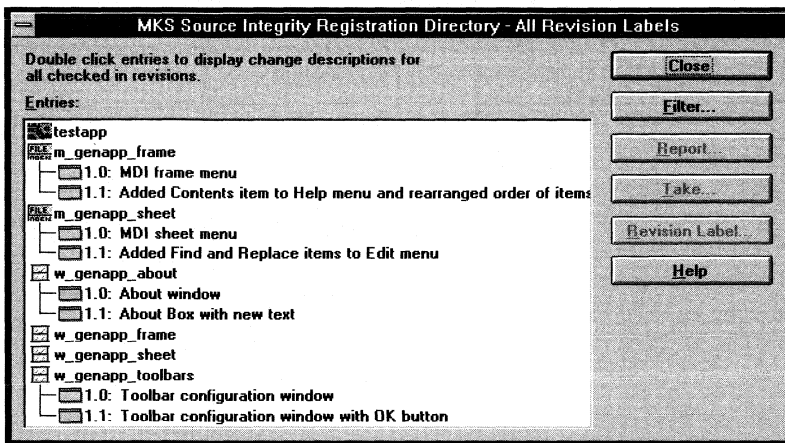
Typically you assign a revision label to a group of objects rather than to a single object.

❖ **To assign a revision label to a group of objects:**

- 1 In the Library painter, click the Directory button.  
or  
Select Source>Registration Directory from the menu bar.

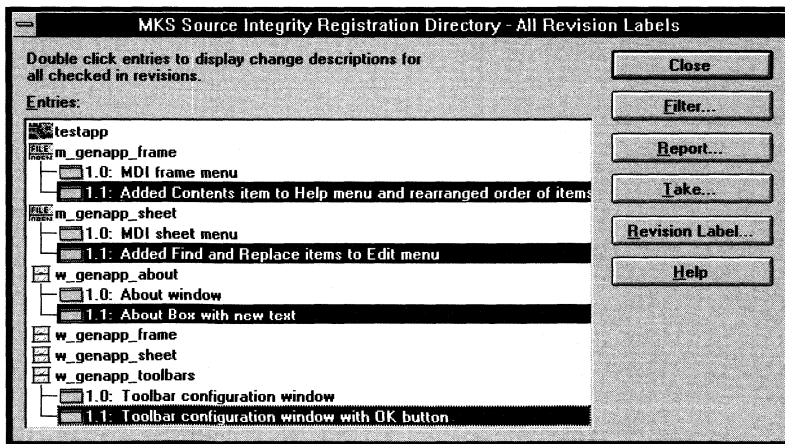
The MKS Source Integrity Registration Directory dialog box displays.

- 2 Double-click the objects for which you want to display revisions.  
The revision history displays for each selected object:



- 3 Select the revisions to which you want to assign the same revision label.  
Press CTRL+click to select noncontiguous revisions.

In this example, all revisions with a 1.1 revision number are selected:



- 4 Click the Revision Label button.

The MKS Source Integrity Assign Revision Label dialog box displays.

- 5 Type a revision label in the Revision Label box, following the rules in the previous table.

- 6 Click OK.

PowerBuilder assigns the revision label to the selected revisions.

❖ **To check the revision label assignment:**

- ◆ To check the results of the revision label assignment, you can do either of the following:

- ◆ Display the archive report for any of the objects containing revisions with the revision label you specified. The archive report includes the revision label.

FOR INFO See "Displaying an archive report" on page 141.

- ◆ Use the revision label to filter the list of revisions in the Registration Directory dialog box.

FOR INFO See "Filtering revision lists by revision labels" next.

## Filtering revision lists by revision labels

You can filter the list of revisions in the Registration Directory dialog box by a particular revision label. This lets you display the change history for only those objects containing revisions with this revision label.

❖ **To filter a list of revisions by revision label:**

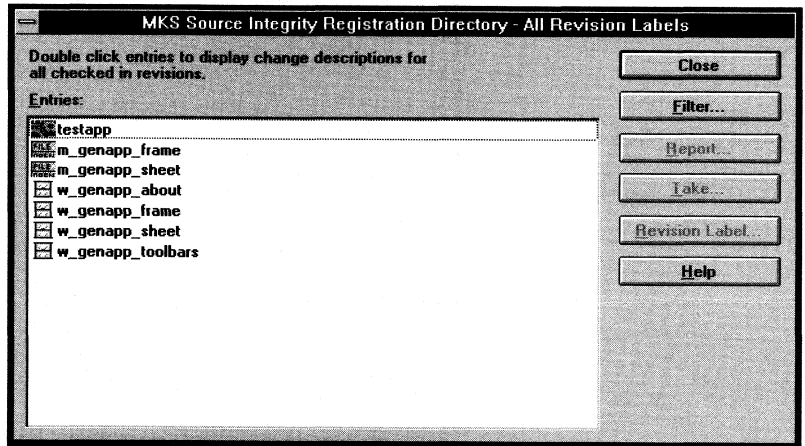
- 1 In the Library painter, click the Directory button.

or

Select Source>Registration Directory from the menu bar.

The MKS Source Integrity Registration Directory dialog box displays listing all the registered objects in your current application.

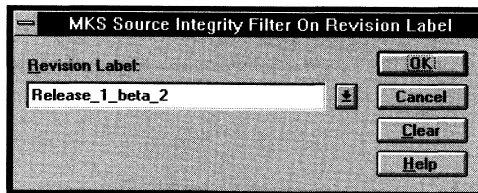
Initially *All Revision Labels* displays in the title bar of the dialog box. This indicates that all objects are displayed even if they have different revision labels assigned. After you filter the list, the title bar changes to show the revision label you are filtering by:



- 2 Click the Filter button.

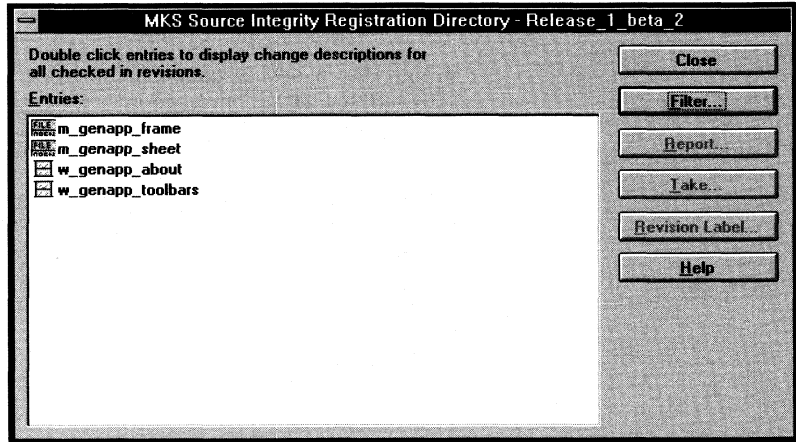
The MKS Source Integrity Filter On Revision Label dialog box displays.

- 3 Type the revision label you want to filter by in the Revision Label box.



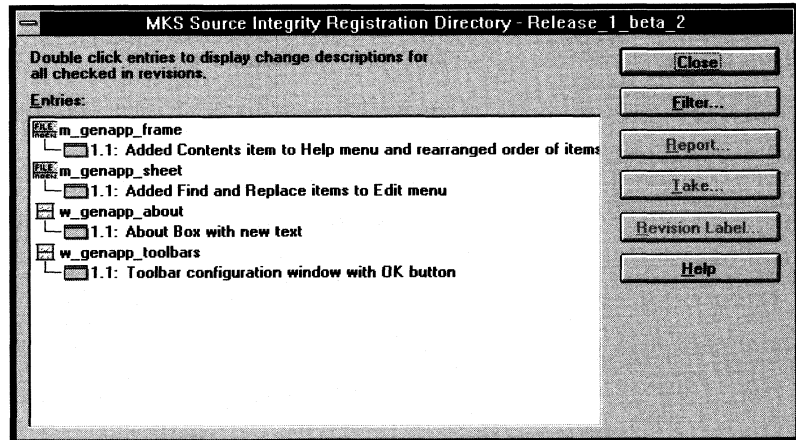
- 4 Click OK.

The Registration Directory dialog box displays showing only those objects containing revisions with the revision label you specified. Notice that *Release\_1\_beta\_2* displays in the title bar to indicate the revision label you are filtering by:



- 5 Double-click an object to display its revision history.

Only those revisions having the revision label you specified display (in this example, the revision label *Release\_1\_beta\_2* was assigned to all 1.1 revision numbers):



## Clearing the filter

When you close the Registration Directory dialog box, PowerBuilder clears the filter automatically so that the next time you open the Registration Directory dialog box, all objects display. However, you may want to clear the filter *without* closing the Registration Directory dialog box.

❖ **To clear the filter without closing the Registration directory dialog box:**

- 1 Click the Filter button in the Registration Directory dialog box.

The MKS Source Integrity Filter On Revision Label dialog box displays with the revision label filter you specified.

- 2 Do either of the following:

- ◆ Click the Clear button.
- ◆ Select the revision label text (if it is not already selected) and press the DELETE key to clear it. Then click OK.

The Filter On Revision Label dialog box closes and all objects display in the Registration Directory dialog box.

## Assigning revision labels when you build a project

When you use the Project painter to build a project that includes objects under Source Integrity's control, you can specify a revision label for all of the objects in the project. This is useful if you want to assign a revision label as part of the build process.

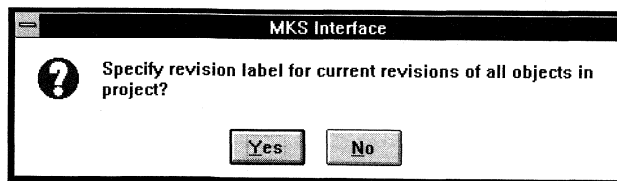
❖ **To assign a revision label when you build a project:**

- 1 Open the project you defined in the Project painter.

FOR INFO For instructions on using the Project painter, see the *PowerBuilder User's Guide*.

- 2 Click the Build button in the Project painter bar.

Before the application executable is built, the following message box displays:



- 3 Click Yes to assign a Source Integrity revision label to your objects.  
The MKS Source Integrity Assign Revision Label dialog box displays.
- 4 Type the revision label you want in the Revision Label box, following the rules described in "Using revision labels" on page 132.  
PowerBuilder builds the executable and assigns the revision label you specified to all objects in your project.



## Viewing an object's revision history

You can view a registered object's revision history at any time by displaying the MKS Source Integrity Registration Directory dialog box. A **revision** is a saved version of an object under Source Integrity's control.

Each time you check an object into its archive, it is saved as a new revision and Source Integrity increments the revision number. For example, if you check out an object with a revision number of 1.0, its revision number becomes 1.1 when you check it back in.

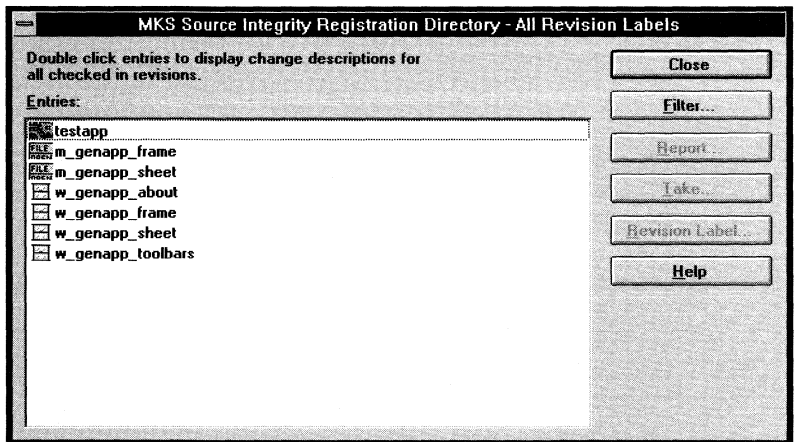
### ❖ To view an object's revision history:

- 1 In the Library painter, click the Directory button.

*or*

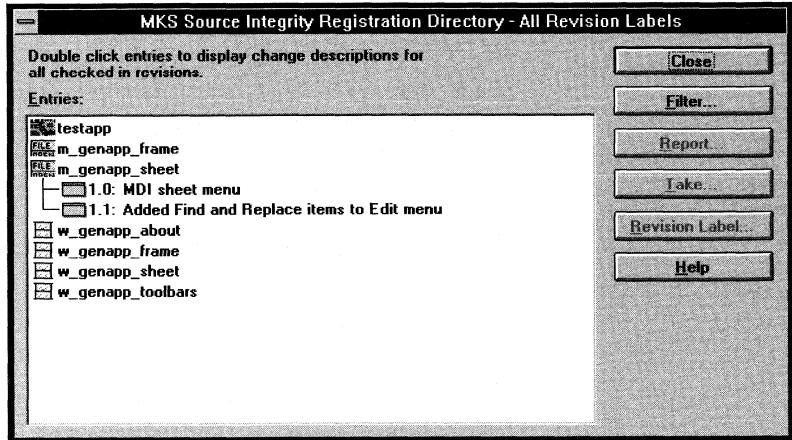
Select Source>Registration Directory from the menu bar.

The MKS Source Integrity Registration Directory dialog box displays listing all the registered objects in your current application:



- 2 Double-click any object to view its revision history.

PowerBuilder displays all revisions for the selected object:



## Displaying reports

You can display two types of reports using the Source Integrity interface. These reports give you more information about the archives and revisions in your application.

- ◆ **Archive report** Describes the entire revision history of an object's archive since its registration
- ◆ **Revision report** Describes one specific revision of an object

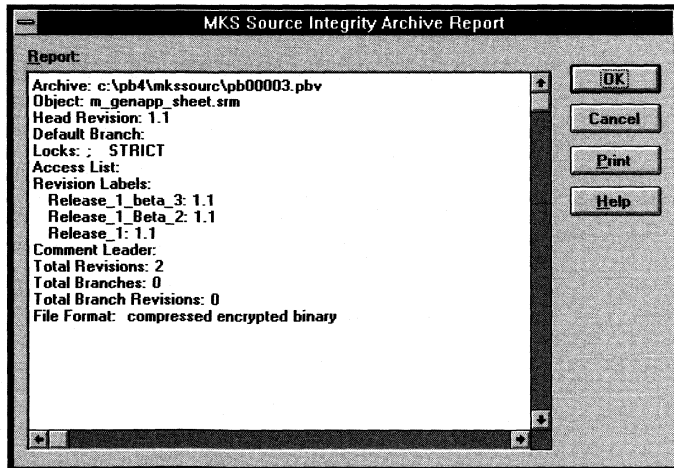
## Displaying an archive report

You can display an object's archive report at any time by selecting the object in the Library painter or in the MKS Source Integrity Registration Directory dialog box. You can view the report, print it, or copy it to a file.

### ❖ To display an archive report from the Library painter:

- 1 Right-click the name of the registered object for which you want a report.
- 2 Select Registration Report from the popup menu that displays.

The MKS Source Integrity Archive Report dialog box displays:



**FOR INFO** For more about the types of information in the archive report, see your Source Integrity documentation.

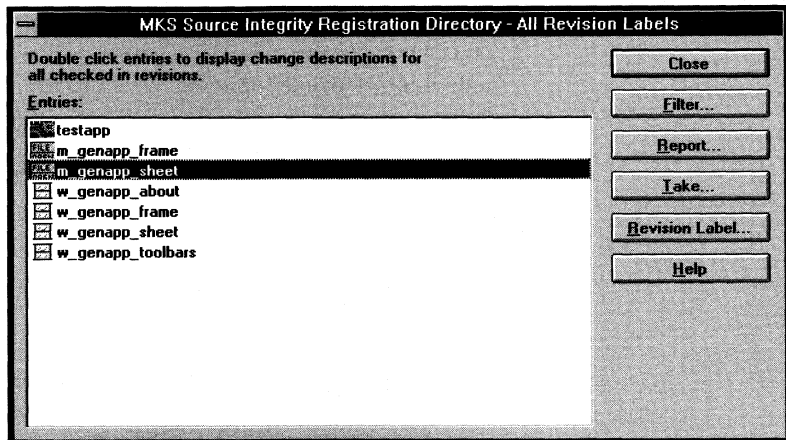
- 3 Click the Print button if you want to print the archive report on your default printer.
- 4 Click OK to close the dialog box.

❖ **To display an archive report from the Registration Directory dialog box:**

- 1 In the Library painter, click the Directory button.  
*or*  
Select Source>Registration Directory from the menu bar.

The MKS Source Integrity Registration Directory dialog box displays listing all the registered objects in your current application.

- 2 Select the name of an object.



- 3 Click the Report button.  
The MKS Source Integrity Archive Report dialog box displays.  
FOR INFO For more about the types of information in the archive report, see your Source Integrity documentation.
- 4 Click the Print button if you want to print the archive report on your default printer.
- 5 Click OK to close the dialog box.

## Displaying a revision report

You can display a revision report at any time by selecting the revision in the MKS Source Integrity Registration Directory dialog box. You can view the report, print it, or copy it to a file.

❖ **To display a revision report:**

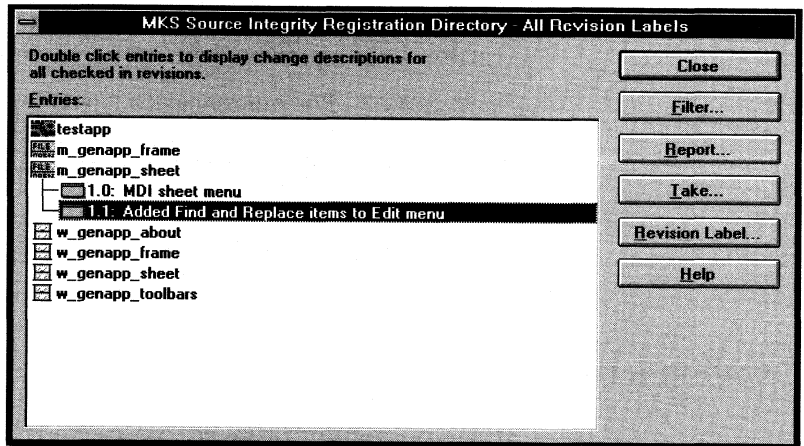
- 1 In the Library painter, click the Directory button.

*or*

Select Source>Registration Directory from the menu bar.

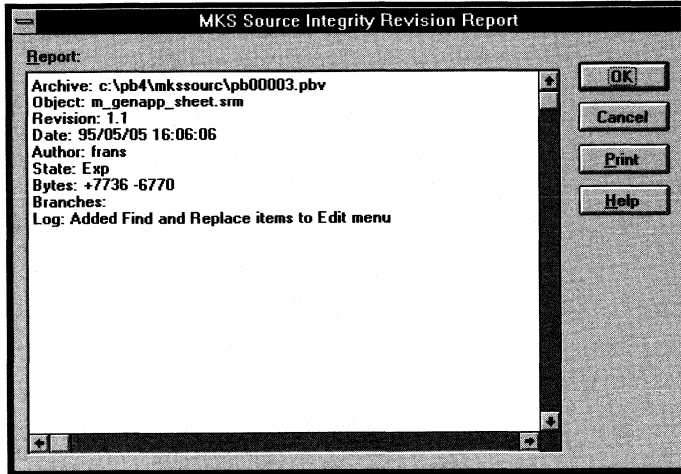
The MKS Source Integrity Registration Directory dialog box displays listing all the registered objects in your current application.

- 2 Double-click an object to view a list of its revisions.
- 3 Select the revision for which you want a report.



- 4 Click the Report button.

The MKS Source Integrity Revision Report dialog box displays:



FOR INFO For more about the types of information in the revision report, see your Source Integrity documentation.

- 5 Click the Print button if you want to print the archive report on your default printer.
- 6 Click OK to close the dialog box.

## Copying a report to a file

If you want, you can copy the text of an archive or revision report to a file.

### ❖ To copy the text of an archive or revision report to a file:

- 1 In the Archive Report or Revision Report dialog box, select the text you want to copy from the Report box.
- 2 Press CTRL+C to copy the selected text to the clipboard.
- 3 Open a file in a text editor of your choice.
- 4 Press CTRL+V to paste the text from the clipboard into the file.
- 5 (*Optional*) Save the file for future reference.

## Restoring earlier revisions of an object

A primary reason for putting your PowerBuilder objects under version control is to enable you to restore earlier revisions of an object. The Source Integrity interface makes this possible.

### Deciding which revision to restore

The first step in restoring an earlier revision of an object is to decide which revision you want to restore. There are several items you can display in the Source Integrity interface to help you decide which revision you want.

#### Use descriptive comments

These methods work best when you carefully comment each revision as you check it back into the archive.

| To see  | Display   | For information, see                               |
|---|---|--|
| A list of all revisions for this object                                   | The object's revision history in the MKS Source Integrity Registration Directory dialog box | "Viewing an object's revision history" on page 139 |
| The entire revision history of an object's archive since its registration | An archive report for an object   | "Displaying an archive report" on page 141         |
| Information about one revision of an object                               | A revision report for a specific revision   | "Displaying a revision report" on page 143         |

### Restoring an earlier revision

Once you decide which revision of an object you want to restore, perform the restoration from the MKS Source Integrity Registration Directory dialog box.

❖ **To restore an earlier revision of an object:**

- 1 Check out the current revision of the object you want to restore to your work library.

FOR INFO For instructions, see "Checking out objects from Source Integrity" on page 123.

- 2 In the Library painter, click the Directory button.

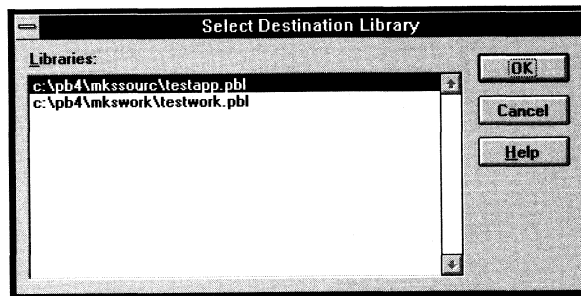
*or*

Select Source>Registration Directory from the menu bar.

The MKS Source Integrity Registration Directory dialog box displays listing all the registered objects in your current application.

- 3 Double-click an object to view a list of its revisions.
- 4 Select the revision that you want to restore.
- 5 Click the Take button.

The Select Destination Library dialog box displays:



- 6 Select a destination library in the Libraries box. (Typically, this is your work library.)
- 7 Click OK.

PowerBuilder copies the selected revision of the object over the current revision and regenerates the object.

- 8 Check the restored revision back into the archive.

FOR INFO For instructions, see "Checking in objects to Source Integrity" on page 127.

The restored revision becomes the current revision of the object.



## Restoring revisions by revision labels

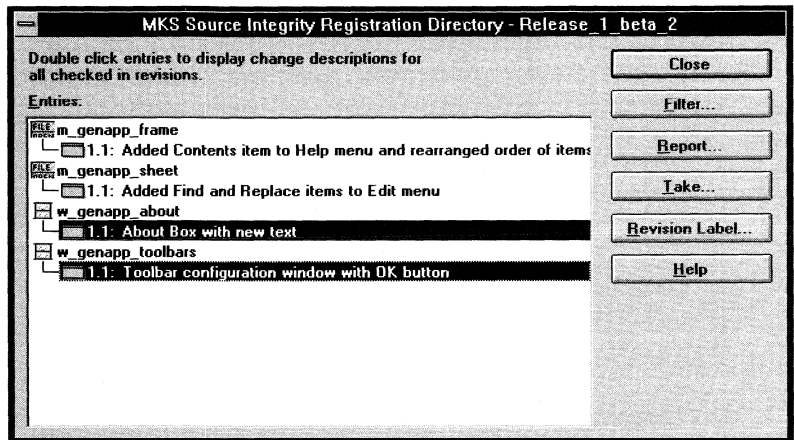
To restore only those revisions having a particular revision label, you filter the list of objects by the revision label and then select the objects you want from the resulting list.

### ❖ To restore revisions by revision label:

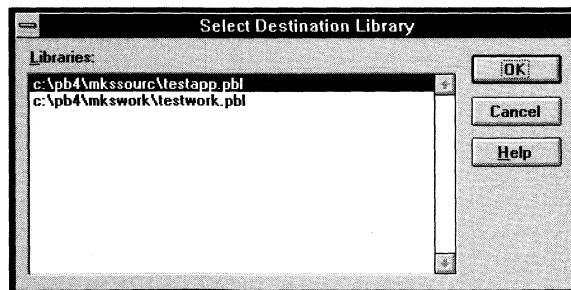
- 1 In the MKS Source Integrity Registration Directory dialog box, click the Filter button to filter by the revision label you want.

FOR INFO For instructions, see "Filtering revision lists by revision labels" on page 134.

- 2 Double-click an object to display the list of revisions for that object.



- 3 Select the revision you want from each object and click the Take button. The Select Destination Library dialog box displays:



- 4 Select a destination library in the Libraries box and click OK.  
PowerBuilder puts the selected revisions into the library you specified.

## Restoring libraries

When you use the Source Integrity interface, you can restore earlier versions of PowerBuilder libraries (PBLs) by selecting Design>Restore Libraries from the Project painter menu bar.

What PowerBuilder does

When you restore libraries, PowerBuilder puts your objects into specified libraries. After you restore your libraries to their new location, you can rebuild the application executable in the Project painter.

Project object

Restoring libraries is possible because of the **project object**. Each time you build an executable in the Project painter, PowerBuilder saves information about the application in the project object. You can use this information later to restore your libraries. PowerBuilder stores the following information about objects in each project object:

- ◆ PowerBuilder object names with PBL name
- ◆ Archive names and revision numbers
- ◆ Revision labels

Each time you build an executable, PowerBuilder updates the project object with the most recent version of this information.

## Listing the objects in a project

After you build your project in the Project painter, you can display a list of the objects in the project.

❖ **To list the objects in a project:**

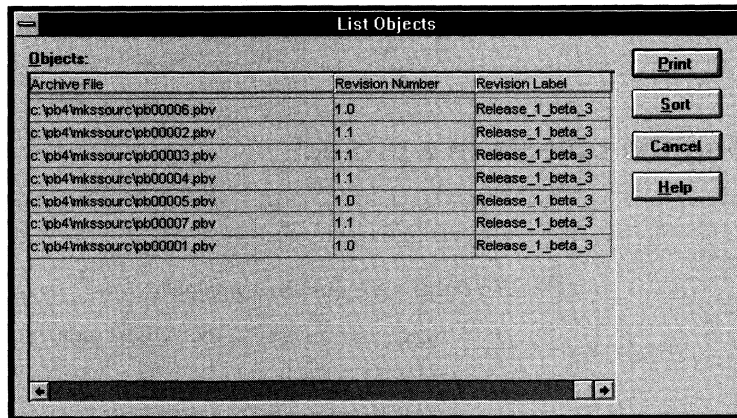
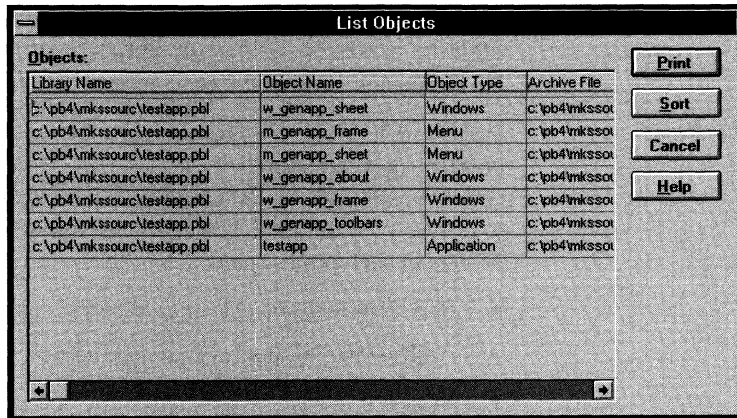
- 1 Build your project in the Project painter.

FOR INFO For instructions on using the Project painter, see the *PowerBuilder User's Guide*.

- 2 Select Design>List Objects from the menu bar.

The List Objects dialog box appears displaying a report of the objects in your project and how they map to objects in your archives.

This example shows two views of the *same* List Objects dialog box in order to display all the columns in the report:



FOR INFO For a description of each column, see "What's in the report" next.

- 3 (Optional) Click the Print button to send the report to your default printer, or click the Sort button to specify the order in which the objects are listed.
- 4 Click Cancel to close the dialog box.

What's in the report

The report in the List Objects dialog box includes the following columns:

| Column          | What it shows  |
|-----------------|--|
| Library Name    | Source library containing the object   |
| Object Name     | Name of the object   |
| Object Type     | Type of object   |
| Archive File    | Name of the archive file containing this object                                |
| Revision Number | Unique number (such as 1.0 or 1.1) that identifies the revision in the archive |
| Revision Label  | Descriptive name that identifies the revision in the archive                   |

What you can do

You can do the following when viewing the List Objects report:

- ◆ Scroll horizontally to see the columns
- ◆ Resize the columns
- ◆ Reorder the columns
- ◆ Print the report
- ◆ Sort the rows in the report

## About the methods for restoring libraries

Because PowerBuilder stores information about the objects in the project object, you can restore your libraries by using either of the following methods:

| Method   | What you do   | Uses Source Integrity? |
|--|---|------------------------|
| Retrieve an earlier revision of the project object from Source Integrity | Use Source Integrity to track changes to the project object   | Yes                    |
| Save the project object with a new name                                  | Each time you build an executable or change the project object definition, select File>Save As in the Project painter menu bar to save the object with a new name in the current library list | No                     |

## Retrieving the project object from Source Integrity

|             |   |
|-------------|---|
| When to use | Use this method if you: <ul style="list-style-type: none"><li>◆ Need to be able to restore any revision level of your project object at any time</li><li>◆ Expect to have many revision levels of your project object</li></ul> |
| Benefits    | Putting your project object under Source Integrity's control is very secure. In addition, this method creates only a single project object in your library listing since Source Integrity takes care of tracking changes to it. |
| Strategy    | Use descriptive comments and version labels when handling project objects under Source Integrity's control to help you determine which project object you want to retrieve.   |

## Saving the project object with a new name

|                   |   |
|-------------------|---|
| When to use       | Use this method if you plan to save a small number of project objects that you may want to restore from.  |
| Benefit           | Saving a project object with a new name is a very simple method to use. To open an earlier version of a project object, you simply double-click it in the Library painter listing. In this way, you have access to the project objects you create each time you build an executable or change the project object definition.  |
| Possible drawback | However, for large, complex development projects with many intermittent builds, this method may not meet your needs. Using this method could create more project objects in your library than you can easily manage.  |
| Strategies        | <p><b>Create a separate library for your project objects</b> If you use this method, you might want to create a separate library to hold project objects that you create each time you build an application executable or change the definition of a project object. This makes it easier to find a project object when you want to open it and restore libraries.</p> <p><b>Use descriptive comments</b> Be sure to provide descriptive comments for your project objects so you can easily identify them.</p> |

## Retrieving the project object from Source Integrity

The project object stores information about the objects, archives, revision numbers, and revision labels in your application. Therefore, when you put the project object under Source Integrity's control, you can use its revision history to restore libraries used to build earlier versions of your application.

### ❖ To restore libraries by retrieving the project object from Source Integrity:

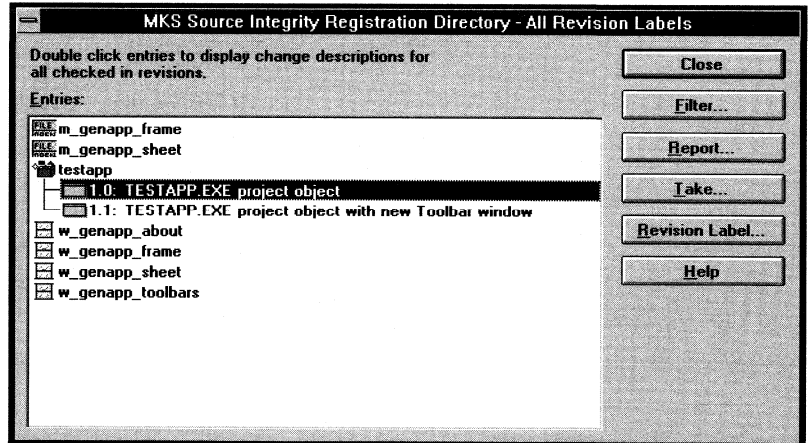
- 1 Create and save the initial version of the project object in the Project painter.

FOR INFO For instructions on using the Project painter, see the *PowerBuilder User's Guide*.

- 2 Register the project object with Source Integrity.

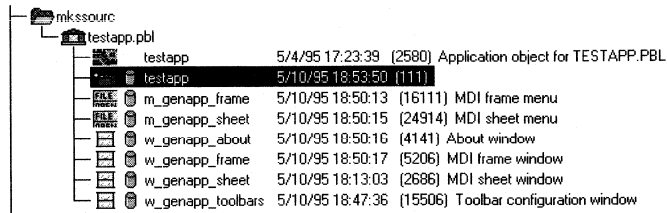
FOR INFO For instructions, see "Registering PowerBuilder objects" on page 118.

- 3 Click the Take button in the MKS Source Integrity Registration Directory dialog box to retrieve an earlier revision level of the project object.



FOR INFO For instructions, see "Restoring earlier revisions of an object" on page 145.

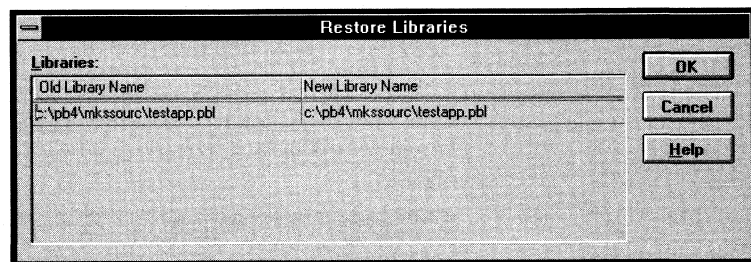
- In the Library painter, double-click the project object you retrieved.



The Project painter opens and displays the selected project object definition.

- Select Design>Restore Libraries from the Project painter menu bar.

The Restore Libraries dialog box displays:



- Type a new library (PBL) name for each old library name listed.

If you specify the names of new libraries, PowerBuilder will create them. If you specify the names of existing libraries, PowerBuilder prompts you before overwriting them.

- Click OK.

PowerBuilder restores the libraries to the state they were in when you built the application executable.

## Saving the project object with a new name

If you decide *not* to put your project objects under Source Integrity's control, you can save the project object with a new name each time you build an executable or change the project object definition. You then open an earlier version of the project object to restore your libraries.



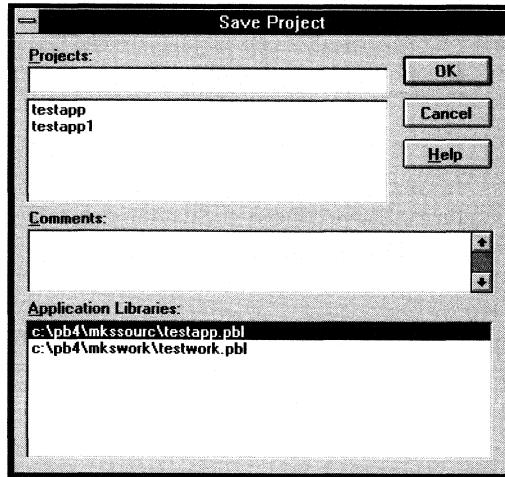
❖ **To restore libraries by saving the project object with a new name:**

- 1 In the Project painter, build the project.

FOR INFO For instructions on using the Project painter, see the *PowerBuilder User's Guide*.

- 2 Select File>Save As from the menu bar.

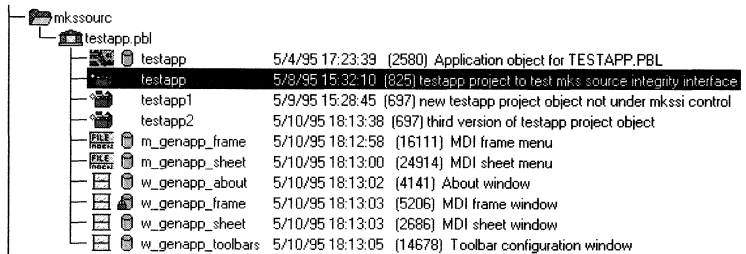
The Save Project dialog box displays:



- 3 Specify the new name for the project object and a comment describing it, and click OK.

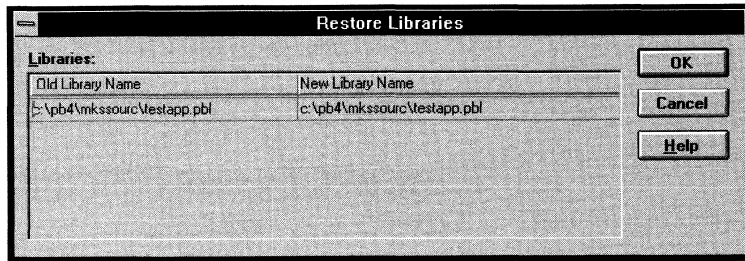
PowerBuilder saves the project object with the new name.

- 4 In the Library painter, double-click the project object you want to open.



The Project painter opens and displays the selected project object definition.

- 5 Select Design>Restore Libraries from the Project painter menu bar.  
The Restore Libraries dialog box displays:



- 6 Type a new library (PBL) name for each old library name listed.  
If you specify the names of new libraries, PowerBuilder will create them. If you specify the names of existing libraries, PowerBuilder prompts you before overwriting them.
- 7 Click OK.  
PowerBuilder restores the libraries to the state they were in when you built the application executable.

## Synchronizing objects

PowerBuilder maintains information about your libraries separately from Source Integrity. A Source Integrity archive is updated only when you perform an operation that affects the archive such as registering, checking out, or checking in an object. Therefore, you may need to resynchronize your PowerBuilder libraries periodically with the Source Integrity archives.

When to synchronize

You need to synchronize if one of the following events desynchronizes objects in your PowerBuilder libraries:

- ◆ Unexpected system failures
- ◆ System crashes that occur as you register, check out, or check in an object
- ◆ Changes made to an archive outside PowerBuilder

What synchronizing does

The Source Integrity interface lets you synchronize registered objects to ensure that the objects in your PowerBuilder library are the same as the latest revision of the objects stored in the Source Integrity archives.

❖ **To synchronize registered objects:**

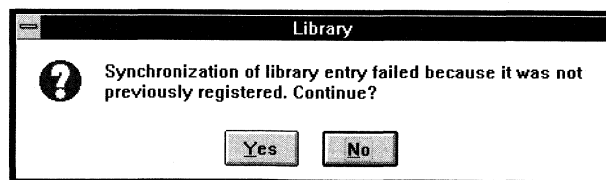
- 1 In the Library painter, select one or more registered objects that you want to synchronize.
- 2 Select Source>Synchronize from the menu bar.

PowerBuilder regenerates the objects from their definitions in the Source Integrity archives.

---

### **Objects that you synchronize must be registered**

The objects that you select to synchronize must be registered with Source Integrity. If you select an unregistered object, the following message box displays and PowerBuilder will *not* synchronize the object:





# Using the PowerBuilder ENDEVOR Interface

**About this chapter**

This chapter describes how to use the PowerBuilder ENDEVOR interface to manage the objects in your PowerBuilder applications.

**Contents**

| <b>Topic</b>                   | <b>Page</b> |
|--------------------------------|-------------|
| About the ENDEVOR interface    | 160         |
| Installation                   | 161         |
| Invoking the interface         | 167         |
| Using version control          | 168         |
| Cross-referencing object names | 174         |
| Running reports                | 175         |

**Source of information**

The information in this chapter was provided by Legent, now part of Computer Associates.

## **About the ENDEVOR interface**

The PowerBuilder ENDEVOR interface lets you control versions of PowerBuilder objects in a repository managed by ENDEVOR Workstation. The version control functionality is accessed through the PowerBuilder Library painter.

### **System requirements**

The interface requires the following:

- ◆ ENDEVOR Workstation release 5.2.2 or 5.2.3
- ◆ PowerBuilder Version 6.0 or higher
- ◆ 50K available disk space in the directory containing PowerBuilder
- ◆ 800K available disk space in the directory containing the ENDEVOR executables and databases

## Installation

❖ **To install the PowerBuilder ENDEVOR interface:**

- 1 Install and configure PowerBuilder and ENDEVOR.
- 2 Install the interface software.
- 3 Edit the repository components:
  - ◆ Define and authorize users.
  - ◆ Change the repository location.
  - ◆ Modify naming conventions.
- 4 Create the repository.

These steps are described below.

---

### **Reporting problems**

If you have any questions or problems during the installation of the PowerBuilder ENDEVOR interface, contact Computer Associates Technical Support.

**FOR INFO** If you have any questions or problems while using the interface, contact Sybase Technical Support.

---

## **Step 1: install and configure PowerBuilder and ENDEVOR**

Install PowerBuilder and the Powersoft-supplied components of the ENDEVOR interface.

**FOR INFO** For how to install these components, see Chapter 1, "Preparing to Use Version Control with PowerBuilder".

Then:

- 1 Install ENDEVOR, following instructions in your ENDEVOR documentation.
- 2 Have each user execute the configuration program. Information about this program is in your ENDEVOR documentation.

## Step 2: install the PowerBuilder ENDEVOR interface

Install the Computer Associates–supplied components of the PowerBuilder ENDEVOR interface.

The ENDEVOR part of the installation process uses two programs, in the order stated: INSTALL and ENPOWCFG. You run INSTALL for each copy of ENDEVOR and ENPOWCFG for each copy of PowerBuilder. For example:

- ◆ If both ENDEVOR and PowerBuilder are installed on your LAN, run INSTALL first, then run ENPOWCFG. The same procedure is true if both ENDEVOR and PowerBuilder are installed locally.
- ◆ If ENDEVOR is installed on the LAN and PowerBuilder is installed on local drives, run INSTALL first and then have each PowerBuilder user run ENPOWCFG.

Typically the interface is installed once on the network. However, the interface install program can be run as often as necessary. If a particular interface file has already been copied, the program asks you to specify Y (overwrite) or N (do not overwrite) before it will proceed.

### ❖ To use the INSTALL program:

---

#### Important

If you already have a copy of POW.BTR and have registered objects in PowerBuilder, bypass this procedure (do *not* overwrite POW.BTR) until you have a chance to go into PowerBuilder and clear (unregister) any objects that are registered.

---

- ◆ At the DOS prompt, name the drive containing the interface diskette and type **install**. For example, if you are using the A drive, type:

```
a:install
```

The program does the following:



- 1 Copies POW.BTR to your ENDEVOR database directory.
- 2 Overwrites the message database (MSG.BTR) with one that includes interface-specific messages.
- 3 Copies the following files to the ENDEVOR executables directory:  
BTRWNDVR.DLL, PBPRINT.BAT, ENPOW301.DLL,  
ENPWKRNL.EXE, PRTPOW.EXE, PBNDV060.DLL, PBINTF.SCL,  
README2.TXT, ENPOWCFG.EXE
- 4 Asks if you want to change the default names for the repository. We recommend that you do not change the names. If you indicate you want to make changes, the install program prompts you for each value.
- 5 Copies ENPOW.INI to your database directory. This file includes the ENDEVOR repository definition.
- 6 Gives follow-up instructions for editing the PowerBuilder PB.INI file and the ENDEVOR PBINTF.SCL file.

❖ **To use the ENPOWCFG program:**

- 1 Become current on the directory containing your ENDEVOR executables and type **enpowcfg**. For example, if your executables are in L:\APPS\ENDEVOR, type:

```
l:\apps\endevor\enpowcfg
```

- 2 Give the location of the PowerBuilder executables directory. The program copies the following files to that directory:

```
ENPOW301.DLL  
PBNDV060.DLL
```

If the program cannot find PowerBuilder executables in the directory you specify, you are presented with the following options:

- ◆ R (Re-type the directory name)
- ◆ Y (Use the specified directory)
- ◆ X (Exit the install procedure; the interface is not installed until you rerun the install program)

### Step 3: edit the repository definitions

The repository is defined in the PBINTF.SCL file that the interface INSTALL program copied to your ENDEVOR executables directory. This file contains definitional statements in ENDEVOR's Software Control Language (SCL).

❖ **To edit the repository definitions:**

- ◆ Make the following changes to the SCL file:
  - 1 Identify your site's PowerBuilder users in the Define Users section. Assign those users to the PB-USER security class.
  - 2 (Optional) Change the repository location.
  - 3 (Optional) Edit the repository names if you made changes during the install process.

## Defining and authorizing users

The SCL defines a single user (PBUSER1) and a security class (PB-USER) that specifies the minimum permissions needed to do version control under PowerBuilder. The SCL also assigns user PBUSER1 to the PB-USER security class.

You need to:

- 1 Open the PBINTF.SCL file in a text editor.
- 2 Create a Define User statement for developers not currently defined to ENDEVOR who plan to use the ENDEVOR-PowerBuilder interface.

```
define user name
description "descriptive-text"
[ password password ].
```

For example:

```
define user PWRUSER
description "Power User".
```

Passwords are optional and can always be added at another time.

- 3 Authorize the users you have just defined, by adding their names to the end of the Security Class definition, repeating the *authorize user* statement for each new user. You may also want to add the names of previously defined ENDEVOR users who are being assigned to the PowerBuilder security class.

The Security Class statement is:

```
define security class PB-USER
description "PowerBuilder Version Control"
permit delete element
define element
display
retrieve
```

```

signin
signout
add
authorize user PBUSER1
authorize user another-user

```

## Changing the repository location

This is an optional step. By default, the repository directories are saved in NDVRCFG, a variable maintained by ENDEVOR. (The value of NDVRCFG is saved in ENDEVOR.INI, which was created when you configured your workstation.)

To change the location of the repository, you need to edit the image and delta directory paths in the Define Stage statement as shown below:

```

define stage pb-stg
description "Repository staging area"
to environment pb-env
system pb-sys
subsystem pb-sub
type *
image directory
"&NDVRCFG&pb-env\pb-stg"
delta directory
"&NDVRCFG&pb-env\pb-stg\delta" .

```

For example, the image and delta files could be located in a directory on the L: drive called PB:

```

image directory
"l:\pb\pb-env\pb-stg"
delta directory
"l:\pb\pb-env\pb-stg\delta" .

```

## Modifying naming conventions

This step is optional. You should use the default values, particularly if you are a new ENDEVOR user. The default values are:

```

Environment = PB-ENV
System = PB-SYS
Subsystem = PB-SUB
Stage = PB-STG

```

If you make changes, you should do a global search and replace so that all instances of a given name are changed.

---

**If you make changes**

When changing an environment, system, subsystem, or stage name, you must modify *two* files: ENPOW.INI, which saves your responses from the interface install process, and PBINTF.SCL, which you run to define the repository.

*Never modify the type names.* The names of the predefined PowerBuilder types must be defined exactly as they appear in the PBINTF.SCL file.

---

## Step 4: define the repository to ENDEVOR

Change to the ENDEVOR executables directory and issue the following command:

```
ndvr pbintf.scl pbintf.log
```

This command invokes ENDEVOR batch and submits the SCL you edited in step 3. PBINTF.LOG records the SCL statements that were parsed and flags syntax errors, if any.

## Invoking the interface

### ❖ To invoke the interface:

- 1 Start PowerBuilder and open the Library painter.

After the workspace is displayed, PowerBuilder displays the ENDEVOR Logon dialog box.

---

#### **If the ENDEVOR dialog box does not appear**

If you don't see the ENDEVOR dialog box, select Source>Connect from the Library painter menu bar and select ENDEVOR.

---

- 2 Type the ID of a user authorized to use the PowerBuilder interface (for example, PBUSER1).
- 3 Click OK.

The following table shows the version control functionality available from within PowerBuilder:

| Function               | Description  |
|------------------------|--|
| Register               | Identifies the object being placed under version control. Typically this action is performed once for each object, but objects can be re-registered if necessary |
| Register Clear         | Deletes all levels of object from the repository. Removes object name from Registration Directory  |
| Check Out              | Copies a registered object into your work PBL. The working copy is the only one you can edit   |
| View Check Out Status  | Lists objects that are checked out, specifying the location of the working copies and users who have the objects checked out                                     |
| Check In               | Checks in the object, creating a new level. Updates the registered object and deletes the working copy   |
| Clear Checkout         | Checks in the object. Optionally, deletes the working copy   |
| Registration Directory | Lists all objects in the repository. To view all change levels of an object, double-click on the object name   |
| Take                   | Replaces the working copy with a previous level of an object   |
| Synchronize            | Replaces the registered copy in PowerBuilder with the current level in ENDEVOR   |

## Using version control

Version control under PowerBuilder allows you to:

- ◆ Register objects
- ◆ Display all objects and their histories in the repository
- ◆ Check out objects for editing
- ◆ Display check out status for all objects
- ◆ Check in objects
- ◆ Cancel changes to objects (clearing check-out status)
- ◆ Delete objects (clearing registration)
- ◆ Revert to a previous object level
- ◆ Re-register objects
- ◆ Synchronize objects

You initiate these tasks from the Source menu in the Library painter, as described below.

## Registering objects

You can register a PowerBuilder object anytime. After you register an object you can check it out of the public library into your work library, modify the object as needed, and check the object back into the public library for safekeeping.

❖ **To register an object:**

- 1 Highlight the object name.
- 2 Select Source>Register from the menu bar.

If you did not previously supply a PowerBuilder description for the object, you are prompted for one at this time.

**FOR INFO** For information about re-registering objects, see "Re-registering objects" on page 172.

## Displaying object histories

You can view a registered object's version history at any time by displaying the ENDEVOR Registration Directory dialog box. A **version** is a saved *version* of an object under ENDEVOR's control.

Each time you check an object into its archive, it is saved as a new version and ENDEVOR increments the version number. For example, if you check out an object with a version number of 1.0, its version number becomes 1.1 when you check it back in.

❖ **To list and view histories of registered objects:**

- 1 In the Library painter, select Source>Registration Directory from the menu bar to list all registered objects.
- 2 (*Optional*) Double-click the listed object to view histories of registered objects.

---

**How objects are found**

The library list associated with an application determines where an interface looks for registered objects. If your registered objects come from PBLs that are not included in that list, they are not listed in the Registration Directory.

---

❖ **To modify a library list:**

- 1 In the Application painter, select Entry>Properties from the menu bar.
- 2 Select the Libraries tab.
- 3 Click Browse and navigate to the library you want to add.
- 4 Click Open to add that library to the list.
- 5 Repeat steps 3 and 4 to add any additional libraries to the list.
- 6 Click OK.

## Checking out objects

After you register an object with the ENDEVOR interface, you must check it out to modify it. When you check out an object, PowerBuilder locks this version of the object so no one else can check it out while you are working on it. Then you can edit the object in the appropriate PowerBuilder painter.

❖ **To check out objects for editing:**

- 1 Highlight the name of the registered copy.
- 2 Select Source>Check Out from the menu bar.
- 3 When prompted, specify the name of your work PBL. (Make sure the work PBL is in the application's library list. Otherwise you will not be able to check in the edits.)

---

**Caution**

Do not move or delete a working (checked-out) copy from its work PBL. Either action corrupts the information shared between PowerBuilder and ENDEVOR.

**FOR INFO** For how to cancel changes made to an object, see "Clearing object check-out status" on page 171. For how to delete a registered object from the archive, see "Deleting objects" on page 171.

---

## Displaying object check-out status

❖ **To display check-out status for objects:**

- 1 Select Source>View Check Out Status from the menu bar.  
This shows all the objects you checked out.
- 2 (Optional) Click the Show All Users option in the Status dialog box.  
This shows the check-out status for all the objects others have checked out.

## Checking in objects

Once you have modified an object you can preserve your changes by checking in the object to the archive.

❖ **To check in an object:**

- 1 Highlight the name of the working (checked-out) copy.



- 2 Select Source>Check In from the menu bar.

ENDEVOR asks you for a new description and creates a new level for the object. (If there are no changes between the working copy and the registered copy, ENDEVOR does not create a new level and discards the new description.)

After ENDEVOR processes the update (creates a new level and its delta), PowerBuilder updates the registered object and deletes the working copy.

## Clearing object check-out status

When you cancel an object's check-out status, you cancel any changes made to that object.

### ❖ To cancel changes to an object:

- 1 Highlight the object name (either the registered or working copy).
- 2 Select Source>Clear Check Out from the menu bar.
- 3 Respond to the prompt about deleting the working copy. We recommend deleting the working copy. This ensures that you do not try to edit an object that is not checked out (and therefore cannot be updated).

## Deleting objects

If the object you want to delete is checked out, clear the check-out status of that object before deleting it.

FOR INFO For how to do this, see "Clearing object check-out status" above.

You can remove an object from ENDEVOR's control by clearing its registration.

### ❖ To delete an object from the repository:

- 1 Highlight the name of the registered object name.
- 2 Select Source>Clear Registration from the menu bar.

The object is dropped from the Registration Directory and deleted from the repository.

## Reverting to a previous level

One of the main reasons for putting your PowerBuilder objects under version control is to enable you to restore earlier versions of an object. The ENDEVOR interface makes this possible.

❖ **To revert to a previous level:**

- 1 Check out the registered copy of the object.
- 2 Open the Registration Directory and double-click on the object name to list the available levels.
- 3 Select the level that you want to restore, and click Take.
- 4 When prompted, specify the PBL that contains the working copy you checked out in step 1.

The level you have just taken overwrites the working copy.

- 5 Check in the working copy.

The level you have restored is now the current (registered) level.

## Re-registering objects

**How re-register works**

When you re-register an object, the PBL from which you registered it becomes the object's home PBL, and the object in the former PBL is no longer registered.

If there is a difference between the copy of the object being re-registered and the previously registered copy, ENDEVOR creates a new level that contains the contents of the re-registered object.

**When to re-register**

You should re-register if you can see an object in the Registration Directory list but cannot check it out or if you want to check out an object with the same name as an object in a different PBL that is already registered (only one object of a given name can be registered and checked out).

❖ **To re-register an object:**

- 1 Highlight the name of the registered object.
- 2 Select Source>Register from the menu bar.

- 3 Answer Yes when asked if you want to re-register and create a new level.

As in the original registration, ENDEVOR uses the first 40 characters of the PowerBuilder description as the level descriptor.

## Synchronizing objects

PowerBuilder maintains information about your libraries separately from ENDEVOR. An ENDEVOR archive is updated only when you perform an operation that affects the archive such as registering, checking out, or checking in an object. Therefore, you may need to resynchronize your PowerBuilder libraries periodically with the ENDEVOR archive.

### When to synchronize

You need to synchronize if one of the following events desynchronizes objects in your PowerBuilder libraries:

- ◆ Unexpected system failures
- ◆ System crashes that occur as you register, check out, or check in an object
- ◆ Changes made to an archive outside PowerBuilder (such as in the ObjectCycle Manager)

### What synchronizing does

The ObjectCycle interface lets you synchronize registered objects to ensure that the objects in your PowerBuilder library are the same as the latest version of the objects stored in the ObjectCycle archive.

If the object in PowerBuilder is out of sync with the one in ENDEVOR, you can resynchronize the PowerBuilder library with the ENDEVOR archive.

#### ❖ To synchronize an object:

- 1 Highlight the name of the registered object.
- 2 Select Source>Synchronize from the menu bar.

The synchronization is a one-way operation (from ENDEVOR to PowerBuilder).

## Cross-referencing object names

When you register PowerBuilder objects, ENDEVOR generates its own unique 8-character name for the object.

To print a report showing the object name and its corresponding ENDEVOR name, run the PBPRINT.BAT program that is in your ENDEVOR executables directory.

You need to supply the location of POW.BTR to the program as follows:

```
pbprint location-of-pow.btr
```

For example, if POW.BTR is in c:\apps\endevor\db, type the following command at the DOS prompt:

```
pbprint c:\apps\endevor\db\pow.btr
```

PBPRINT initiates the PRTPOW utility that generates the report and sends the information to your screen.

---

### **Sending the report to a file**

To send the report to a file, add a redirect command to the PRTPOW command on the second line of the PBPRINT file. For example, to redirect to a file called NAMES.RPT, edit the line as follows:

```
prtpow %1 > names.rpt
```

---

## Running reports

To run reports, invoke ENDEVOR Workstation and use the Reports menu to generate the reports about the objects being managed.

**FOR INFO** For details on available reports, see the ENDEVOR Workstation documentation.

---

### **Use ENDEVOR Workstation only to run reports**

ENDEVOR Workstation should be used only to run reports or to display object history/changes. Performing actions such as Move, Signout, or Signin on a PowerBuilder object outside of PowerBuilder (that is, from within ENDEVOR Workstation) corrupts the information maintained by the interface.

---



# Using the PowerBuilder Apple SourceServer Interface

## About this chapter

This chapter describes how to configure and use the PowerBuilder Apple SourceServer interface to manage the objects in your Macintosh PowerBuilder for Macintosh applications.

## Contents

| <b>Topic</b>                                     | <b>Page</b> |
|--|-------------|
| Overview of the Apple SourceServer interface     | 178         |
| Installing Apple SourceServer                    | 179         |
| Connecting to Apple SourceServer                 | 180         |
| Configuring the Apple SourceServer interface     | 181         |
| Modifying registered objects                     | 188         |
| Assigning and using version labels               | 191         |
| Using object histories and running reports       | 193         |
| Restoring an earlier revision level of an object | 195         |
| Synchronizing objects                            | 196         |

## Before you begin

Make sure you've installed Apple SourceServer before you try to use the PowerBuilder Apple SourceServer interface. Each developer who will use Apple SourceServer needs both Apple SourceServer itself and the PowerBuilder Apple SourceServer interface.

## Overview of the Apple SourceServer interface

The PowerBuilder Apple SourceServer interface lets you manage source on your Macintosh without having to run Apple SourceServer. The Apple SourceServer source control features display in the Library painter, where your file management activities take place.

**FOR INFO** See Chapter 1, "Preparing to Use Version Control with PowerBuilder" for general version control information.

### Managing your objects

Once you have installed the Apple SourceServer interface and configured your environment, you can start managing your objects. Source control activities available with the Apple SourceServer interface include:

- ◆ Registering objects
- ◆ Checking objects out and in and modifying objects
- ◆ Assigning and using version labels
- ◆ Using object histories and running reports
- ◆ Restoring earlier revisions of objects
- ◆ Synchronizing your objects with Apple SourceServer archives



## Installing Apple SourceServer

Before you can use the PowerBuilder interface for Apple SourceServer, you must install the interface as well as the Apple SourceServer program.

**FOR INFO** For information about installing Apple SourceServer itself, see the documentation that comes with Apple SourceServer. For information on using the Installer to install the interface, see the *PowerBuilder Installation Guide*.

## Connecting to Apple SourceServer

After you install Apple SourceServer, you must connect to it when you next open the Library painter. After that, PowerBuilder will automatically connect to Apple SourceServer each time you open the Library painter.

❖ **To connect to Apple SourceServer:**

- 1 In the Library painter, select Source>Connect from the menu bar.
- 2 In the Connect dialog box, choose SourceServer from the popup menu, then click OK.

Before you've configured an application

If the current application hasn't been put under source control when you connect to Apple SourceServer, PowerBuilder prompts you to set up a configuration file for it.

**FOR INFO** For information on configuring Apple SourceServer, see "Setting up an application configuration file" on page 181.

If you have problems

To confirm that Apple SourceServer is running, check that it appears on the Macintosh application menu or that its status window appears on the desktop.

If PowerBuilder fails to connect to Apple SourceServer, check that:

- ◆ Apple SourceServer has been installed
- ◆ The Apple SourceServer Interface file has been installed in the System Folder:Shared Libraries:Extensions:Powersoft 6.0 folder

## Configuring the Apple SourceServer interface

Before developers can check out and check in objects, you must perform some configuration tasks to put each application under source control. There are also configuration tasks for each developer to perform who will work on the application.

What the project manager does

You (the project manager) will set up each application for source control by:

- ◆ Setting up a configuration file with your user ID and a new project folder for the archive database
- ◆ Registering the objects in the application's public libraries

What each developer does

Each developer working on the application will:

- ◆ Configure an individual work environment by:
  - ◆ Setting up an individual configuration file that specifies a unique user ID and points to the project folder you created
  - ◆ Creating one or more work libraries and putting them on the application's library search path
- ◆ Register objects created in the course of development

## Setting up an application for source control

For each application, you will create a configuration file that stores information on your user ID, the database location, the project folder, and the Apple SourceServer database (called ProjectorDB) you will use. Then you will register the application's objects.

### Setting up an application configuration file

The first step in setting up an application is creating the configuration file and the project folder with its archive database.

Developers will each have a copy of the configuration file with a unique user ID. Each developer on the project needs access to the project folder.

The configuration file is associated with the application object. When a developer switches to another application, PowerBuilder uses its configuration file to find the archive database for that application.

❖ **To set up an application configuration file:**

- 1 In the Application painter, open the application to be configured.
- 2 Open the Library painter, then create a new configuration file or modify an existing one:
  - ◆ To create a configuration file, click Yes at the prompt.
  - ◆ To modify a configuration file, select Source>Configuration from the menu bar.
- 3 Fill in or modify the Apple SourceServer Configuration dialog box:

| <b>Box</b>         | <b>Action</b>  |
|--------------------|--|
| User               | Type the user ID you will use for source control<br><hr/> <b>Your Apple SourceServer user ID</b><br>PowerBuilder will use this user ID when you use the Apple SourceServer interface. If you checked out files under another user ID in PowerBuilder before using Apple SourceServer, that user ID is ignored. <hr/> |
| Project path       | Click New, then navigate to the folder where the project files are to be stored<br><br>Type the name of the new archive database in the Save SourceServer Database As box, then click Save   |
| Configuration file | Click New, then navigate to the folder where you want the configuration files saved<br><br>Type the name of the new configuration file in the Save Configuration As box, then click Save   |

- 4 Click Done to complete the configuration.

## Registering application objects

Once you have created the project folder with its archive database, you can register your objects. Developers can check out registered objects to make changes to them, then check them back in to the archive for safekeeping. You will register the objects in the public libraries for the application. All developers will need access to these public libraries.

---

### To take advantage of Apple SourceServer




To take full advantage of Apple SourceServer and to avoid problems, you should register *all* objects in *all* your application libraries.

---

❖ **To register objects:**

- 1 In the Library painter, find the library containing the objects you want to register.
- 2 Double-click the library to display the objects, and select the objects you want to register.
- 3 Select Source>Register from the menu bar and then fill in the Apple SourceServer Checkin dialog box for each selected object.

PowerBuilder assigns a registration icon to each object:

|   |  |
|---|--|
|  | pbexamfe.pbl Front-end library for PowerBuilder 6.0 Examples. Includes applic: |
|  | exampl60 3/6/96 13:21:49 (5333) PowerBuilder 6.0                               |
|  | d_categories 3/6/96 13:21:50 (3524)  |

---

### To view a registered object without making any changes

You can view a registered object by opening the object for read-only access. Do not check out the object.

---

## Configuring an individual work environment

Each developer needs to configure an individual work environment by:

- 1 Installing Apple SourceServer.  
**FOR INFO** For information about installing Apple SourceServer itself, see the Apple SourceServer documentation. For information on installing the interface, see the *PowerBuilder Installation Guide*.
- 2 Connecting to Apple SourceServer, as described in "Connecting to Apple SourceServer" on page 180.
- 3 Configuring each application to use the archive database, already set up by the project manager, by:
  - ◆ Creating a configuration file that identifies an individual user ID and specifies the location of the project folder
  - ◆ Creating one or more work libraries to hold the checked-out versions of objects the developer is working on

- ◆ Allowing access to the public libraries for the application (these libraries contain the registered objects). This is a networking issue that the developers must work out within the development group
- ◆ Modify the library search path to include their individual work library.

FOR INFO For how to modify the library search path, see "Creating work libraries" on page 185.

Once the work environment is configured, developers will need to check existing objects out and in, and register new objects.

FOR INFO For information on checking objects out and in, see "Checking objects in to Apple SourceServer" on page 190. For information on registering new objects, see "Registering application objects" on page 182.

## Setting up an individual configuration file

Each developer needs a configuration file for each application. The file contains the developer's user ID and points to the project folder containing the archive database of registered objects for the application.

### ❖ To set up an individual configuration file:

- 1 In the Application painter, open the application to be configured for Apple SourceServer.
- 2 Open the Library painter and then create a new configuration file or modify an existing one:
  - ◆ To create a configuration file, click Yes.
  - ◆ To modify a configuration file, select Source>Configuration from the menu bar.
- 3 Fill in or modify the Apple SourceServer Configuration dialog box:

| Box  | Action   |
|------|--|
| User | Type the user ID you will use for source control   |
|      | <b>Your Apple SourceServer user ID</b><br>PowerBuilder uses this user ID when you use the Apple SourceServer interface. If you checked out files under another user ID in PowerBuilder before using Apple SourceServer, that user ID is ignored. |

| Box                | Action   |
|--------------------|--|
| Project path       | Click Browse, then navigate to the ProjectDB file within the project folder set up for the application<br>Select the ProjectDB file and click Open                                   |
| Configuration file | Click New, then navigate to the folder where you want the configuration files saved<br>Type the name of the new configuration file in the Save Configuration As box, then click Save |

- 4 Click Done to complete the configuration.

## Creating work libraries

After checking out objects, you put them into a work library where working copies of objects are stored. Usually an application's public libraries reside on a network drive, accessible to everyone working in the development group. Each developer will have a local work library.

### ❖ To create a work library:

- 1 In the Library painter, click the Create button in the PainterBar.  
*or*  
Select Library>Create from the menu bar.
- 2 In the Create Library dialog box, navigate to where you want to locate your work library.
- 3 Specify a filename for your work library, then save it, with a comment describing its purpose (for example, that it is a work area for objects from a particular archive).
- 4 Click OK to create the library.
- 5 Register the library.

**FOR INFO** For information on registering objects, see "Registering application objects" on page 182.

When you create a work library, you also need to add that library to the library search path PowerBuilder uses to find the libraries your application references.

### ❖ To modify your library search path:

- 1 In the Application painter, select Entry>Properties from the menu bar.
- 2 Select the Libraries tab.
- 3 Locate the new library and select the file.

- 4 Click Add.  
PowerBuilder adds the library to your library list.
- 5 Repeat steps 2 and 3 to add any additional libraries.
- 6 Click Done.

Compiling with  
objects from your  
work library

To build a new application executable using objects that are checked out to your work library, your work library must be listed *before* your archive libraries in the library search path dialog box. During the compile process, PowerBuilder uses the first instance of the objects the application references. If the archive library is before the work library in your library search path, PowerBuilder will find and use the archive library versions of the objects.

**FOR INFO** For information on setting up libraries, see the *PowerBuilder User's Guide*.

---

**To move your work library to the top of the list**

Remove libraries that should appear after your work library, add your work library, then add the libraries that should follow.

**FOR INFO** For how to modify your library search path, see the procedure above.

---

## Viewing a list of registered objects

Any time you need to see what is registered, you can view the list of registered objects.

❖ **To view a list of registered objects:**

- ◆ Click the Directory button.  
*or*  
Select Source>Registration Directory from the menu bar.

## Clearing an object's registered status

To remove an object from Apple SourceServer control, you clear its registration. PowerBuilder deletes the archive file, and any object history is lost.



---

**Before deleting a registered object from a library**

You must clear an object's registration before you delete that object from the library.

---

❖ **To clear an object's registration:**

- 1 In the Library painter, select the object.
- 2 Select Source>Clear Registration from the menu bar, and click Yes to confirm.

## Modifying registered objects

Once objects are registered in Apple SourceServer, you can:

- ◆ View the check-out status of archived objects
- ◆ Check an object out of the archive
- ◆ Modify an object
- ◆ Check an object back in to the archive

**FOR INFO** For information on check-out and check-in, see the library management discussion in the *PowerBuilder User's Guide*.

## Viewing a list of checked-out objects

Before checking an object out, you can verify its checked-out status.

- ❖ **To view a list of checked-out objects:**
  - ◆ Select Source>View Check Out Status from the menu bar.

## Checking out objects

Checking out a registered object locks the archive version so that no one else can check out that object. You can only check out objects to the work libraries defined in your library search path.

- ❖ **To check out objects:**

- 1 Select the objects to check out.

**To check out one object** CONTROL+click the object and then select Check Out from the popup menu.

**To check out more than one object at a time** Select the objects from the archive library list. Then click the Check Out button in the PainterBar *or* select Source>Check Out from the menu bar.

- 2 In the Check Out Library Entries dialog box, select the destination work library from the library list and click Open.

The destination library is where you will save the working copy of the object or objects you are checking out. If you specify a library not in the current application's library search path, PowerBuilder won't be able to save the working copy.

When you check out an object, PowerBuilder:

- ◆ Makes a working copy of each selected object and stores it in the destination library you specified
- ◆ Locks the object in the archive so no one else can check it out
- ◆ In the public library, assigns a lock icon to the object showing that the object is locked:

```

pbexamfe.pbl Front-end library for PowerBuilder 6.0 Examples. Includes ap
  exempl60          3/6/96 13:21:49 (5333) PowerBuilder
  d_categories     3/6/96 13:21:50 (3524)
  d_event_list     3/6/96 13:21:51 (43482)
  d_example_detail_dddw 3/6/96 13:21:52 (2993)
    
```

- ◆ In the work library, assigns a check-out icon to the object indicating that it is the working copy:

```

worklib.pbl Work library for checked-out objects.
  d_event_list     3/6/96 13:21:51 (43482)
  d_example_detail_dddw 3/6/96 13:21:52 (2993)
    
```

---

### To build an executable using checked-out objects

Be sure that your work library is listed *before* your archive library in your library search path. Otherwise, PowerBuilder will use the objects it finds in your archive instead of the objects in your work library.

---

## Modifying objects

You should always modify the checked-out versions of objects, to be sure that no one else is modifying the same objects. When you open an object that someone else has checked out, you get a read-only copy.

- ❖ **To open an object you have checked out for modification:**
  - ◆ Double-click the checked-out object in your work library.  
The painter associated with that object will also open.

## Checking objects in to Apple SourceServer

It is best to check in objects as follows:

- ◆ **Your objects** Check in your objects as soon as you have finished modifying them. This gives other developers access to them.
  - ◆ **All objects** Check in all objects before a major build of the application executable. This will ensure that the most recent versions are used to build the new executable.
- ❖ **To check objects back in to the archive:**
- 1 Select objects for checking in.  
  
**To check in one object** CONTROL+click the object and then select Check In from the popup menu.  
  
**To check in more than one object at a time** Select the objects from your work library list. Then click the Check In button on the PainterBar or select Source>Check In from the menu bar.
  - 2 Fill in the Apple SourceServer Check In dialog box and click OK.  
  
The comment you type will appear in the Apple SourceServer archive report and the Apple SourceServer revision report.

What happens

When you check in an object, PowerBuilder:

- ◆ Replaces the object in the original library with the working copy and deletes the working copy from your work library.
- ◆ Increments the object's revision level. For example, if the object's revision level is 2 when you check it out, when you check it back in it receives a revision level of 3.
- ◆ Unlocks the archive version of the object so that other developers can check it out.

## Assigning and using version labels

A version label is a symbolic name identifying a group of revisions of objects stored in separate archives. You can use version labels to associate the revisions that comprise a version of an application so that you can select them quickly and easily. A version label remains associated with a revision even after you check in new revisions.

Keep in mind that you:

- ◆ Can assign the same version label to any number of revisions of objects stored in different archives
- ◆ Can assign multiple version labels to a single revision
- ◆ Cannot assign the same version label to more than one revision of an object in an archive

### Assigning a version label to a group of objects

Normally you will want to assign a version label to a group of objects, not to an individual object.

❖ **To assign a version label to a group of objects:**

- 1 Select Source>Registration Directory from the menu bar.
- 2 In the Registration Directory dialog box, double-click each object whose revision levels you want to display.
- 3 When the change history displays for each object, use **COMMAND+click** to select each of the revisions you want to assign the same version label.
- 4 Click the Version Label button and type a version label.

---

**To check version label assignments**

Display an archive report for any of the archives that contain revisions you assigned a version label. Or use the filter option in the Registration Directory dialog box.

---

### Filtering a list of revisions by version label

You can use version labels as a filter to retrieve specific objects.

❖ **To filter by version label and (optionally) retrieve filtered objects:**

- 1 Select Source>Registration Directory from the menu bar.
- 2 In the Registration Directory dialog box, click Filter.
- 3 In the Filter On Version Label dialog box, type the version label.

The Registration Directory dialog box displays, showing only the objects that contain revisions with the version label you specified.

- 4 Double-click an object name to show the revision that has the version label you specified as the filtering argument.

- 5 *(Optional)* To retrieve objects from the filtered list:

- ◆ Select the revision you want for each object and click Take.
- ◆ Select a destination library in the Select Destination Library dialog box.

## Using object histories and running reports

You can get an overview of an object's change history, or generate a report to detail either a specific revision level or the full change history.

### Getting an overview of an object's change history

❖ **To get an overview of an object's change history:**

- 1 Click the Directory icon.  
*or*  
Select Source>Registration Directory from the menu bar.
- 2 In the Apple SourceServer Registration Directory dialog box, double-click the object whose change history you want to see.
- 3 Click Close when you are finished.  
The Registration Directory dialog box closes and you return to the Library painter.

### Generating change history reports

You can generate two detailed reports about archives and revisions:

- ◆ **Archive report** Shows object's entire change history
- ◆ **Revision report** Shows object's specific revision level

### Generating a report

You can generate either a full archive report or a specific revision report, and copy the text of a report into a text file.

❖ **To generate a report:**

- 1 Select Source>Registration Directory from the menu bar.  
The Registration Directory dialog box displays.
- 2 Select the information for the report you want to generate:
  - ◆ **Archive report** Select an object name.
  - ◆ **Revision report** Double-click an object name and select the revision level you want.

3 Click Report.

**FOR INFO** For information about the categories presented in an archive or revision report, see your Apple SourceServer documentation.

---

**To copy text from a report to a file**

You can copy text from a report into a file. To do this, select the text you want to copy and press **COMMAND+C** to copy the selected text to the clipboard. Then, using a text editor, paste the text from the clipboard into a file by pressing **COMMAND+V**.

---



## Restoring an earlier revision level of an object

Putting your objects under source control permits the restoration of earlier revision levels of your objects.

❖ **To restore a previous revision level of an object:**

- 1 Check out the current revision of the object.

**FOR INFO** For instructions, see "Modifying registered objects" on page 188.

- 2 Select Source>Registration Directory from the menu bar.
- 3 In the Apple SourceServer Registration Directory dialog box, double-click the object to see all its revision levels, then select the revision level you want.
- 4 Click the Take button, select a destination library, and click OK.  
PowerBuilder copies the revision level of the object over the current revision you checked out and regenerates (recompiles) the object.
- 5 Check the restored revision back in to the archive.

The restored revision becomes the current revision of the object.

---

**To restore the replaced version**

Repeat the procedure above, in step 3 selecting the replaced revision.

---

## Synchronizing objects

PowerBuilder maintains information about your libraries separately from Apple SourceServer. An Apple SourceServer archive is updated only when you perform an operation that affects the archive such as registering, checking out, or checking in an object. Therefore, you may need to resynchronize your PowerBuilder libraries periodically with the Apple SourceServer archive.

### When to synchronize

You need to resynchronize your libraries with the Apple SourceServer archives if one of the following events desynchronizes your PowerBuilder library entries:

- ◆ Unexpected system failures
- ◆ System crashes during a registration, check-in, or check-out operation
- ◆ Changes made to an archive outside PowerBuilder

If an event occurs that could desynchronize your library entries, you may need to resynchronize your libraries with the Apple SourceServer archives.

### What synchronizing does

Synchronizing makes sure that objects in your PowerBuilder library are the same as the latest revisions stored in Apple SourceServer. PowerBuilder maintains information about your libraries separately from Apple SourceServer. The Apple SourceServer archive is updated only when you perform an operation that affects that archive, such as checking objects out or in.

#### ❖ To synchronize objects:

- 1 Select the objects in the Library painter.
- 2 Select Source>Synchronize from the menu bar.

PowerBuilder regenerates (recompiles) the objects from their definitions in Apple SourceServer, and displays on the status bar the name of each object as it is synchronized, and displays on the status bar the name of each object as it is synchronized.