PDE/400 ™

Professional Development Environment

User Tutorial

Welcome!

Thank you for taking some time to review PDE/400. PDE is a powerful productivity system, providing control for your entire software development and maintenance process. With Project Management and Change Management, Object Creation, Software Testing, Version Integration, Software Distribution and Application Documentation, PDE provides a powerful framework for both the large IS department and the AS/400 programmer in the small shop.

This tutorial will help you discover some of the exciting features of PDE/400. You'll be able to easily review the PDE system using sample applications and quickly see for yourself the new levels of control and productivity now available. But first, let's get a brief overview of how PDE is used.

Applications, Versions and Libraries

Within the PDE/400 system, software is referenced by **Application, Version** and **Production Library**. An *application* might be a Payroll system or an Accounts Payable system. Within an application will be one or more *versions,* such as Version 3 or the Version for the Chicago location. Within an application/version, will be one or more *production libraries*. These are the libraries containing the objects that make up the application/version. For example, an application/version might be made up of two libraries: The first library contains the object programs and commands; the second library contains the data files for the application/version. Figure 1.0 below shows an example of this structure.

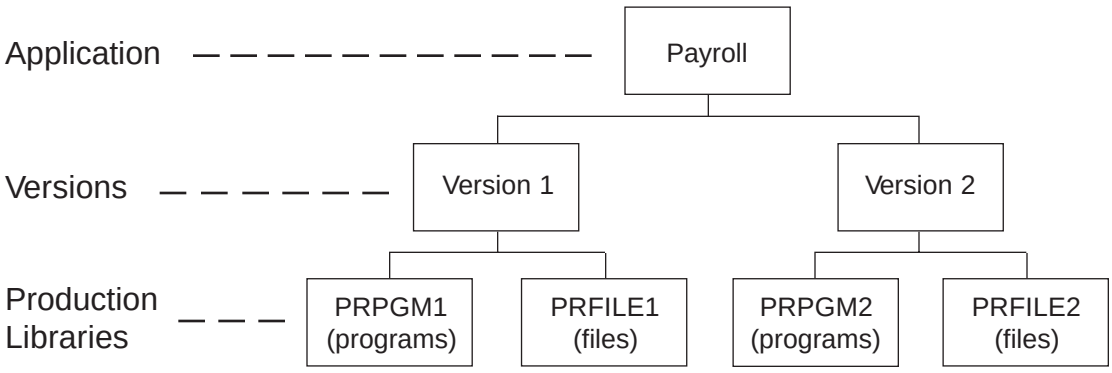


Figure 1.0 - Sample Payroll System showing Application, Version, and Production-Library Heirarchy

Setting up an Application within PDE

To begin with, the source for each application is cataloged into the PDE/400 system. After this process is completed, PDE will begin keeping track of all changes made in the application while maintaining the integrity of the source code and objects.

Working with Archives

All application source is stored in a special PDE archive. Only users with proper authority can *check out* the source, modify it, and *check in* the changed source. PDE ensures that your source and objects match and prevents changes from being overlaid. When changes are made to a source and the source is checked in, the archive is updated with the changes from the project. Using a special compressed format, several hundred versions of a program may be kept in less room then the original unachieved source.

The Development Process

As requests come in for changes to an application, you can easily set up a *project* and check out the appropriate source and objects so you can make the necessary changes. As source is changed and the objects are recreated, the original object attributes are automatically incorporated into the new object, using PDE's *Intelligent Object Creation* feature. The project may then be placed in a *test environment* in which you can test your changes without affecting production data. After the changes have been completely tested, you can check the project back into production. All changes are automatically logged, so months later you can have PDE show you exactly what changes were made during the project. PDE also offers many helpful documentation and cross-reference reports.

Easy to Use with a lot of Power under the Surface

While developing PDE, we have made it our goal to keep it simple and transparent for the user while incorporating extremely powerful features into the system. Many of PDE's capabilities are not immediately apparent on the surface because the interface is so easy to use. However, as you begin to use the product, PDE/400’s power will become quite apparent.

Tutorial Mode

When running this tutorial, you will enter a special tutorial mode. You will be operating using special tutorial files and libraries, free from the worry of overlaying your own systems. Only one user may be in tutorial mode at a time. (A licensed version of PDE outside of tutorial mode may be used by an unlimited number of users.) After you have completed the tutorial, all tutorial files and libraries can be removed from your system or restored so that another user can review the tutorial.

Examples you can Follow

This tutorial contains 12 independent examples. ***We recommend that you first try Example 1.*** **This example contains basic concepts which are used in many of the other examples.** The first six examples demonstrate using PDE to complete various application development and maintenance "mini projects". The other six examples show some of PDE's application documentation and research capabilities.

Time Estimates

1. Using PDE for a Small Project	30 min. (Do first)
2. Adding a field to a Physical File with Logical Files	20 min.
3. Parallel Development	5 min.
4. Researching Changes From a Previous Project	5 min.
5. Lineal Development	5 min.
6. Distributing Changes to Other AS/400's	5 min.
7. Library Detail Analysis Report	2 min.
8. Unreferenced Source Report	2 min.
9. Unreferenced Objects Report	2 min.
10. Object Last Used Report	2 min.
11. Display-File Sample	2 min.
12. Printer Spacing Chart	2 min.

Placing PDE/400 in tutorial mode

To start the tutorial...

- Key GO PDE400/DEMOMENU and press Enter.
- If this is your first time in the tutorial, select Option 8 to reset the tutorial files. This process takes a few moments.
- Select Option 1 to place the system in tutorial mode.
  You may now begin the first example of the tutorial. If you quit the tutorial and come back later you will want to first take this option again.

Using PDM Operation Codes

For your convenience, PDE uses Op-codes in PDM for three functions:

Function	Default PDM Op-code
Intelligent Object Creator	C
Project Manager	PM
Test Environment	TE

If you have changed these because of conflicts with existing codes you use, simply substitute the appropriate code in the following examples.

Example 1:
Using PDE for a Small Project

Introduction

First, be sure that you have placed PDE in tutorial mode as explained on page 4. Let's try a small sample project: In this scenario you are the programmer for a sporting goods store. A user has asked you to add a field to the sporting equipment maintenance screen. Follow along and see how PDE is used during this project.

Starting Project Manager

- Select Option 1 - Management from the menu.
  The *PDE/400 Management Menu* will be displayed.
- Select Option 1 - Project Manager from this menu.
  A window is displayed.

Selecting the Application

- Perhaps you can't remember the name of the *Application*. Prompt the field with F4 to view all the applications on the system.
- Select the Sporting Equipment application with a "1" and press Enter.
- Key a "1" in the *Version* field for version 1.
- Key "PDESPORTS" in the *Production Library* field. This is the name of the library where the production objects are kept for this application.
- Press Enter. The Project Management screen will be displayed. This screen shows all of the current projects for this application.

Setting up a New Project

- Press F6. The *Enter Project Information* screen will be displayed.
- On the Project line, enter the project title "Add GROUP field to maintenance screen". (You can also key any desired "Explanation or Comments" on the following lines.)
- Press Enter to add the project to the PDE/400 system. Note that the *development stage* is now at "Setup", indicating that you are beginning the project.

Identifying the Source to Check Out

Next, let's identify the source you wish to check out for this project...

- Select the newly-added project with a "5" and press Enter.
  A window is displayed showing all source associated with this application.
- Find the sports equipment display file (EQUIPSC) and select it for check-out with a "C".

Identifying the Source (cont.)

When you add a field to a display file, you must also change all programs which access that display file. If you can't remember which programs use that display file you can let PDE's scan find out for you...

- Press F10 to display the scan window.
- Key the name of the display file (EQUIPSC) in the *Find* field.
- Key "RPG" in the *Type* field to only search RPG programs.
- Press Enter. PDE will scan the source and find any RPG programs containing the display file name. The matching programs are then displayed for browsing or check-out. Notice that the scan brought up one program.
- If you'd like, you can select the program for browse with a "B" to see for yourself that the program uses the display file and will need to be checked out. Then press Enter and the previous window will be re-displayed.
- Select the program for check-out with a "C".
- Since you have identified all source needed for this project, press Enter to return to the *Project Management* screen.

Checking out the Source and Objects

You are now ready to check out the source and objects for the project...

- Place a "6" (check out source) next to the project and press Enter.
  The *Modify Development Stage Library List* screen is displayed. This screen indicates the library list to be used for source creation and for project testing. The production library is always the last library in the list. The source and objects will be checked out to the first library in the list, in this example - "PDEWORK". You can change this list to fit your needs. Source and objects can be checked out to your personal work library by simply changing the first library in the list to your own library. For this example we won't change it.

- Press Enter on this screen.
  PDE will now check out the source to a special file in your work library. A copy of each associated object will also be placed in the work library. A message stating "Check out is complete" will be displayed upon completion.

  Note that the development stage has changed to "1 of 2" and the library name shown is the library containing the checked out source and objects. Having identified and checked out the necessary items, you are now ready to make the necessary programming changes.

- Press F3 to exit project management.

Making the Programming Changes

Now, let's go into PDM and complete this project...

- Key "WRKOBJPDM PDEWORK" and press Enter.

Notice the two source files in the library which start with "#002SRC". The first one contains the display file to be changed and the second contains the program to be recreated.

- Place a "12" next to the "#002SRCFIL" source file and press Enter.
  This is the display file you checked out.
- Place a "2" next to the member and press Enter to change it.
- The lines which we want to add to the source are currently comments at lines 0010, 0033 and 0034. Make these lines active by removing the asterisks.
- After making the changes, press F3 and Enter to end the job and return to PDM.

For this demonstration, let's change PDM so object creation is done interactively...

- Press F18.
- Change the *Compile in batch* line to "N" and press Enter.

Using PDE's *Intelligent Object Creation*

Now you are ready to recreate the display file. Instead of using Option 14 to create objects, you will want to use PDE's Intelligent Object Creator. This function is an extremely powerful utility which automatically performs all of the steps necessary to properly recreate an object. It keeps track of ownership, authority, and all attributes...

- Place a "C" next to the display file and press Enter.
  PDE will automatically recreate it with all attributes identical to the original.
  (If you wished to change any object attributes, you could press F4 before pressing Enter. The prompted *Create Object* command would be displayed and you could make any desired changes.)
- After the display file has been recreated, press Enter to return to the PDE's *Work with Objects* screen.
- Place a "12" next to "#002SRCPGM" and press Enter. This source file contains the program which must be recreated. No programming changes are necessary, but since you have changed the externally-described display file contained in this program, you now need to recreate the program so the display file changes are incorporated.
- Place a "C" next to the program and press Enter. The program will be recreated.

The project is now complete and ready for testing. Testing can often be difficult because you don't want to disrupt production. The best testing method is to use a special library containing test files. This is the method used in this example.

Advancing the Project for Testing

The project is ready to move to the test phase.

- Place a "PM" *(Project Manager)* next to the program just recreated (or any other source file or member associated with the project) and press Enter. The Project Manager will automatically be brought up with the cursor on the appropriate project.
- Place a "7" next to the project and press Enter. The project will be advanced to the next library in the Development Stage Library List. In our example, this library is "PDESPORTST", a test library. Note that the development stage is now "2 of 2".
- Press F3 to exit Project Manager.

Testing the Changes

Just for fun, let's first run the original program from the production library to see how it used to function...

- Key "ADDLIBLE PDESPORTS" and press Enter.
- Key "CALL EQUIP" and press enter. A maintenance screen is displayed which allows you to roll through several sports items in a file. Notice that the new field you added, "Group", is not yet on the screen because the untested changes have not yet been put into production.
- Press F3 to exit the program.
- Key "RMVLIBLE PDESPORTS" and press Enter to return your library list to its original state.

Now, let's test the changes which we have made...

- Place a "TE" *(Test Environment)* next to the project program and press Enter. PDE will change the library list to the development stage library list minus the work library (PDEWORK).
- Select Option 1 and press Enter to see the new library list now in effect. Notice that the test library (PDESPORTST) is at the top of the user portion of the library list and the application production library (PDESPORTS) is at the bottom.
- Press Enter again to return to the previous screen.
- Key "CALL EQUIP" and press Enter. The maintenance screen will be displayed, showing the new "Group" field which you added. The test is successful.
- Press F3 to exit the program.

Some important things to note about the Test Environment:

1. The program, display file and data file being used in this test are all located in the test library, not in the production library.
2. Debug mode is automatically started so production data cannot accidentally be modified. Only files in libraries that have a type of "*TEST" can be modified. If the user tries to update production files, they will get an error message.

Testing the Changes (cont.)

Now you have completed testing the project.

- Press F3 to exit the Test Environment.
  As you exit the test environment, PDE automatically changes the library list back to its state prior to the test environment.

Checking the Changes into Production

You are now ready to advance the project and check the project back into production.

- Place a “PM” next to the program and press Enter.
- On the Project Management screen, advance the project by placing a “7” next to the project and pressing Enter. Since there are no more development stages for the project (2 of 2), the project will now be checked back into production.

The Automatic Check-in Process

The following occurs when changes are checked in:

1. The newly created objects will be moved into the production library.
2. The source will be re-archived into the source archive.
3. The project will be marked as completed.

When finished, a message will be displayed at the bottom of the screen to inform you that “Check in is complete”.

- Press F3 to exit Project Manager.

CONGRATULATIONS! You have completed the project and the users are happy. Because of PDE's control, you are assured that no one else's changes have been overlaid and that the production object and the archived source match. You have also sampled a little of what the Intelligent Object Creator can do for you. For future reference, PDE has created a complete record of the project. The project may later be reviewed to see what source and objects were changed, who made the changes, and when the changes were implemented into production. If detail is needed, PDE will even show every change that was made within each source.

Review

In this section you have...

- Set up a project.
- Identified and checked out the source and objects involved.
- Changed the source and effortlessly recreated the objects with their original attributes.
- Tested the project in the safety of the Test Environment.
- Checked the changes back into production.

Example 2:
Adding a field to a physical file with connected logical files and special attributes

Introduction
In this example you will use PDE to accomplish a task that is normally quite involved and tedious: adding a field to a physical file which has logical files built over it and has special attributes that must be retained. We will speed up the pace a little bit, allowing you to use the concepts you learned in the first example.

Starting the Project
You learned how to set up a new project in the last example. Bring up Project Manager for the Sporting Equipment application. Add a new project with the title “Add 'Location' to EQUIPMST file”.

Checking out the Source
- Use a “5” to bring up the *Identify Source for Project* window and place a “C” on physical file EQUIPMST but don't press Enter.

Now let’s try something new. Suppose you don't know which objects reference this physical file. PDE can identify them for you...
- Press F11 to display the objects which reference the selected physical file. PDE will find each object and display the source for you to browse or check out. First, a window will be displayed where you can select which fields will be changed.
- Press F9 to select all of the fields. Then, press Enter. A list of the source for each object that references the EQUIPMST file will be displayed.
- Place a “C” on EQUIPSC and EQUIP to mark them for check out. You don't need to check out the associated logical files because PDE will simply reconnect them to the physical file after it is recreated.
- Check out the source which has been selected and let’s start the project.

Changing the Physical File Source
- Use PDM to bring up the work library and edit the physical file source with SEU.
- Remove the “*” on line 9 to activate the line. This line which was a comment now defines the new field.
- End SEU.

Recreating the File
- Recreate the physical file by placing a “C” on it in PDM and pressing F4.
- Change the *Recreate Referenced Objects* option to “*YES”. This option will automatically recreate all objects which reference the selected object. These objects need to be recreated so that they'll contain the format changes from the file. Using this option in PDE/400’s Intelligent Object Creator will eliminate the possibility of level checks.

Recreating the File (Cont.)

- Now, for interest's sake, let's press Enter to see what special attributes PDE has determined need to be incorporated in the new object. Note that *Maximum members* is 2 and *Reuse deleted records* is *YES. PDE looked at the old copy of the object and pulled in the various attributes.
- Press Enter. First, the physical file will be recreated. Then, all objects in the work library which reference the physical file will be recreated.

  While these objects are being recreated, let's find out about some of the functions which are taking place. The PDE Intelligent Object Creator automatically handles the many necessary tasks for you.

Intelligent Object Creation

1. Deletes all logical files which are connected to the physical file.
2. Recreates the physical file with it's original attributes
3. Maps the data contained in each member of the original physical file to the correct member in the new physical file.
4. Reconnects all logical files originally connected to the physical file. All logical files retain their original attributes.
5. Sets up the same authority, ownership and journaling for the physical file and all connected logical files.
6. Recreates all objects which reference these files, being sure to re-establish all members, data, logicals, attributes, ownership, authority and journaling. These objects are recreated in the proper order.

Needless to say, this process can become quite a task when done manually; there are so many things to keep track of. Using PDE, it is done almost transparently. PDE's Intelligent Object Creator takes care of all of these important and necessary functions. *This capability alone can be worth the cost of the PDE/400 system, but is only a small part of the total package.*

Completing the Project

After all of the objects have been recreated, you would normally test the project and then check it back into production. Since you've already learned about these functions, you can go on to the next example.

Review

In this example you have...

- Used PDE to indentify referenced objects to check out
- Added a field to physical file source
- Used PDE's Intelligent Object Creator to correctly recreate the file, it's logical files, and all objects which reference the file

Example 3:
Parallel Development

Introduction

Occasionally two programmers need to be making different changes to the same source at the same time. This is called *parallel development.* When parallel development occurs, care must be taken to ensure the integrity of your code.

First, each programmer may not even be aware that the other programmer is changing the same source. It's not uncommon in this situation to have the changes of one programmer overlaid by those of the other. When the overlay is finally discovered, recreating the changes will probably involve re-analyzing the original problem, re-making the changes, re-testing and re-discussing the situation with the user.
A frustrating situation at best.

If both programmers are aware that parallel development is taking place, they must coordinate with each other to make sure both sets of changes are merged together into a single source. Doing this process manually can be time consuming and error prone.

Also, you must make sure that the two sets of changes don't conflict with each other. The source should be visually inspected with both sets of changes highlighted, and the object should be re-tested to be sure it functions properly.

PDE automatically recognizes when parallel development is taking place.
The following example demonstrates how PDE helps you perform these functions with ease.

- Start Project Manager for the Sporting Equipment application.
  Note projects 1 and 2. Both of these projects have checked out the same program. The changes have already been made in both of these projects and they are ready to be checked in.
- Check in the first project. Nothing special happens in this process and the program object is moved back into the production library.
- Now check in the second project. Since both projects made changes to the same program and the first project has been checked in, the PDE Project Manager will determine that parallel development has taken place. Now it will merge the changes from both projects together into the source in the second project.
- Press Enter on the *Parallel Development* window to start the merge process. That’s all there is to it. PDE does all the work. When the process is complete, a message will be displayed stating “Project returned to first development stage”.

A report is produced during the merge process and can be reviewed to determine if the parallel changes conflict. This report shows the changes which were made in each project and where these changes were merged into the source. For your convenience the report is reproduced as Figure 3 on the next page.

The Member Merge Report

- Let's review the report which was produced during the parallel development check-in process. The report is shown below in Figure 3. Note that certain lines have been added to and deleted from the source by each project.
- Since the parallel changes do not conflict, you would simply recreate the applicable objects and check the project back into production. Since you've done this in earlier examples, you can go on to the next example.

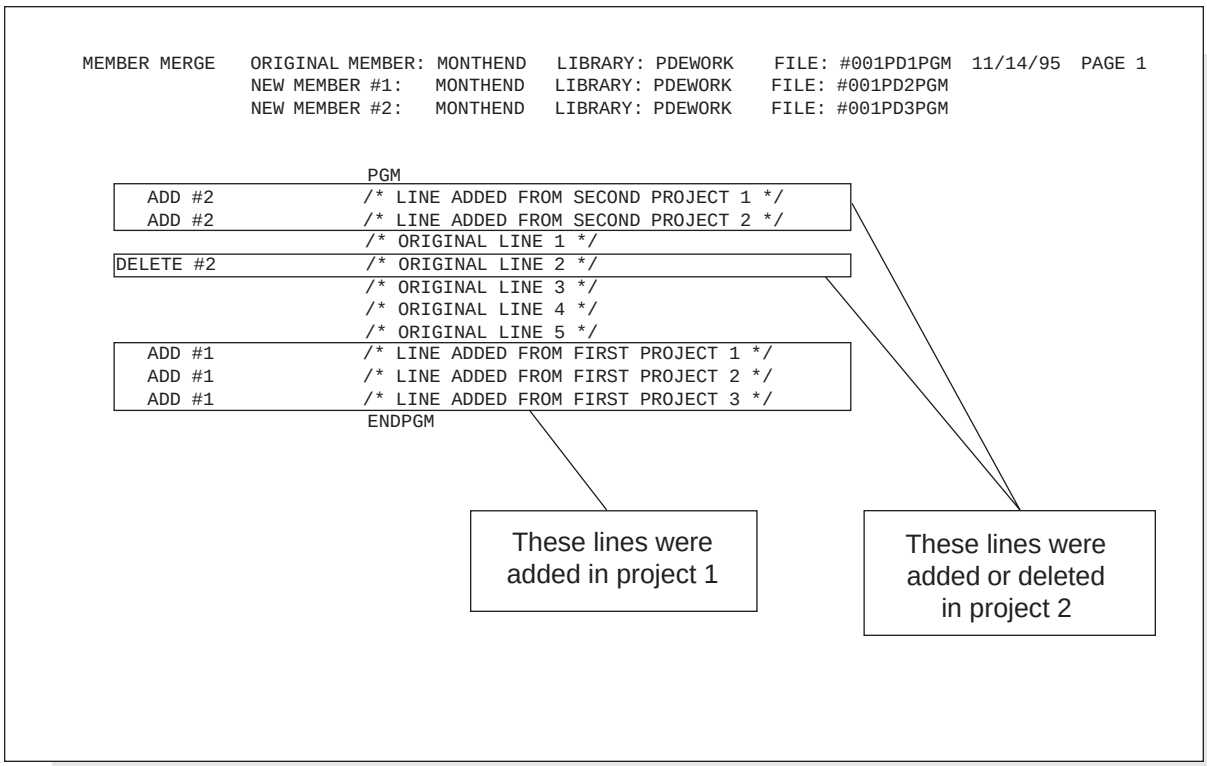


Figure 3 - Member Merge Report for Example 3

Review

In this example you...

- Used PDE to merge together changes made in the same source by two different programmers
- Produced and reviewed the generated merge report

Example 4:
Researching Changes from a Previous Project

Introduction

Have you ever had a programmer go on vacation only to have a user complain about a problem you thought the vacationing programmer had fixed a couple of months ago? How do you determine what he did to the application? With PDE, this is easy. You simply find the project where he made the changes and ask PDE to show you what changes were made. All source and source changes are kept in a special compressed format. Using this format, it is possible to store a hundred or more versions of a source in less room than the source took in its original uncompressed form.

Printing the Changes From the Previous Project

- Exit PDM.
- Exit the PDE/400 Management Menu.
- Select Option 2 - Application Development from the PDE/400 menu.
- Select Option 2 - Work with Archives from the next menu.
- For application, enter “VACN PLAN”. For version, enter “2”. For production library, enter “PDEVACPLN2”.
- When your cursor is on the release number, press F4. Here are all of the projects which have been completed on the Vacation Planner system with completion date and user.
- Let’s see what happened in release 1. Place a “5” on that release to review the details. Here, you can see the project Description, Explanation and Comments, the source which was modified and the development library list.
- Now press Enter to return to the previous window.
- Place a “1” on that project and press Enter.
- Press Enter again on the next screen.
- Now, press F8 to display only the source that was changed in this project.
- Place a “7” next to each source to print a report of the changes which were made to the source by this project and press Enter. These reports are displayed for you in the following Figures 4.1 - 4.3.
- Press F3 to exit the *Work with Archives* screen.

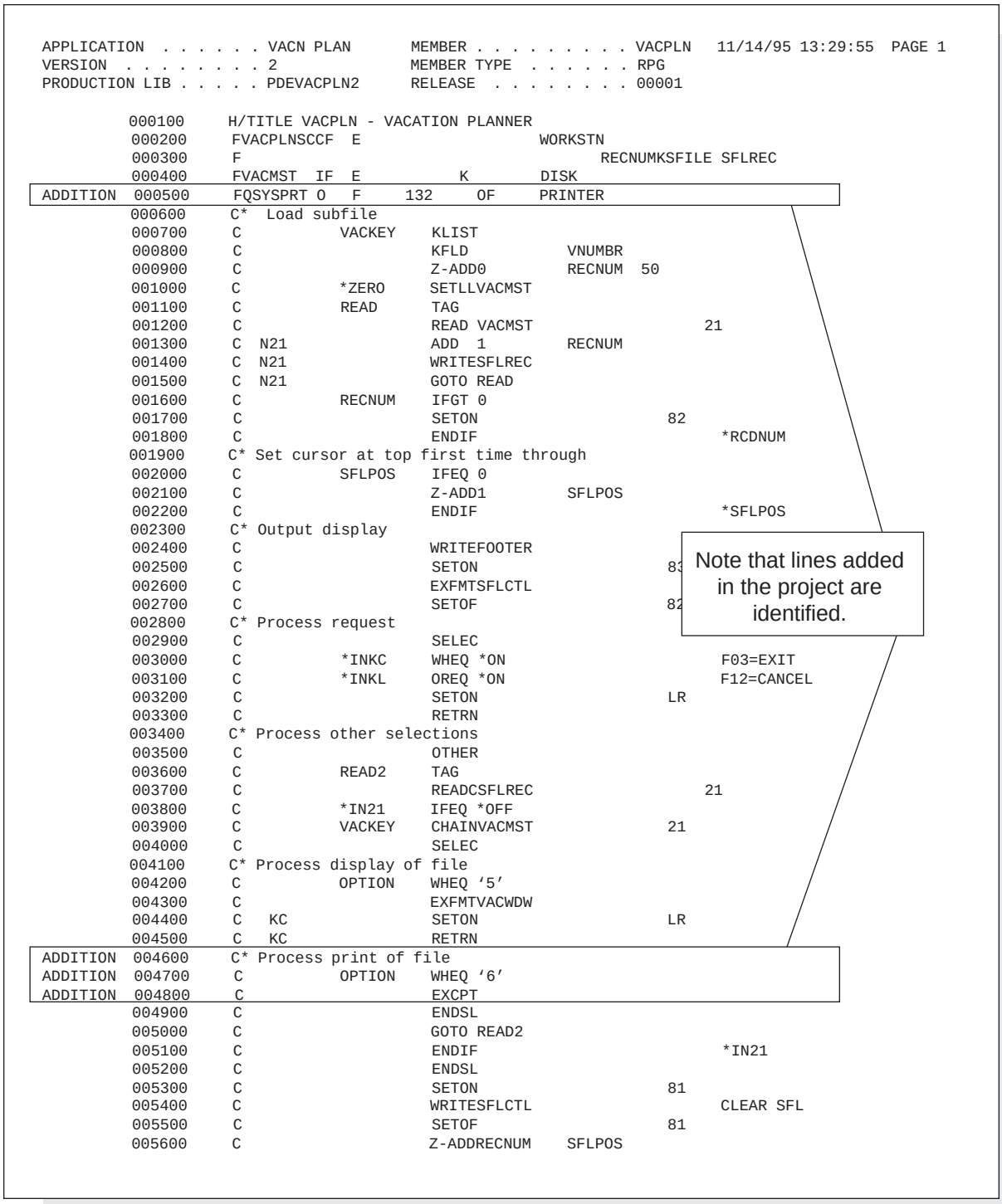


Figure 4.1 - Page 1 of Report Showing Changes Made in a Project

```
APPLICATION  . . . . . VACN PLAN      MEMBER . . . . . VACPLN   11/14/95 13:29:55   PAGE 2
VERSION  . . . . . 2                  MEMBER TYPE  . . . . . RPG
PRODUCTION LIB . . . . PDEVACPLN2     RELEASE  . . . . . 00001

ADDITION  005700     OQSYSPRT E    2 7
ADDITION  005800     O                        UDATE Y    9
ADDITION  005900     O                                  48 'VACATION PLANNER'
ADDITION  006000     O                                  75 'Page'
ADDITION  006100     O                        PAGE  Z   79
ADDITION  006200     O        E  2
ADDITION  006300     O                        VSDESC    40
ADDITION  006400     O        E  1
ADDITION  006500     O                        VLDSC0    65
ADDITION  006600     O        E  1
ADDITION  006700     O                        VLDSC1    65
ADDITION  006800     O        E  1
ADDITION  006900     O                        VLDSC2    65
ADDITION  007000     O        E  1
ADDITION  007100     O                        VLDSC3    65
ADDITION  007200     O        E  1
ADDITION  007300     O                        VLDSC4    65
ADDITION  007400     O        E  1
ADDITION  007500     O                        VLDSC5    65
ADDITION  007600     O        E  1
ADDITION  007700     O                        VLDSC6    65
ADDITION  007800     O        E  1
ADDITION  007900     O                        VLDSC7    65
ADDITION  008000     O        E  1
ADDITION  008100     O                        VLDSC8    65
ADDITION  008200     O        E  1
ADDITION  008300     O                        VLDSC9    65
```

Note that lines added in the project are identified.

Figure 4.2 - Page 2 of Report Showing Changes Made in a Project

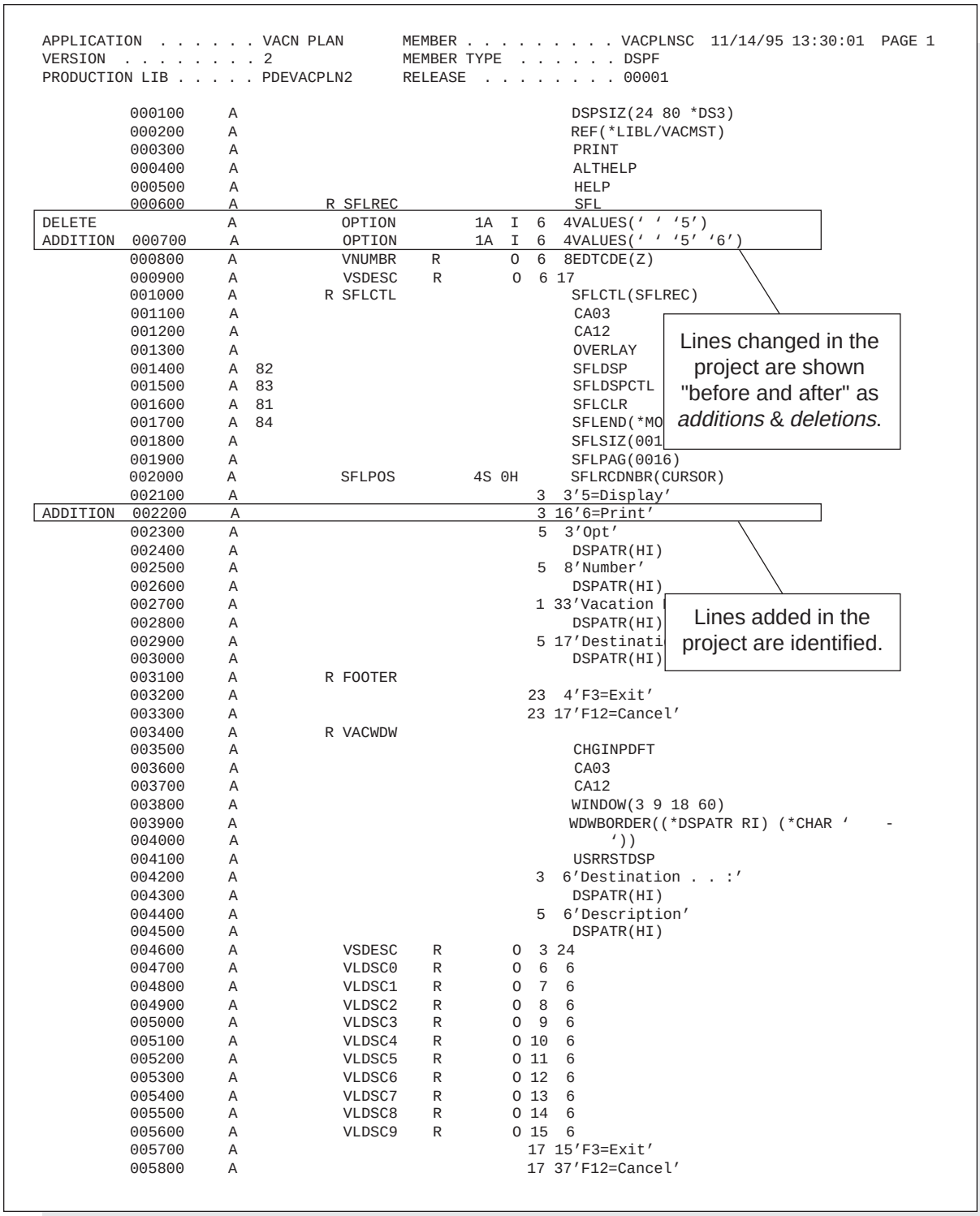


Figure 4.3 - Additional Report Showing Changes Made in a Project

Example 5:
Lineal Development

Introduction

This example will demonstrate how PDE controls what we call Lineal Development. Have you ever needed to create a new version of an application while still maintaining the original version? In this situation, bugs are sometimes found in the original version of the application. After making the corrections, you then must remember to make these same changes in the new version of the application. This situation is called Lineal Development because each successive version of the application is a lineal descendant of the previous version. PDE can make it easy for you; the change in one version can be automatically "rippled" into the related version.

Overview

Let's see how this works. The PDE/400 tutorial has two versions of a Vacation Planner application. The first version uses library "PDEVACPLN1". Version 2 of the Vacation Planner has an added print feature so users can print vacation information and uses library "PDEVACPLN2", which is a lineal descendent of the first version.

Check out the Project and Make Changes

- Bring up the first version of the Vacation Planner application with project manager. Note that a project has been setup in this version. This project was set up because the users needed a date added to the screen.
- Check this project out to the PDEWORK library.
- Exit Project Manager.
- Access the PDEWORK library with PDM. In the source file with the text of *"PDEVACPLN1 - Add date to screen"* edit the DSPF source using SEU.
- Remove the "*" from lines 28 and 29 and end SEU. This activates the two lines which define the date field.
- Now, recreate the display file with a "C" to invoke the Intelligent Object Creator.

Check in the Project

- Place a "PM" on the source to bring up Project Manager.
- Check this project back into the production library. PDE will determine that a lineal descendent exists for this application/version and ask if you want to create a project in the lineal descendent to ripple these changes to that descendent.
- Answer "Y" for yes and press Enter, because we want this added date field to also be a part of the new version of the Vacation Planner. As part of the check in process, PDE will create a new project in Version Two of the Vacation Planner application to make these changes.

“Ripple” the Changes into the Related Version

- Press F2 and bring up version two of the vacation planner application. Note that the new project has been setup in Version Two of the Vacation Planner application and is ready to be checked out.
- Check out the source for this project. When you do this, the project manager will merge the changes from the original project with the source from this application/ version. Normally, you would review the report which is produced by the merging process. A copy of this report has been included below in Figure 5. On the report below, changes marked with “#2” show the differences between the two versions. Changes marked with “#1” show changes which were rippled from the first project. Once you've determined that the merged code is correct, the objects can be recreated and checked back into production. Because this process has already been covered in previous examples, you can go on to the next e xample.

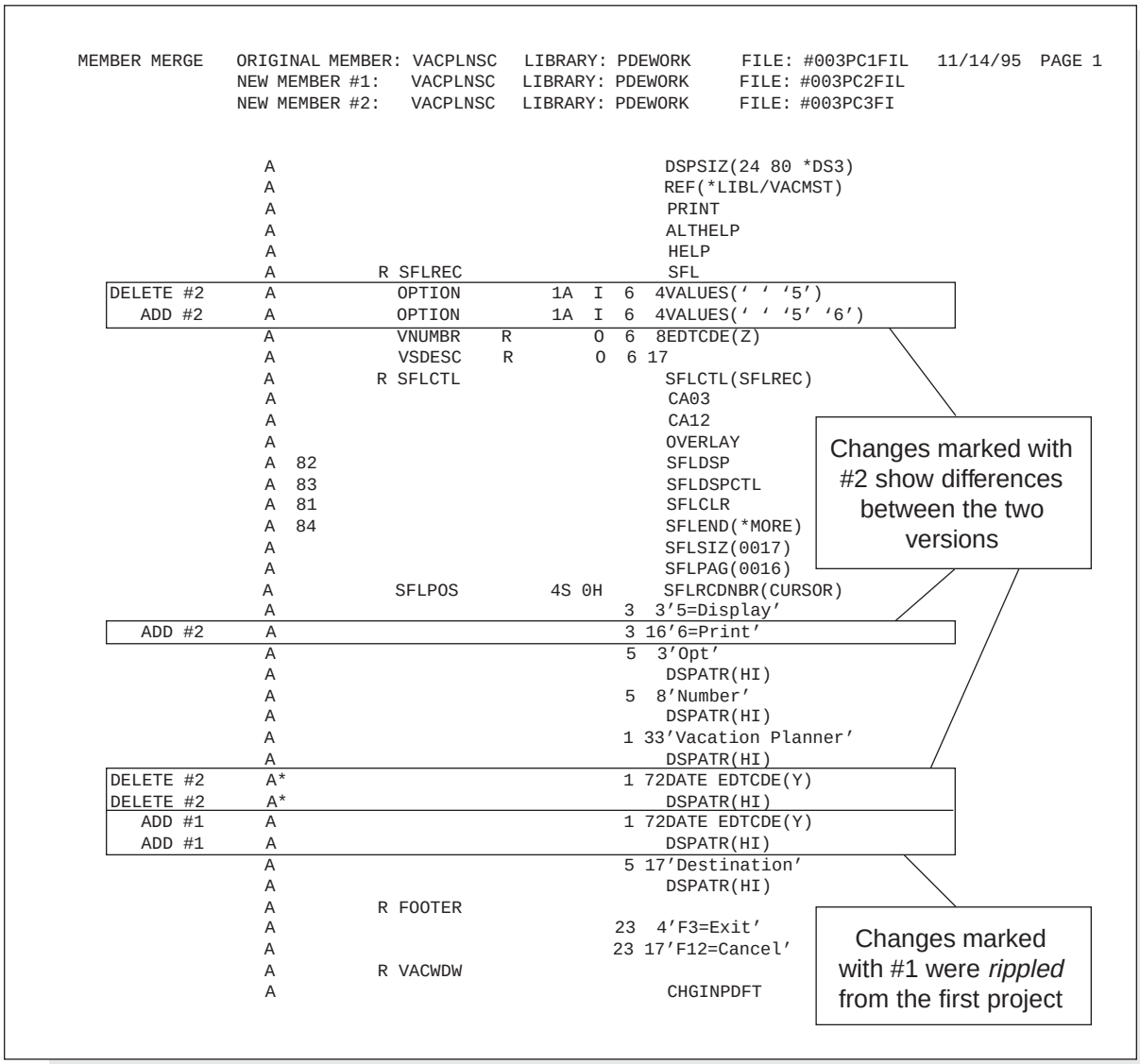


Figure 5 - Member Merge Report for Example 5

Review

In this example you have...

- Added a date field to a screen in Version 1 of an application
- Checked the project in and been notified by PDE of another version which is a descendent of Version 1
- Allowed PDE to create a new project for the same changes to be made in Version 2 of the application
- Used PDE to automatically merge the changes made in your project into the newer version of the application

Example 6:
Distributing Changes to Other AS/400's

Introduction

Once your changes are ready to be distributed, PDE will make the distribution task painless and foolproof. You simply need to select the location to which you wish to distribute; all releases that they have not yet received will be installed on your user's systems when they install the module created by PDE. This makes the task of sending upgrades to your users simple and accurate.

Creating the Distribution Module

- Select Option 1 - Management from the PDE/400 menu.
- Select Option 3 - Create Distribution Module from the next menu.
- Enter the Application, Version and Production Library for "Version 2" of the "Vacation Planner" application and press F4 on the *Distribution Location* field. PDE will display all locations which have been setup in the Distribution Location file. (This is maintained off of the *Maintenance* menu.)
- Place a "1" on "Atlanta" and press Enter. All releases which have not yet been distributed to the Atlanta location will be included when the distribution module is created. PDE keeps track of the current release level of each location and selects the appropriate releases.
- Now enter the name of the library in which you want the distribution module created. All objects from the appropriate releases will be stored in this automatically-created library, as well as the installation software which will be used to install these objects.
- Press Enter to create the distribution module. The specified library will be created. It is then ready to be sent to the user using whatever method desired.

When you create a distribution module, a new library will be created containing a save-file of all your new objects and the software necessary to install those objects on your user's system. You may add any of your own installation software if you like. Then, simply save the library and send it to your users.

Once the users have restored the library at their site they simply need to call an install program to begin the upgrade. The new objects will be copied to the user's system, and file data is automatically remapped, logical files are rebuilt and reconnected, etc. If any problems are encountered, the user's library is restored to its state prior to the install.

PDE's Reporting and Documentation Functions

The previous six examples demonstrated using PDE to complete various sample projects which could be encountered in the course of a programmer's busy schedule. The following six examples show some of PDE's reporting and documenting capabilities which are available on the Documentation menu.

Example 7:
Print the Library Detail Analysis Report

Introduction

Suppose you have a situation where you are trying to figure out what is taking all the room in one of your libraries. It seems like the library is larger that it should be and you want to quickly check it out.

- From the *PDE/400 - Documentation* menu, select Option 2 - Library Analysis - Detail.
- Enter the name of the library and press Enter.

A sample of this report is included below for you to review.

```
11/09/95  15:46:47                    LIBRARY DETAIL ANALYSIS REPORT - PRLIB                    ALC400    Page   1

                                Last                                                  Dlt Rec
         Name       Size (K)   Used    Owner      Text                              Space (K)  Mbr  Created   Saved

*PGM

  RPG    PRCHK             9 10/31/95  QPGMR      Print payroll checks                              11/03/82
  RPG    PRGL             10 10/31/95  QPGMR      Payroll general ledger interface                  11/07/82
  RPG    PRPAY            52 10/31/95  QPGMR      Calculate employee pay                            11/02/82
  RPG    PRQTR             8  9/30/95  QPGMR      Print quarterly reports                           11/15/82
  RPG    PRTIM            28 10/31/95  QPGMR      Calculate employee time                           11/16/82
  RPG    PRYR             24 12/31/94  QPGMR      Print year end reports                            11/10/82

                         131  Total  *PGM

*FILE

  PF-DTA PRMST           116 11/09/95  QPGMR      Payroll Master File                        70     1 10/01/81
  PF-SRC QCLSRC           28  9/03/95  QPGMR      Payroll CL source                                 4 10/07/82
  PF-SRC QCMDSRC           8 11/09/95  QPGMR      Payroll Command source                            3 10/07/82
  PF-SRC QRPGSRC          43 10/14/95  QPGMR      Payroll RPG source                                8 10/07/82

                         195  Total  *FILE                                                  70  Total

*CMD

         PRMONTH           2 10/31/95  QPGMR      Payroll Monthly Job                               11/14/82
         PRQUARTER         2  9/30/95  QPGMR      Payroll Quarterly Job                             11/20/82
         PRYEAR            2 12/31/94  QPGMR      Payroll Yearly Job                                12/07/82

                           6  Total  *CMD

*MENU

         PRMENU            7 10/31/95  QPGMR      Payroll Menu                                       5/25/83

                           7  Total  *MENU


                         339  Total All Objects
```

Note that a significant amount of space is taken up by deleted records in this file

Figure 7 - Library Analysis Detail Report

Example 8:
Printing the Unreferenced Source Report

Introduction

Do you ever wonder about the status of all the source programs on your system? You can't remember whether some of the programs are being used or are just old programs you forgot to delete. PDE can tell you if there are any source programs that are not actually being used in your application.

- From the *PDE/400 - Documentation* menu, select Option 3 - Unreferenced Source.
- Fill out the screen and press Enter to start the process .

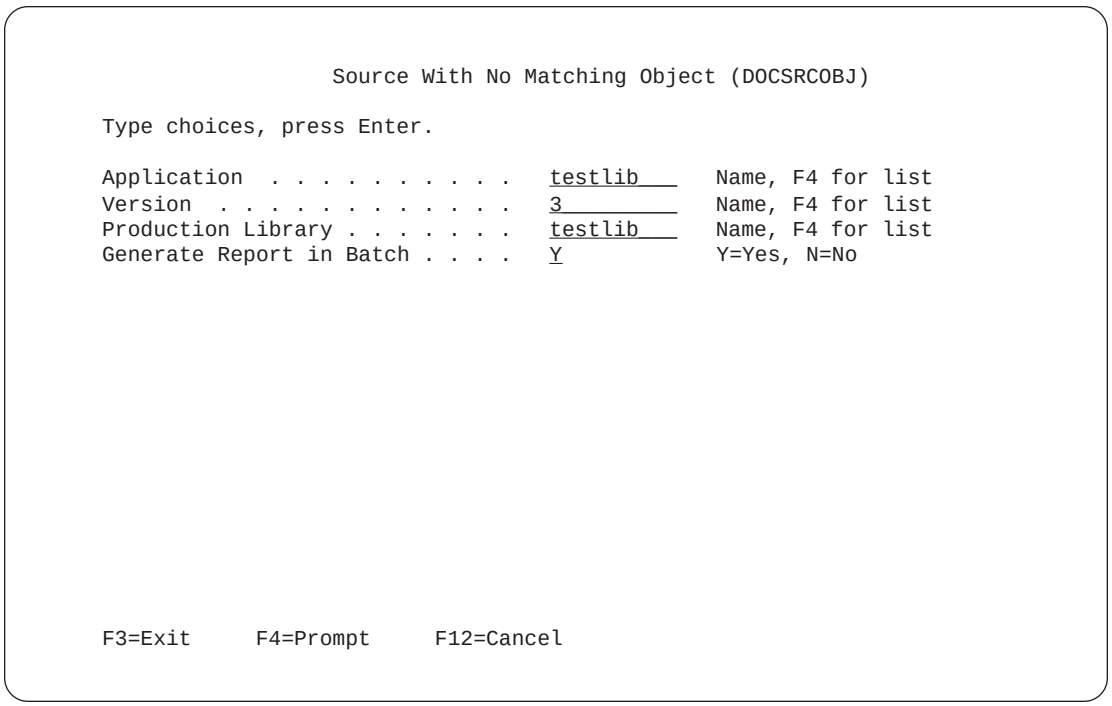


Figure 8.1 - Source Without Object Entry Screen

A sample of the report is shown on the following page. Note that this report will only work for applications which have been cataloged into the PDE/400 system.

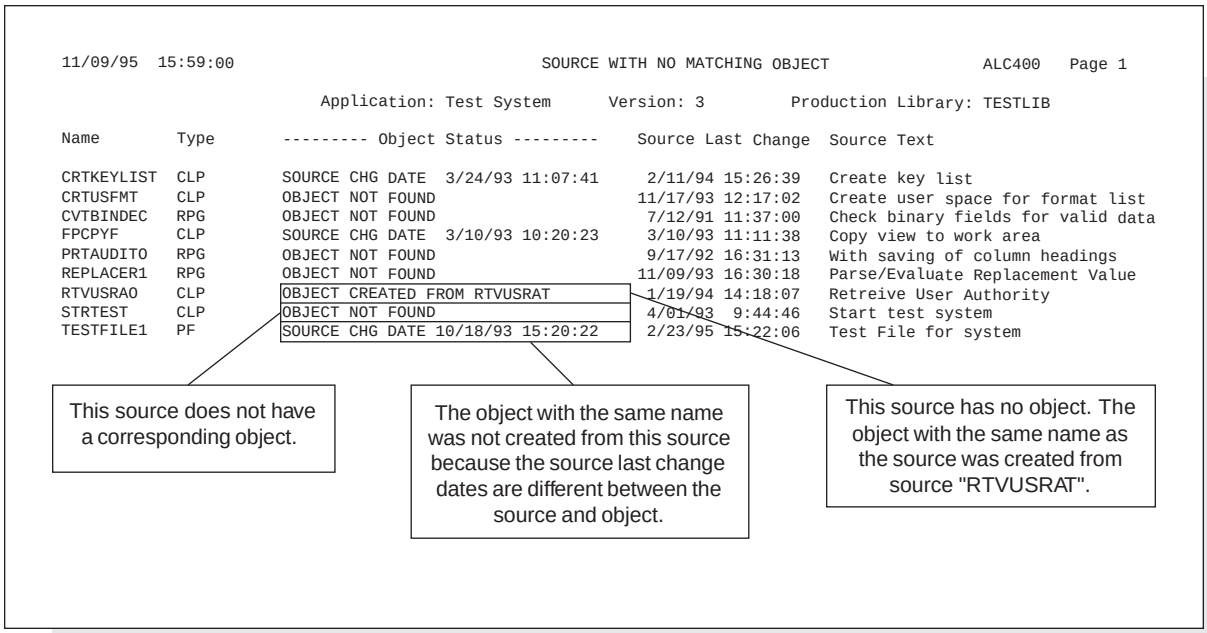


Figure 8.2 - Source Without Object Report

Example 9:
Printing the Unreferenced Objects Report

Introduction

Suppose you are trying to determine if each object in an application has a corresponding source. You have just been put in charge of the application and you want to make sure that everything exists. Ask PDE/400.

- From the *PDE/400 - Documentation* menu, select Option 4 - Unreferenced Objects.
- Fill out the screen and press Enter to start the process.

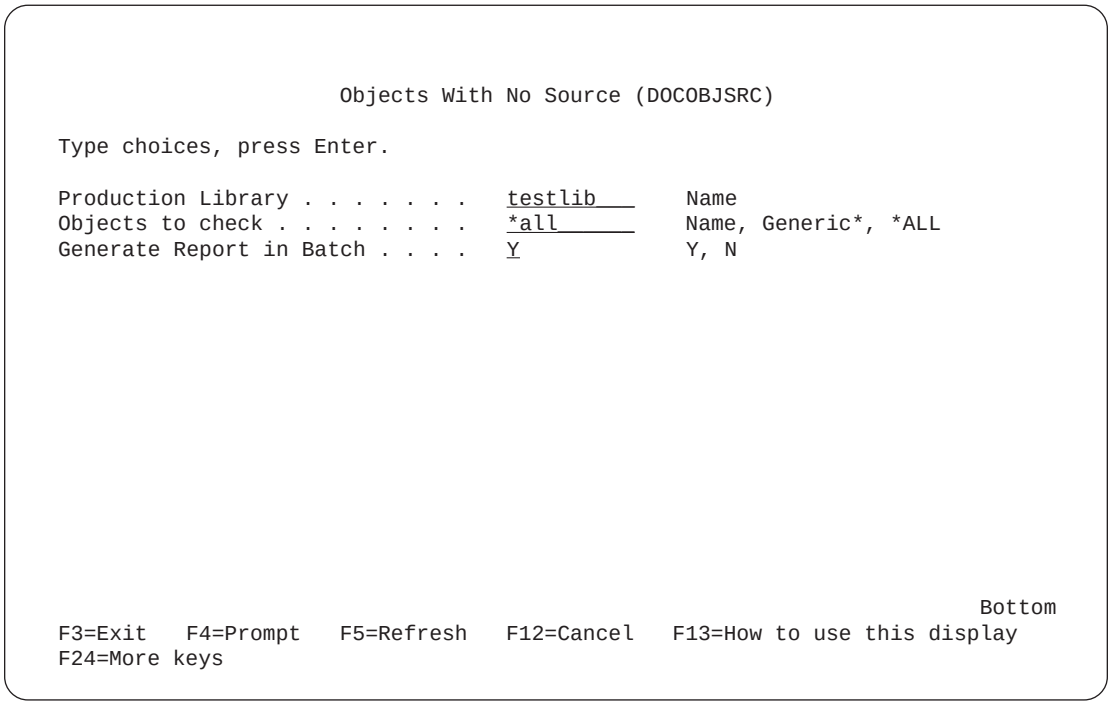


Figure 9.1 - Objects Without Source Entry Screen

After entering the information and running the job, the resulting report would satisfy your query. A sample of this report is included on the following page. Note that this report will only work for applications which have been cataloged into the PDE/400 system.

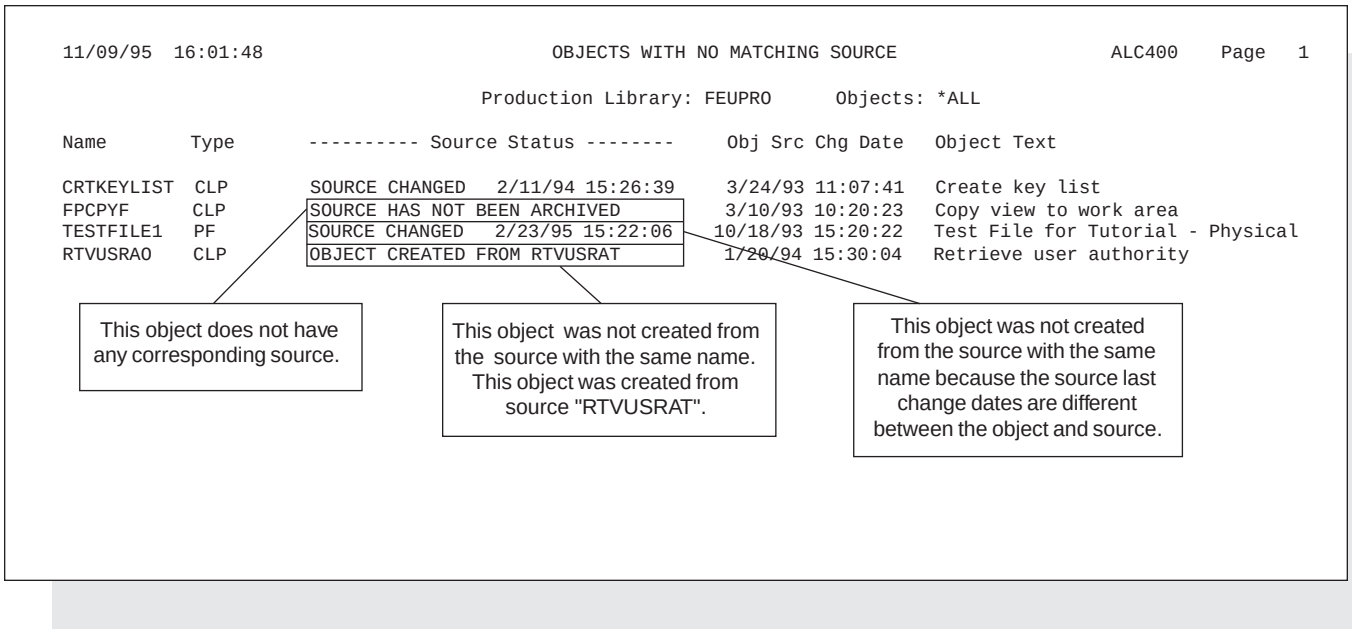


Figure 9.2 - Objects Without Source Report

Example 10:
Printing the Object Last Used Report

Introduction

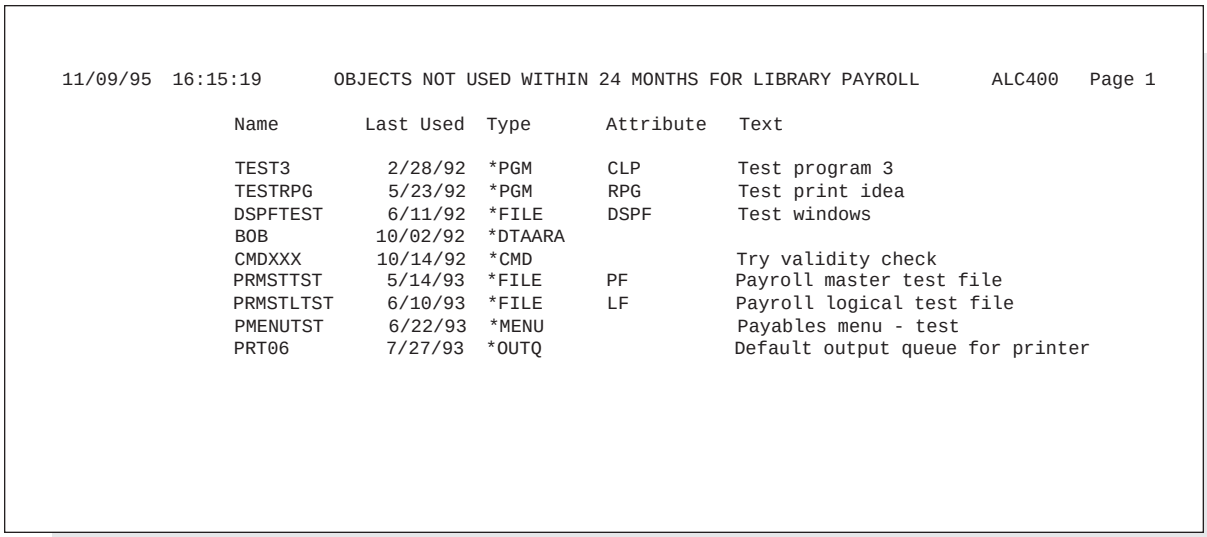
Lets say that you need to reduce the size of a library. You've decided to remove all unnecessary programs which have not been used in the last two years and copy them to an archive tape. PDE/400 can help you with the process.

- From the *PDE/400 - Documentation* menu, select Option 5 - Object Last Used.
- Fill out the screen and press Enter to start the process.

```
                        Objects Last Used (DOCOBJUSE)

Type choices, press Enter.

Library  . . . . . . . . . . . .   payroll___    Name
Months Older than  . . . . . . .   24           Number
Generate Report in Batch . . . .   Y            Y, N

                                                                    Bottom
F3=Exit   F4=Prompt   F5=Refresh   F12=Cancel   F13=How to use this display
F24=More keys
```

Figure 10.1 - Object Last Used Entry Screen

PDE will do all of the research and provide you with a report of it's findings. A sample of the report is included on the following page.



```
11/09/95  16:15:19      OBJECTS NOT USED WITHIN 24 MONTHS FOR LIBRARY PAYROLL      ALC400   Page 1

               Name        Last Used  Type       Attribute   Text

               TEST3         2/28/92  *PGM       CLP         Test program 3
               TESTRPG       5/23/92  *PGM       RPG         Test print idea
               DSPFTEST      6/11/92  *FILE      DSPF        Test windows
               BOB          10/02/92  *DTAARA
               CMDXXX       10/14/92  *CMD                   Try validity check
               PRMSTTST      5/14/93  *FILE      PF          Payroll master test file
               PRMSTLTST     6/10/93  *FILE      LF          Payroll logical test file
               PMENUTST      6/22/93  *MENU                  Payables menu - test
               PRT06         7/27/93  *OUTQ                  Default output queue for printer
```

Figure 10.2 - Object Last Used Report

Example 11:
Print a Display File Sample

Introduction

Suppose you want to add a field to several screens in a display file. You would like the layout for each screen in the display file so you can decide where to place the field in each screen. PDE can print these screens for you.

- From the *PDE/400 - Documentation* menu, select Option 14 - Display File Sample.
- Fill out the screen and press Enter to start the process.

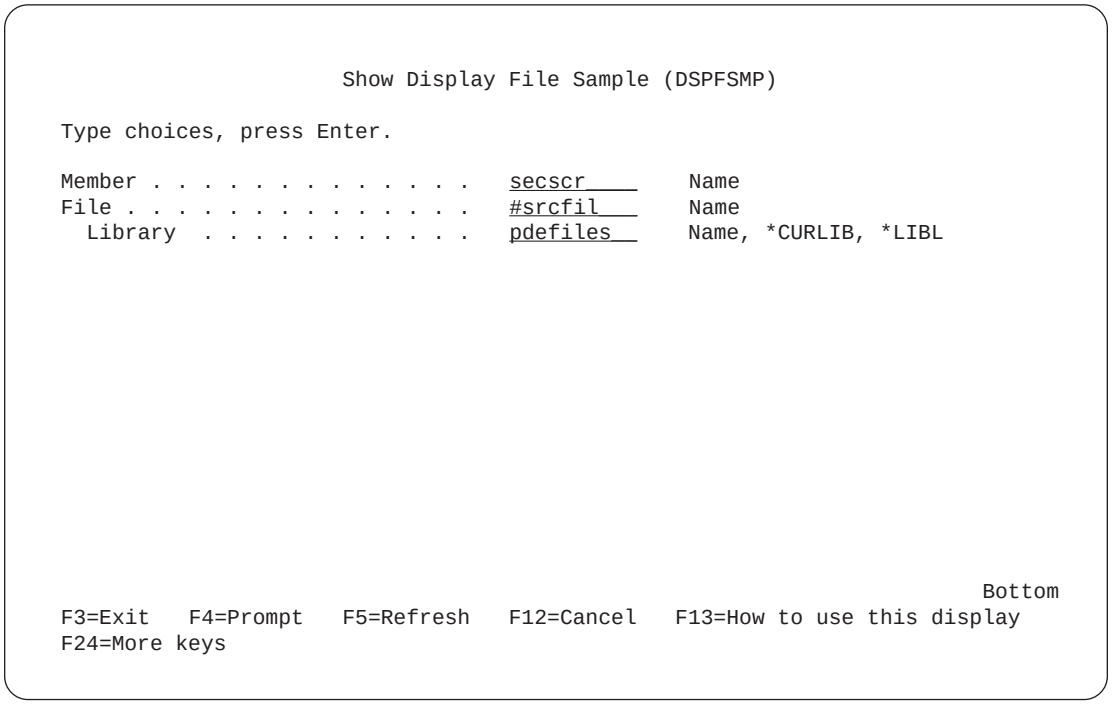


Figure 11.1 - Display-file Sample Screen

PDE will create a report showing what each screen in the display file looks like. A sample of this report is shown on the following page.

```
Source Library . . . PDEFILES
Source File  . . . . #SRCFIL
Source Member  . . . SECSCR
Record Format  . . . SFLREC, SFLCTL
Screen Size  . . . . 24 X 80

1...*...10....*...20....*...30....*...40....*...50....*...60....*...70....*...80
                            Work with User Security

2=Update    4=Delete
                                            Production   --Authority to--
Opt  User          Application  Version      Library     Maintain    View
 X   XXXXXXXXXX    XXXXXXXXXX   XXXXXXXXXX  XXXXXXXXXX       X         X
 X   XXXXXXXXXX    XXXXXXXXXX   XXXXXXXXXX  XXXXXXXXXX       X         X
 X   XXXXXXXXXX    XXXXXXXXXX   XXXXXXXXXX  XXXXXXXXXX       X         X
 X   XXXXXXXXXX    XXXXXXXXXX   XXXXXXXXXX  XXXXXXXXXX       X         X
 X   XXXXXXXXXX    XXXXXXXXXX   XXXXXXXXXX  XXXXXXXXXX       X         X
 X   XXXXXXXXXX    XXXXXXXXXX   XXXXXXXXXX  XXXXXXXXXX       X         X
 X   XXXXXXXXXX    XXXXXXXXXX   XXXXXXXXXX  XXXXXXXXXX       X         X
 X   XXXXXXXXXX    XXXXXXXXXX   XXXXXXXXXX  XXXXXXXXXX       X         X
 X   XXXXXXXXXX    XXXXXXXXXX   XXXXXXXXXX  XXXXXXXXXX       X         X
 X   XXXXXXXXXX    XXXXXXXXXX   XXXXXXXXXX  XXXXXXXXXX       X         X
 X   XXXXXXXXXX    XXXXXXXXXX   XXXXXXXXXX  XXXXXXXXXX       X         X
 X   XXXXXXXXXX    XXXXXXXXXX   XXXXXXXXXX  XXXXXXXXXX       X         X
 X   XXXXXXXXXX    XXXXXXXXXX   XXXXXXXXXX  XXXXXXXXXX       X         X
 X   XXXXXXXXXX    XXXXXXXXXX   XXXXXXXXXX  XXXXXXXXXX       X         X
 X   XXXXXXXXXX    XXXXXXXXXX   XXXXXXXXXX  XXXXXXXXXX       X         X
                                                                        More...
```

```
Source Library . . . PDEFILES
Source File  . . . . #SRCFIL
Source Member  . . . SECSCR
Record Format  . . . SECUSER
Screen Size  . . . . 24 X 80

1...*...10....*...20....*...30....*...40....*...50....*...60....*...70....*...80


                ................................................
                :                                              :
                :          Add User Security Information       :
                :                                              :
                :  User ID . . . . . . . . . . . : XXXXXXXXXX  :
                :                                              :
                :  Application . . . . . . . . . : XXXXXXXXXX  :
                :  Version . . . . . . . . . . . : XXXXXXXXXX  :
                :  Production Library  . . . . . : XXXXXXXXXX  :
                :                                              :
                :  Authority to Maintain . . . . : X   Y=Yes   :
                :  Authority to View . . . . . . : X   N=No    :
                :  Text:                                       :
                :   XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX   :
                :                                              :
                :           F3=Exit         F12=Cancel         :
                :                                              :
                :..............................................:
```

Figure 11.2 - Display-file Sample Report

Example 12:
Print a Printer Spacing Chart

Introduction

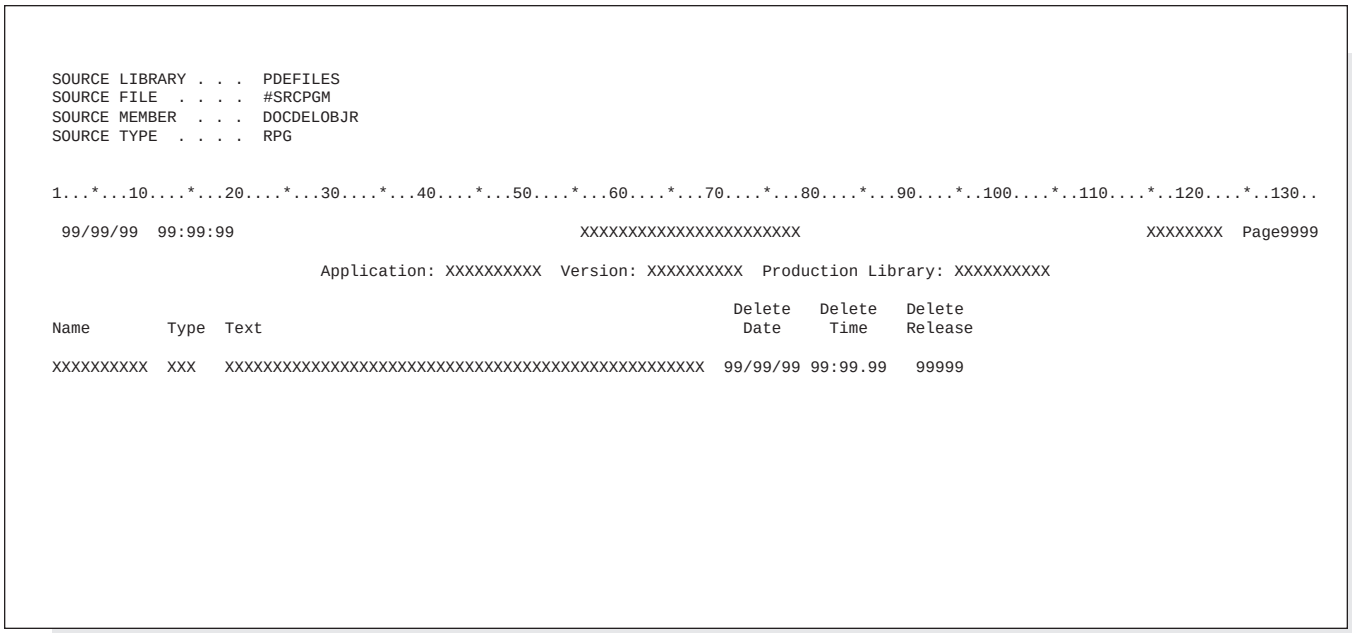
Lets say you want to add a field to a report. The specifications for the report are contained in an RPG program. You need to see what the report looks like so you can determine where to add the field. PDE/400 can print an example of the report.

- From the *PDE/400 - Documentation* menu, select Option 12 - Printer Spacing Chart - RPG.
- Fill out the screen and press Enter to start the process.

```
                        Show RPG Print Chart (RPTSMP)

Type choices, press Enter.

Member . . . . . .   docdelobjr    Name
File . . . . . .   #srcpgm__    Name
  Library  . . . . . .   pdefiles__    Name, *CURLIB, *LIBL

                                                                  Bottom
F3=Exit   F4=Prompt   F5=Refresh   F12=Cancel   F13=How to use this display
F24=More keys
```

Figure 12.1 - Printer Spacing Chart Screen

PDE will create a report showing what each report created in the program looks like. A sample of this report is shown on the following page.



```
SOURCE LIBRARY . . .  PDEFILES
SOURCE FILE  . . . .  #SRCPGM
SOURCE MEMBER  . . .  DOCDELOBJR
SOURCE TYPE  . . . .  RPG

1...*...10....*...20....*...30....*...40....*...50....*...60....*...70....*...80....*...90....*..100....*..110....*..120....*..130..

 99/99/99  99:99:99                          XXXXXXXXXXXXXXXXXXXXX                                XXXXXXXX  Page9999

                          Application: XXXXXXXXXX  Version: XXXXXXXXXX  Production Library: XXXXXXXXXX

                                                                          Delete   Delete   Delete
Name        Type  Text                                                     Date     Time    Release

XXXXXXXXXX  XXX   XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX  99/99/99 99:99.99   99999
```

Figure 12.2 - Printer Spacing Chart Report

Some of PDE/400's Features and Capabilities

- Change Management
- Track change requests
- Maintain and archive unlimited versions
- Source check out/in control system
- List and check out related objects
- Automatic rippling of changes in related versions
- Parallel development with two programmers in the same source
- Conflict warning and automatic source merging
- Source compression, archiving and unarchiving
- Central object database
- Source scan with browse or check-out of matching source
- Project Management
- Project history reporting
- Flexible multi-stage development cycle
- Intelligent Object Creation
- Automatically compile objects in logical sequence
- Recreate objects with original ownership
- Recreate objects with original attributes
- Recreate objects with original authority
- Automatic recreation of physical files and all related logical files
- Automatic mapping of data into modified files
- Automatic recreation of referenced objects
- Source compare facility
- Merging of source during parallel development
- Special Test Environment
- Automatic distribution of changes to multiple local or remote systems
- In-depth security and authorization control
- Insure that source and objects match
- Prevent changes from being overlaid
- Define as many development, test, and production environments as desired
- Meet audit requirements
- Application Documentation and cross-referencing
- Detailed project change reporting
- Library analysis summary report
- Library analysis detail report
- Objects without source report
- Source without objects report
- Objects's last use report
- Source comparison report
- Printer file layout report
- Display file layout report
- Data file layout report
- Job explosion report
- Job implosion report
- Job flowchart report
- Field cross-reference report
- Much more

The End

You have now reached the end of this tutorial. You've sampled various aspects of PDE/400 and seen how Increased Control and Increased Productivity have been brought together for you in this integrated package. With just a little bit of use, you will find that you won't want to ever again program outside of PDE's Professional Development Environment.

PDE/400 offers much more than what's been seen in these examples. Please feel free to continue exploring the system. Begin using PDE as the framework for your development and maintenance and see how much easier the entire process can be. **Once you move forward with PDE/400, you won't want to go back!**

If you have any questions or would like to discuss the concepts behind PDE, we would like to talk with you. Please call us toll-free at 1-800-588-8695.