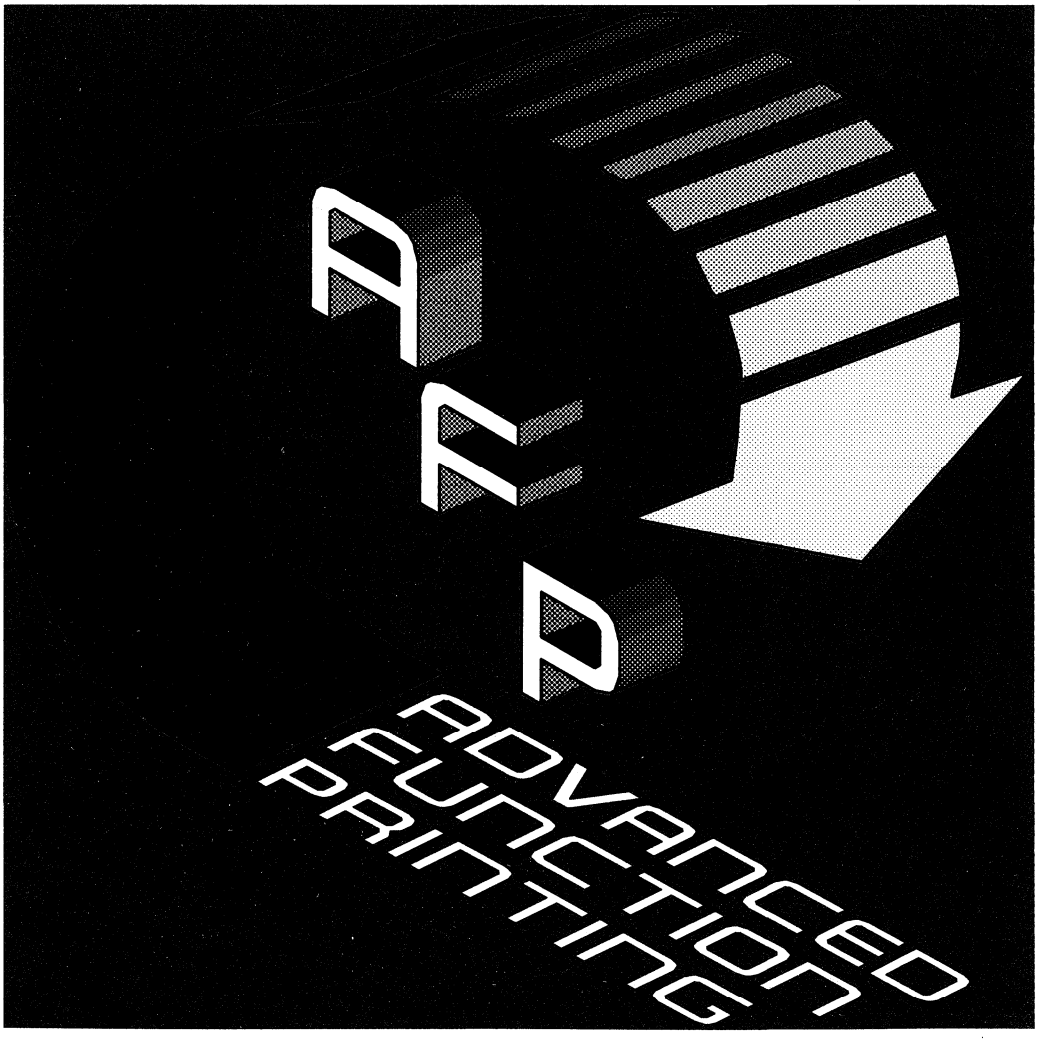
IBM SAA PrintManager
Application System/400

S544-3699-01

PrintManager
Application Programming Interface Reference

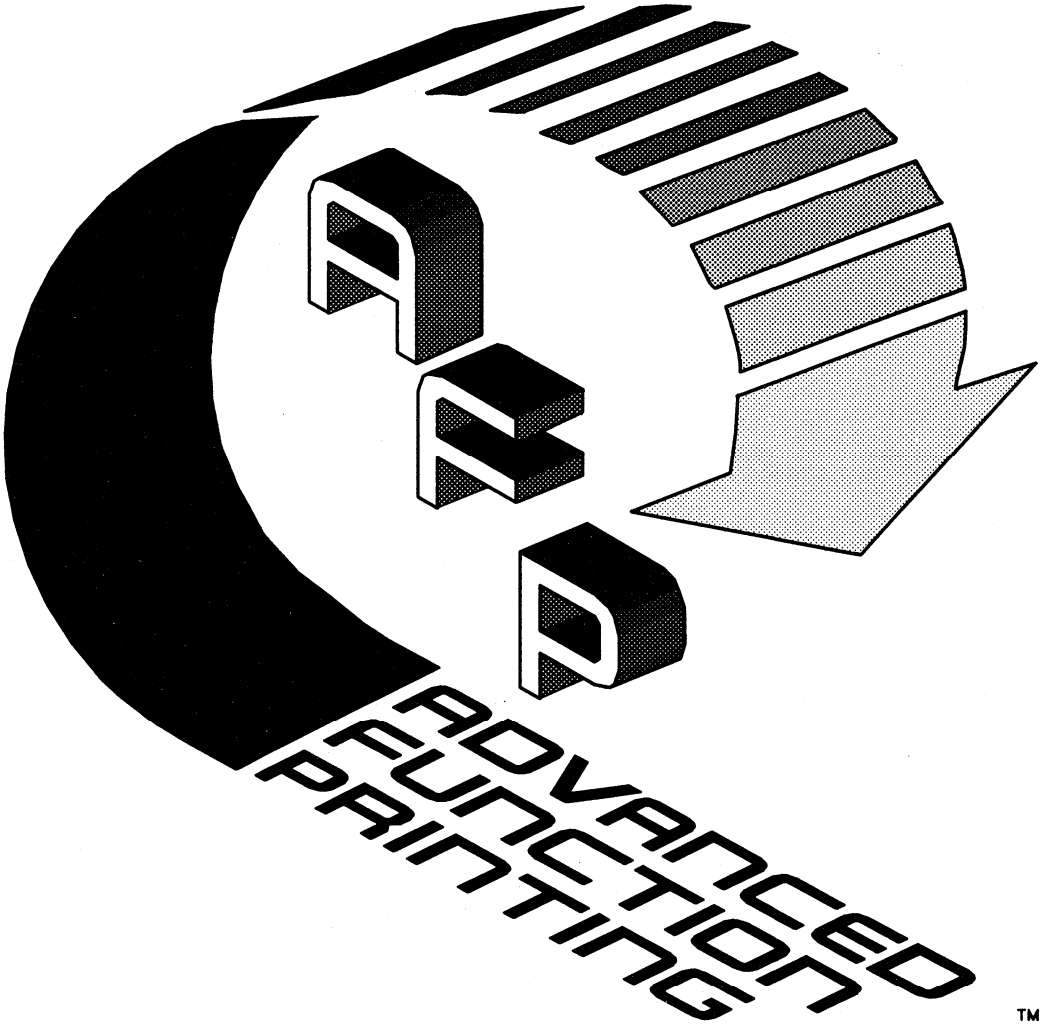


IBM

IBM SAA PrintManager
Application System/400

S544-3699-01

PrintManager
Application Programming Interface Reference



Note!

Before using this information and the product it supports, be sure to read the general information in "Notices" on page xi.

Second Edition (December 1991)

This edition replaces and makes obsolete the previous edition (S544-3699-00). This edition applies to IBM's Systems Application Architecture and to the following:

Release 1 of IBM SAA PrintManager (5688-179)
Version 2 Release 1 of Operating System/400 (5738-SS1)

and to all subsequent releases and modifications until otherwise indicated in new editions or Technical Newsletters. Be sure to use the correct edition for the level of the product.

Order publications through your IBM representative or the IBM branch office serving your locality. Publications are not stocked at the address given below.

A form for reader's comments appears at the back of this publication. If the form has been removed, address your comments to:
IBM Corporation, Department 588
6300 Diagonal Highway
Boulder, Colorado 80301-9191
U.S.A.
When you send information to IBM, you grant IBM a nonexclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

Contents

Figures

Tables

Notices

References in this publication to products or services of IBM do not imply that IBM will make them available in all countries where IBM does business or that only products or services of IBM may be used. Noninfringing equivalents may be substituted, but the user must verify that such substitutes, unless expressly designated by IBM, work correctly. No license, expressed or implied, to patents or copyrights of IBM is granted by furnishing this document.

Programming Interface

This publication is intended to help the customer to do programming with the PrintManager Application Programming Interface (API). This publication documents General-Use Programming Interface and Associated Guidance Information provided by PrintManager.

General-Use programming interfaces allow the customer to write programs that obtain the services of PrintManager.

Trademarks

The following are trademarks of the IBM Corporation:

Advanced Function Printing	AFP
Application System/400®	AS/400®
Graphical Data Display Manager	GDDM
IBM®	
IBM 3800 Page Printing Subsystem	
IBM 3812 Page Printer	
IBM 3816 Page Printer	
IBM 3820 Page Printer	
IBM 3825 Page Printer	
IBM 3827 Page Printer	
IBM 3835 Page Printer	
IBM SAA PrintManager	
MVS	
MVS/ESA	
Operating System/2®	OS/2®
Operating System/400®	OS/400®
Print Services Facility	PSF
Systems Application Architecture	SAA
Virtual Machine/Extended Architecture	VM/XA
VM	

The following are trademarks of other companies:

PostScript® Adobe Systems Incorporated

Summary of Changes

This edition of *PrintManager Application Programming Interface Reference* incorporates technical corrections to the information in the first edition (S544-3699-00). Information has been added to Chapters 3 and 8 to further describe how to customize the samples that are shipped with IBM SAA PrintManager. All technical corrections and additions are marked with the "|" symbol in the margin.

Chapter 1. Introduction

This chapter:

- Identifies the book's purpose and audience
- Gives an overview of PrintManager, its components, and their relationship
- Gives a brief overview of the Systems Application Architecture (SAA) solution
- Tells how to use this book
- Describes related publications.

Who Should Read This Book

This book provides reference and usage information for the PrintManager Application Programming Interface (API). It is intended for designers and programmers who want to create print descriptors to use with applications created with the PrintManager product implementation of the PrintManager Interface. With the API, you can create common print descriptors that contain common information for printing across the supported environments.

What PrintManager Is

PrintManager is the collective name of a group of IBM licensed programs or operating system functions designed to provide common access to printing, including Advanced Function Printing (AFP), across the supported environments. PrintManager consists of the following:

- The product implementation of the PrintManager Interface, an element of the IBM Systems Application Architecture (SAA) Common Programming Interface (CPI). This implementation allows you to write portable applications for sending print files to a system spool for printing.

  Among other benefits, the PrintManager Interface allows you to specify and validate print-option values (such as those currently specified with MVS SYSOUT JCL parameters or with the PSF/VM PSF command) from within an application in a form that is consistent across the supported environments. Applications that use the PrintManager Interface, therefore, are portable because they can be developed for one environment and used with little or no modification in another environment.

- An Application Programming Interface (API) that allows you to create print descriptors that can contain common information about printer routing, printer capabilities, and printer and job defaults. Applications that use the PrintManager Interface can use print descriptors created by the API.

  To create and maintain print descriptors, you can use either:

  - For the VM and MVS environments, the Print Descriptor Tool (PrdT), which includes a set of tags and commands, or
  - The API verbs.

- For the VM and MVS environments, a Print Request Facility (PRF) that provides the casual user with a consistent way to submit print jobs. The PRF provides both a command interface and an interactive menu, which is Interactive System Productivity Facility (ISPF) based. You can customize the interactive menu (and its functions) to meet the needs of your organization.

The PrintManager components provide capabilities to handle major printing functions within each environment, as shown in Table 1-1.

Table 1-1. PrintManager Components by Environments

PrintManager Component	MVS[1]	VM	OS/400	OS/2
PrintManager Interface	X	X	X	
Print Descriptor Tool (PrdT)	X	X		
Application Programming Interface (API) Verbs	X	X	X	
Print Request Facility (PRF)	X	X		

note:

[1] Available to TSO/E and MVS batch job users only.

PrintManager is packaged as follows:

- For MVS and VM customers, as the IBM SAA PrintManager licensed program, which includes the PrintManager Interface, Print Descriptor Tool, API verb, and PRF components.
- For OS/400 customers, the PrintManager Interface and API verb components are included within the operating system and are known as PrintManager/400.

Note: PrintManager provides C programming language support for the PrintManager Interface and API verbs in the OS/400, MVS, and VM environments. PrintManager also provides RPG and COBOL programming language support for the PrintManager Interface verbs in the OS/400 environment.

PrintManager provides print-management capability within a single environment and enhances the ability to route print jobs from one system to another, using a communication program. For example, you can send jobs from one IBM System/370 or IBM System/390 to another, using the existing networking facilities (Remote Spooling Communication Subsystem [RSCS] for VM and Job Entry Subsystem [JES2 or JES3] for MVS) or from the OS/2 environment to a VM or an MVS system, using IBM SAA Application Connection Services.

PrintManager defines a set of print options that are consistent across the supported environments, and it allows you to specify these print options within the application. PrintManager also provides the ability to:

- Select AFP resources from a system library on the printing system
- Package AFP resources with the print job (*inline*) when you send the job from one system to another. Refer to your printer-driver documentation for information on how inline resources are handled in each environment.

These print options, combined with the ability to create common, portable applications and printing definitions (with print descriptors), make possible easy and consistent access to printing across your organization. The PRF component of IBM SAA PrintManager provides additional ease of use and function for submitting print jobs.

To summarize, with PrintManager you can:

- Use the PrintManager Interface and API verbs (or, for MVS and VM, the PrdT) to create batch applications or installation-specific end-user interfaces for printing
- Use the PRF to submit print jobs.

What the SAA Solution Is

The SAA solution is based on a set of software interfaces, conventions, and protocols that provide a framework for designing and developing applications.

The SAA solution:

- Defines a Common Programming Interface that you can use to develop consistent, integrated enterprise software
- Defines Common Communication Support that you can use to connect applications, systems, networks, and devices
- Defines a Common User Access that you can use to achieve consistency in screen layout and user-interaction techniques
- Offers some applications and application development tools written by IBM.

Supported Environments

Several combinations of IBM hardware and software have been selected as SAA environments. These are environments in which IBM will manage the availability of support for all applicable SAA elements and the conformance of those elements to SAA specifications. The SAA environments are the following:

- MVS
  - Base system (TSO/E, APPC/MVS, batch)
  - CICS
  - IMS
- VM CMS
- Operating System/400 (OS/400)
- Operating System/2 (OS/2).

How to Use This Book

This book provides reference and use information for the PrintManager API, which you can use to create, update, and maintain print descriptors. Because these print descriptors are designed to work with PrintManager Interface applications, you should also be familiar with the information in *Systems Application Architecture Common Programming Interface PrintManager Reference*, S544-3698.

A system checklist is included with each PrdT command or tag function in Chapter 5, Print Descriptor Tool Reference and with each API verb description in Chapter 7, API Verb Reference. If the function is available on a particular system that PrintManager supports, that column is marked with an "X." Otherwise, the column is blank. In the sample shown in Table 1-2 on page 1-4, the function is available in MVS, VM, and OS/400, but not in OS/2.

Table 1-2. Sample System Checklist			
MVS	**VM**	**OS/400**	**OS/2**
X	X	X	

Related Documentation

For more information about SAA, PrintManager, OS/400, VM, MVS, C language, and printing (including AFP), refer to the appropriate publications listed in this section.

For the SAA Solution

An introduction to the SAA solution in general can be found in *SAA Overview*, GC26-4341.

An introduction to the Common Programming Interface can be found in *Common Programming Interface: Summary*, GC26-4675.

More detailed information on the components of the Common Programming Interface is available in the following SAA manuals:

Application Generator Reference, SC26-4355
C Reference—Level 2, SC09-1308
COBOL Reference, SC26-4354
Communications Reference, SC26-4399
Database Reference, SC26-4348
Dialog Reference, SC26-4356
FORTRAN Reference, SC26-4357
PL/I Reference, SC26-4381
Presentation Reference, SC26-4359
PrintManager Reference, S544-3698
Procedures Language Reference, SC26-4358
Procedures Language Level 2 Reference, SC24-5549
Query Reference, SC26-4349
Repository Reference, SC26-4684
RPG Reference, SC09-1286.

General programming advice can be found in *Writing Applications: A Design Guide*, SC26-4362. An introduction to the use of the AD/Cycle application development tools can be found in *AD/Cycle Concepts*, GC26-4531.

A definition of the Common User Access can be found in *Common User Access: Advanced Interface Design Guide*, SC26-4582, and *Common User Access: Basic Interface Design Guide*, SC26-4583.

More information on the Common Communications Support can be found in *Common Communications Support: Summary*, GC31-6810.

An introduction to distributed data in the SAA world can be found in *Concepts of Distributed Data*, SC26-4417.

More information on SAA system management can be found in *An Introduction to SystemView*, GC23-0576.

Ordering Information: Contact your local IBM branch office for information on how to order the above publications. They also can be obtained through an authorized IBM dealer. The entire set of SAA publications can be ordered by specifying the bill-of-forms number SBOF-1240.

For PrintManager

You should use this book with three companion publications in the PrintManager library:

IBM SAA PrintManager Overview, S544-3347, which provides general information about the IBM SAA PrintManager licensed program. It further describes the three IBM SAA PrintManager components (the API, the PrintManager Interface, and the PRF), discusses the benefits of IBM SAA PrintManager, shows how to use IBM SAA PrintManager for distributed printing, and gives examples of how to use IBM SAA PrintManager in some typical business environments.

Systems Application Architecture Common Programming Interface PrintManager Reference, S544-3698, which provides reference and use information about the PrintManager Interface in its supported environments.

IBM SAA PrintManager User's Guide, S544-3346, which provides information about the Print Request Facility (PRF) component of IBM SAA PrintManager.

The PrintManager publications are also available in softcopy form (BookManager READ versions) distributed on a CD ROM disc. This CD ROM disc is available for a nominal charge by specifying order number SK2T-1982.

For IBM SAA Application Connection Services

For general information about the IBM SAA Application Connection Services licensed program and for a list of its publications, refer to *IBM SAA Application Connection Services: General Information Manual*, GC33-0618.

For OS/400

For more information about OS/400 and the PrintManager functions within OS/400, refer to:

Application System/400 Publications Guide, GC41-9678, for a description of the information in the Application System/400 (AS/400) library.

Application System/400 System Introduction, GC41-9766, for an overview of the features and capabilities of the AS/400 system.

Application System/400 Guide to Programming for Printing, SC41-8194, for information on the printing functions in AS/400.

Application System/400 Programming: CL Reference, SBOF-0481, for reference information on CL commands.

For VM

For more information about using PrintManager in the VM operating system, refer to:

Virtual Machine/System Product CMS User's Guide, SC19-6210, *IBM Virtual Machine/Extended Architecture CMS User's Guide*, SC23-0356, or *IBM Virtual Machine/Enterprise Systems Architecture CMS User's Guide*, SC24-5460 for information on using CMS commands.

Virtual Machine/System Product CMS Command Reference, SC19-6209, *IBM Virtual Machine/Extended Architecture CMS Command Reference*, SC23-0354, or *IBM Virtual Machine/Enterprise Systems Architecture CMS Command Reference*, SC24-5461, for reference information on CMS commands.

Virtual Machine/System Product CP Command Reference, SC19-6211, *IBM Virtual Machine/Extended Architecture CP Command Reference*, SC23-0358, or *IBM Virtual Machine/Enterprise Systems Architecture CP General User Command Reference*, SC24-5433 for reference information on CP commands.

For MVS

For more information on JCL commands, refer to *MVS/ESA JCL Reference*, GC28-1829.

For C Language

For more information on writing C language applications, refer to:

IBM C/370 User's Guide, SC09-1264

Application System/400 Languages: C/400 User's Guide, SC09-1303.

For Printing

For more information about printing (including AFP), refer to the following publications:

A Guide to IBM's Advanced Function Printing, G544-3095, which contains an overview of AFP, its benefits, and the AFP licensed programs. This book also provides examples of typical AFP applications.

Advanced Function Printing: Software General Information, G544-3415, which describes IBM's AFP licensed programs and contains a list of the publications for the various AFP products.

Advanced Function Printing: Printer Summary, G544-3135, which lists the printers supported by AFP software.

Advanced Function Printing: Data Stream Reference, S544-3202, which defines the AFP data stream.

Print Services Facility/VM: Application Programming Guide, S544-3466 or *Print Services Facility/MVS: Application Programming Guide*, S544-3084, for information about PSF application programming.

Print Services Facility/VM: System Programming Guide, S544-3467 or *Print Services Facility/MVS: System Programming Guide*, SH35-0091, for information PSF system programming.

Application System/400 Guide to Programming for Printing, SC21-8194, for information on the printing functions in AS/400.

For GDDM

For information about GDDM device types, refer to *GDDM Base Programming Reference*, SC33-0332.

Chapter 2. Overview of the API

The PrintManager Application Programming Interface (API) is designed to allow you to create print descriptors, which can contain information about printer routing, printer capabilities, and printer and job defaults for printing across the environments that PrintManager supports. Applications written to the PrintManager Interface can use print descriptors created with the API.

The PrintManager API thus allows you to create a common set of printing definitions that you can use with the PrintManager PRF or with other print applications written to the PrintManager Interface. This chapter discusses the benefits of the API and provides an example of how you might use PrintManager to more effectively manage your printing operations.

PrintManager provides two ways to use the API:

1. For the VM and MVS environments, the Print Descriptor Tool (PrdT), which consists of:
   - The PrdT commands, which you use to create and manage print descriptors and print-descriptor groups. These commands include the PRD TOOL command, which you use to process input files that contain PrdT tags.
   - The PrdT tags, which are similar in format to the tags used with Generalized Markup Language (GML), and provide functions that you can use to create and manage print descriptors and print-descriptor groups.

   For more information on these commands and tags, refer to Chapter 4, Overview of the Print Descriptor Tool and Chapter 5, Print Descriptor Tool Reference. For a PrdT tag example, refer to Appendix B, Example of a PRD TOOL Input File.
2. The API verbs, which are used to write applications to create and manage print descriptors and print-descriptor groups. For more information on the API verbs, refer to Chapter 6, Using the API Verbs and Chapter 7, API Verb Reference. For information on writing C applications, refer to Appendix C, PrintManager API Verb C Language Applications. For a C language programming example, refer to Appendix D, Example of a C Language Print-Descriptor Edit Session.

Note: Print descriptors created using one code page may produce unpredictable results if used with another code page.

Benefits of the API

Print descriptors created with the API can contain information to describe where a print job will be printed, how it will be processed, and how the output will appear. Print descriptors can also be used to describe the capabilities of system components (for example, printers and printer drivers). As described in the following sections, print descriptors can define printing for your entire organization, and they can allow you to manage your printing more effectively than operating-system-specific programming methods.

Printing Definitions for an Enterprise

Print descriptors can define printing for an entire business enterprise because they can contain:

- *Printer-routing information.* Print descriptors allow you to set system-independent names for the printers in your organization. A user can access a printer by its print-descriptor name alone, without having to know whether the printer is a local or remote printer, or on which system the printer is located. For example, a print descriptor can contain all required routing information for a printer on an MVS system. A VM or an MVS user simply selects the printer by its print-descriptor name, with no additional routing information required.
- *Printer capabilities.* You can also use print descriptors to ensure that print jobs are routed to printers with the correct capabilities. If, for example, you want to print memos on an IBM 3812 Page Printer, any print descriptor that routes a job to that printer should specify simplexing, because the 3812 cannot print two-sided output.

  For print descriptors that represent the physical capabilities of the printer, any application that queries a print descriptor can use this information to help route jobs to printers with the desired capabilities.
- *Printer and job defaults.* Print descriptors can contain default values for printers or classes of print jobs. This allows you to set defaults for printers and print job standards for classes of jobs. For example, if you want to ensure that a memo is always formatted the same way, you can create print descriptors that specify the desired formatting and make only these print descriptors available for printing memos.

  PrintManager Interface application users can override default values established in a print descriptor, but only within a range of valid values specified in the print descriptor. For more information, refer to "Print Option Defaults, Validation, and Merging" on page 9-4.
- *Print option validation information,* which can be used to validate values specified for a print job. For more information, refer to "Print Option Defaults, Validation, and Merging" on page 9-4.

The API, therefore, allows you to create print descriptors for different printer capabilities and types of print jobs, which allows you to standardize classes of print jobs and easily route them to printers with the desired physical capabilities. Print descriptors, as well as PrintManager Interface applications, can be used to specify inline resources. Refer to your printer-driver documentation for information on how inline resources are handled in each environment.

Enhanced Print Management

You can, of course, use operating-system-specific programming, such as MVS Job Control Language (JCL), to manage your print operations. Print descriptors (and the print options they can contain) have the following advantages over programming languages such as JCL:

- *Consistency.* JCL is a system-specific language, but print descriptors provide a consistent way to create system-independent printing definitions easily accessed by all your organization's users.
- *Portability.* Print descriptors, like PrintManager Interface applications, are portable because they are consistent in format and in the ways they define print jobs. For example, you might create print descriptors for a job printed on IBM 3825 Page Printers printers on an MVS system. If you add more 3825s to a VM system, you can exchange print descriptors between systems.
- *Simplified, time-saving, cross-system print management.* Currently, the capability exists to change a print application. For PSF/MVS, you do this by rewriting the JCL in the application. In this case, you have to stop printing, change the application JCL, then resubmit the print job. With the API, you simply create a new print descriptor and refer to it the next time you run a PrintManager Interface application. Because different applications can refer to the same print descriptor, you can change the printer routing and job specifications for several applications without having to change the applications themselves. Your print operations can continue to run as you make changes to your printing definitions.

  Print descriptors can help you manage your print operations more easily and productively. They can provide users with system-independent names for all of the printers in your organization. You can create print descriptors that match print jobs with the capabilities of your printers and print descriptors that set print options for certain classes of print jobs, such as memos or reports.

  Because PrintManager Interface applications can refer to print descriptors, print descriptors allow you to control print options at the application level. Print descriptors also allow you to specify AFP resources inline. Inline resources can be easily routed to another system, because they are "packaged" with the print job and do not have to be sent separately or to reside on the remote system. Refer to your printer-driver documentation for information on how inline resources are handled in each environment.

By using all the benefits of the API, you can standardize your organization's printing, control access to your printers while making it easier for end users to select the correct printers and options, migrate print descriptors to other systems, and allow your printing operations to grow with little or no change. The API, working with the PrintManager Interface, provides greater flexibility in your print operations and allows you to manage your printing more productively.

Using the API in Your Organization

As you can see from "Benefits of the API" on page 2-2, the API can help you and your organization to more effectively manage printing. Here's one example of how you might use the API in a typical business situation.

For some groups within an organization, you may want to restrict certain functions both for administrative reasons and to simplify the user's tasks. For example, a group of secretaries needs to print memos in a standard page format on 8½ by 11 inch memo paper on one of several IBM 3816 Page Printers. How could you use PrintManager to meet these needs? Here's one approach you might take:

1. Create print descriptors for the 3816 printers that specify all the requirements of the organization's standard memos. For example, for both internal and external memos, you could create print descriptors that specify the printer routing and the desired output controls (electronic forms, logos as page segments, the type of memo paper, page formatting, and so forth).

   You can both simplify the secretaries' printing tasks and control the availability of printing capabilities by restricting the set of options to those defined in the print descriptors. This ensures that the group will produce memos according to the standards of the organization.

2. The secretaries use the PRF to submit the memos. With the PRF menu, a secretary can specify the number of copies and duplexing or simplexing.

What if another group with similar print needs is created in another part of the company? Let's assume that the printing will be done on IBM 3825 printers on a different system. You can easily meet this group's needs by using the original print descriptors to create a new set of print descriptors (which you can update for the 3825 capabilities and printer routings). The print application itself (the PRF) need not change.

For more examples of how different business groups can use the API and the other PrintManager functions, refer to *IBM SAA PrintManager Overview*, S544-3347.

Chapter 3. Overview of Print Descriptors

As Chapter 2, Overview of the API discusses, print descriptors provide a consistent, cross-system way to manage your organization's printing needs. This chapter:

- Tells what a print descriptor is
- Discusses print-descriptor concepts
- Provides system-specific information about print descriptors
- Tells how to set up your system to use print descriptors.

Note:

PrintManager provides two ways to use the API to create and maintain print descriptors:

1. For the MVS and VM environments, the Print Descriptor Tool (PrdT), which is discussed in Chapter 4 and Chapter 5.
2. The API verbs, which are discussed in Chapter 6 and Chapter 7.

What a Print Descriptor Is

Print descriptors contain information about print jobs and print system components. Print descriptors are stored in *print-descriptor groups*, which are files (on VM or MVS) or objects (on OS/400). As Figure 3-1 on page 3-2 shows, there are two kinds of print descriptors:

- *Standard print descriptors (StdPrds)*, which can contain print options and print-descriptor references.
- *Group-list print descriptors (GLPrds)*, which contain the list of the StdPrd groups available to a user and the search order of these groups.

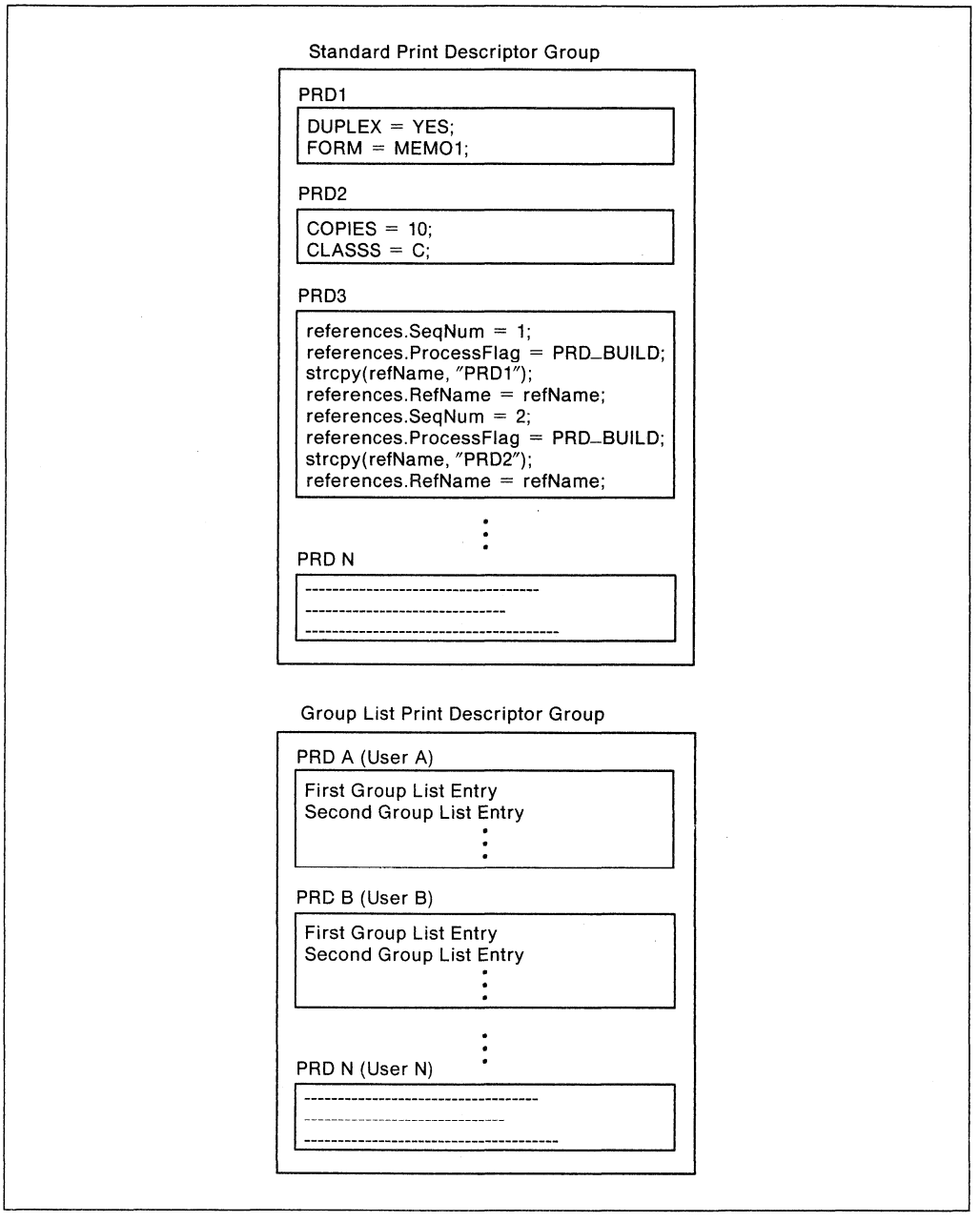


Figure 3-1. Print Descriptor Types and Contents

Contents of Standard Print Descriptors (StdPrds)

A StdPrd can contain print options, their defaults, and valid values. You can use some print options to describe where a print job will be printed, how it will be processed, and how the output will appear. You can use other options to describe the capabilities of system components (for example, printers and printer drivers). For more information, refer to Chapter 9, Print Options.

A StdPrd can also contain any number of references to other print descriptors (up to a total of 64K bytes). These references can be used to merge print-option information from the referenced standard print descriptors.

Note: A StdPrd, however, is limited to 64K bytes each of print options and print-descriptor references. A StdPrd cannot contain group-list entries.

The ability to merge print-option information allows you to share common print descriptors (and common printing definitions) across your organization. For example, if you change a print descriptor used to define a standard memo, any other print descriptor that references that print descriptor can also reflect these changes when print-option information is merged.

You can also use merging to define new valid values for a given print option that are more restrictive than those defined in the original print descriptor. For example, if the original print descriptor and a referenced print descriptor contain different ranges, the merged range will be the intersection of the two ranges. For instance, if the original print descriptor specifies a range of 1—3 and the referenced print descriptor a range of 2—4, the merged range will be 2—3.

You can use print-descriptor references to merge print-option information by using the API to build composite print descriptors from existing print descriptors. For more information on the rules for merging, refer to "How Print-Option Information Is Merged" on page 9-7. For more information on using API verbs to build print descriptors, refer to "PDBLDD (Build Descriptor) PrdBuildDescriptor" on page 7-6. For more information on using the PrdT to build print descriptors, refer to "Build a Composite Print Descriptor" on page 5-25.

Contents of Group-List Print Descriptors (GLPrds)

GLPrds allow you to:

- Refer to print descriptors by system independent names (refer to "Print-Descriptor Name Formats" on page 3-4)
- Define a search order for print descriptors.

GLPrds contain the list of the StdPrd groups available to a user and the search order of these groups (as determined by their sequence numbers). You can specify whether a group will be searched. For more information, refer to 5-26 (for the PrdT function) or 7-34 (for the API verb function).

Each *list entry* in a user's GLPrd refers to a different print-descriptor group, which can contain one or more print descriptors. Refer to "Print-Descriptor Concepts" for more information on GLPrds and print-descriptor storage.

Note: A GLPrd is limited to 64K bytes of list entries. A GLPrd cannot contain print options or print-descriptor references, and must have a type of **PRD_GROUP_LIST** (refer to "Assigning Print-Descriptor Types" on page 3-11 for more information).

Print-Descriptor Concepts

The following sections:

- Discuss print-descriptor name formats
- Provide a general model of how print descriptors (including GLPrds) are referenced and stored
- Tell how to search for, store, and delete print descriptors
- Tell how to control a user's access authority to print-descriptor groups
- Tell how to exchange print descriptors between systems
- Tell how to assign print descriptor types.

Print-Descriptor Name Formats

You can use the *universal* name format for print descriptors, because this format provides a consistent print descriptor name across systems within your organization. You can also specify a specific print-descriptor group that contains the print descriptor. You can name or refer to a print descriptor in three formats as shown in Figure 3-2.

Universal: **PRDNAME** = *Prd_Name*
Relative: **GRPALIAS** = *PrdGrp_Alias_Name* **PRDNAME** = *Prd_Name*
Exact: **GRPEXACT** = *PrdGrp_SysSpec_Name* **PRDNAME** = *Prd_Name*

Figure 3-2. Print-Descriptor Name Formats

PrintManager treats each name format as follows:

- A *universal* name refers only to a print descriptor. A print descriptor referenced this way will be searched for in the print-descriptor group search order defined in the GLPrd.
- A *relative* name refers to both a print descriptor and its print-descriptor group via an *alias* name for the group. This alias name is mapped to a physical print-descriptor group in the print-descriptor group list. An alias group name thus allows you to use system-independent group names for cross-system access of print descriptors.
- The *exact-name* format is similar to the relative-name format, except that system-specific names are used instead of group alias names.

Print-descriptor names must conform to the following rules:

- *Characters used for the name.* Any characters (letters, symbols, or numbers) are valid for universal and relative name formats. The characters of the **GRPEXACT** portion of an exact name must conform to the naming conventions for file types for the operating system.
- *Name lengths.* Universal names can be up to 32766 characters long and must contain at least one nonblank character. The length of the **GRPEXACT** portion of an exact name must conform to the length restrictions for file types for the operating system.

Notes:

1. Using universal and relative names requires a GLPrd.
2. When working on a GLPrd, you should use the exact-name format.
3. If you create print descriptors with the API verbs, it is recommended that you use upper case names for these print descriptors, because the PrdT translates all names input by a user to upper case. For more information, refer to "Invoking the Print Descriptor Tool Commands" on page 5-1.

   The PRF also translates all print-descriptor names input by a user to upper case, which you must consider if you create print descriptors for the PRF as described in Chapter 8, "Customizing the Print Request Facility" on page 8-1.

Print-Descriptor Storage

Figure 3-3 is a VM system example of print-descriptor groups for two users on the same system. As this example shows:

- A single group can contain the GLPrds for multiple users.
- Each user's GLPrd can refer to unique or common print-descriptor groups.
- The print descriptor search order is determined by each group's sequence number. In this example, the sequence numbers (10, 20, 30) allow adding groups to the search order without changing the sequence numbers of the original groups. For example, if you added a group to the second user's GLPrd with a sequence number of 25, the new group would be searched after the current second group (DEPT 583) and before the current third group (LISTINGS).

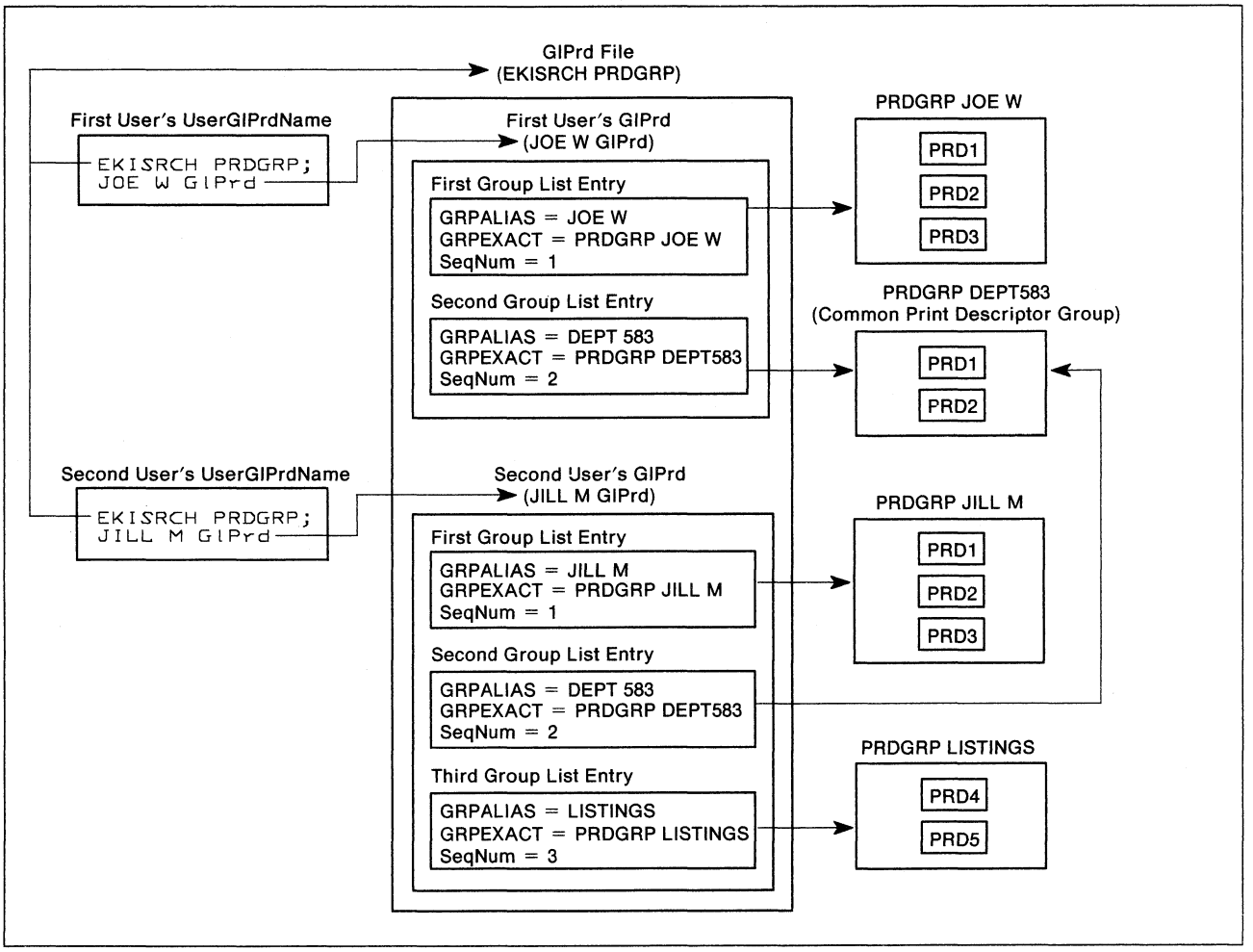


Figure 3-3. General Print-Descriptor Storage Model

As described in "What a Print Descriptor Is" on page 3-1, there are two kinds of print descriptors: standard print descriptors (StdPrds) and group-list print descriptors (GLPrds), both of which are stored in print-descriptor groups. GLPrds define the list of print-descriptor groups that a user can access using the universal or relative name formats. StdPrds can contain print options, which are used to define print-job specifications or capabilities of a system component such as a print driver or a printer. StdPrds can also contain print-descriptor references, which are used to merge print-option information from the referenced standard print descriptors.

Each individual user can have access to a unique set of print descriptors, or a group of users can share access to a common set of print descriptors. For the universal or relative name formats, the set of print descriptors that a user has access to is defined by that user's GLPrd. Each system supported by PrintManager uses a system-specific method of referencing a user's GLPrd, and this method is known as a UserGLPrdName. In the VM example in Figure 3-3 on page 3-5, this is shown as "First User's UserGLPrdName" (for user JOE W) and "Second User's UserGLPrdName" (for user JILL M). Each of these UserGLPrdNames specifies the system-specific name of the user's GLPrd and the name of the group that contains the GLPrd. For example, the UserGLPrdName for the first user specifies EKISRCH as the group that contains JOE W GLPrd. The entries in this GLPrd refer to the two StdPrd groups that the first user can access: JOE W and DEPT583. In this example, JOE W would be searched before DEPT583.

The print descriptors in these groups can be referenced in any one of the three name formats (universal, relative, or exact). For example, the first user can get print descriptor PRD1 in group JOE W group using the universal name format **PRDNAME** = PRD1, because JOE W is first in this user's group search order. To get PRD1 in group DEPT583, this user would specify the relative name format **GRPALIAS** = DEPT583 **PRDNAME** = PRD1.

Using exact-name format does not require a GLPrd, and it allows a user to specify any print descriptor in any print-descriptor group for which the user has access authority. For example, if the first user has access authority to group JILL M, that user can access print descriptor PRD1 in group JILL M by specifying **GRPEXACT** = PRDGRP JILL M **PRDNAME** = PRD1. You may, however, want to restrict a user's access authority to another user's print-descriptor groups (refer to "Access Authority to Print Descriptor Groups" on page 3-10).

Print-descriptor name formats, therefore, provide great flexibility in specifying the location of a print descriptor. Print descriptors defined in a UserGLPrdName can be accessed with the universal or relative formats, whereas the exact format allows users to access print descriptors in groups not defined in a UserGLPrdName.

Refer to Table 3-1 on page 3-7 for examples of searching for print descriptors and Table 3-2 on page 3-9 for examples of storing print descriptors on this system.

Searching for, Storing, and Deleting Print Descriptors

The following sections tell how the API searches for, stores, and deletes print descriptors, which are the basic functions that you use to manage print descriptors.

Searching for a Print Descriptor

The API searches for a print descriptor when you:

- Get an existing print descriptor for editing. For more information, refer to 5-19 (for the PrdT function) or 7-11 (for the API verb function).
- Build a composite print descriptor. For more information refer to 5-25 (for the PrdT function) or 7-6 (for the API verb function).
- Change a print descriptor from system-specific to interchange format. For more information refer to 5-35 (for the PrdT function) or 7-9 (for the API verb function). Also refer to "Exchanging Print Descriptors between Systems" on page 3-10.
- Delete a print descriptor from a print-descriptor group. For more information refer to 5-34 or 5-8 (for the PrdT functions), or 7-28 (for the API verb function).

To search for the specified print descriptor, the API follows these rules:

- If you use a universal or relative print-descriptor name, the API searches the print-descriptor groups listed in the GLPrd.
- If you use an exact print-descriptor name, the API searches the print descriptor group you specify.

Note: When you work with a GLPrd, you should use the exact-name format of the GLPrd. In VM, it is recommended that the UserGLPrdName be in upper case.

Table 3-1 shows some examples of searching for print descriptors (for example, getting a print descriptor for editing) in the example system shown in Figure 3-3 on page 3-5.

Table 3-1. Print Descriptor Search Examples

User	Print-Descriptor Name	Name Format	Search Results
Joe W.	**PRDNAME** = PRD2	universal	Print descriptor PRD2 is retrieved from group JOE W.
Joe W.	**PRDNAME** = PRD2 **GRPALIAS** = DEPT583	relative	Print descriptor PRD2 is retrieved from group DEPT583.
Joe W.	**PRDNAME** = PRD3 **GRPEXACT** = PRDGRP JOE W	exact	Print descriptor PRD3 is retrieved from group JOE W.
Joe W.	**PRDNAME** = PRD3 **GRPEXACT** = PRDGRP DEPT583	exact	No print descriptor retrieved (print descriptor PRD3 does not exist in group DEPT583). An error code will be returned.
Joe W.	**PRDNAME** = PRD4	universal	No print descriptor retrieved (print descriptor PRD4 does not exist in a group available to Joe W.). An error code will be returned.
Jill M.	**PRDNAME** = PRD2	universal	Print descriptor PRD2 is retrieved from group JILL M.
Jill M.	**PRDNAME** = PRD2 **GRPALIAS** = DEPT583	relative	Print descriptor PRD2 is retrieved from group DEPT583.
Jill M.	**PRDNAME** = PRD4	universal	Print descriptor PRD4 is retrieved from group LISTINGS.

Storing a Print Descriptor

The API stores a print descriptor when you:

- Store a print descriptor in a group. For more information refer to 5-29 (for the PrdT function) or 7-31 (for the API verb function).
- Translate a print descriptor from interchange to system-specific format. For more information refer to 5-37 (for the PrdT function) or 7-13 (for the API verb function). Also refer to "Exchanging Print Descriptors between Systems" on page 3-10.

When you store a print descriptor, you can:

- Name a new print descriptor
- Rename an existing print descriptor
- Store an existing print descriptor under its current name
- Assign or update a print descriptor type. Refer to "Assigning Print-Descriptor Types" on page 3-11 for more information.

A print-descriptor name format can specify its print-descriptor group. When you store a print descriptor, you can specify whether you are creating a new print descriptor or updating an existing one (you use the **PRD_CREATE** value when storing a new print descriptor, **PRD_UPDATE** when updating an existing print descriptor, and **PRD_CREATE_OR_UPDATE** when you want to ensure that the print descriptor is stored under the name you specify whether it already exists). Each print descriptor in a print-descriptor group must be named uniquely in that print-descriptor group. You can, however, use the same print-descriptor name across multiple print-descriptor groups. You have access only to print-descriptor groups stored on your local system.

PrintManager stores print descriptors in groups as follows:

- If you specify a relative or exact print-descriptor name when storing a print descriptor, it will be stored in the print-descriptor group you specify, regardless of the group that originally contained the print descriptor.
- If you specify a universal print-descriptor name when storing a print descriptor:
  - If a print descriptor exists in the current session as the result of a previous retrieve operation, it will be stored in the print-descriptor group where it was previously retrieved from.
  - If the print descriptor is new, the print descriptor will be stored in the first print-descriptor group listed in the GLPrd.
  - When storing a print descriptor in a group after the print descriptor is translated to system-specific format (via an import operation), the print descriptor will be stored in the first print-descriptor group listed in the GLPrd.
- If you do not specify a name when storing a print descriptor and if a print descriptor exists in the current session as the result of a previous retrieve operation, it will be stored in the print-descriptor group where it was previously stored.

Notes:

1. If you specify a new print-descriptor group name when storing a print descriptor, PrintManager creates the new print-descriptor group. You may need to add this print-descriptor group name to the GLPrd so that the print descriptor is included in the search order. For more information, refer to 5-26 (for the PrdT function) or 7-34 (for the API verb function).

   As a check, when storing a print descriptor or translating it to system-specific format (via an import operation), you can specify the following values:

   - For the print descriptor:
     - If you specify **PRD_CREATE** and the print descriptor already exists, you will get an error message.
     - If you specify **PRD_UPDATE** and the print descriptor does not exist, you will get an error message.

- If you specify **PRD_CREATE_OR_UPDATE** and the print descriptor does not exist, it will be created. If it does exist, it will be updated.

- For the print-descriptor group:
  - If you specify **PRD_AUTO_CREATE** and the print-descriptor group already exists, you will get an error message.
  - If you specify **PRD_NOAUTO_CREATE** and the print-descriptor group does not exist, you will get an error message.

2. When working with a GLPrd, you should use the exact-name format.

Table 3-2 shows some examples of storing print descriptors in the example system shown in Figure 3-3 on page 3-5.

Table 3-2. Print Descriptor Store Examples

User	Retrieve Name	Store Name	Create/Update Value	Store Results
Joe W.	**PRDNAME** = PRD2 **GRPALIAS** = DEPT583	**PRDNAME** = PRD2 **GRPALIAS** = JOE W.	**PRD_CREATE**	Print descriptor PRD2 is retrieved from group DEPT583 but not stored in group JOE W (PRD2 already exists).
Joe W.	**PRDNAME** = PRD2 **GRPALIAS** = DEPT583	**PRDNAME** = PRD2 **GRPALIAS** = JOE W.	**PRD_UPDATE**	Print descriptor PRD2 is retrieved from group DEPT583 and replaces PRD2 in group JOE W.
Joe W.	**PRDNAME** = PRD2	**PRDNAME** = PRD2	**PRD_UPDATE**	Print descriptor PRD2 is retrieved from group JOE W and is stored in the same group.
Joe W.	**PRDNAME** = PRD2 **GRPALIAS** = DEPT583	**PRDNAME** = PRD2	**PRD_UPDATE**	Print descriptor PRD2 is retrieved from group DEPT583 and is stored in the same group.
Jill M.	**PRDNAME** = PRD4	**PRDNAME** = PRD4 **GRPALIAS** = JILL M	**PRD_CREATE**	Print descriptor PRD4 is retrieved from group LISTINGS and is stored as a new print descriptor in group JILL M.
Jill M.	**PRDNAME** = PRD4 **GRPALIAS** = LISTINGS	**PRDNAME** = PRD4 **GRPALIAS** = JILL M	**PRD_CREATE_ OR_UPDATE**	Print descriptor PRD4 is retrieved from group LISTINGS and replaces PRD4 in group JILL M.

Deleting a Print Descriptor from a Group

Deleting a print descriptor from a print-descriptor group follows the same rules described in "Searching for a Print Descriptor" on page 3-6. To ensure that you delete the print descriptor you want:

- List the print descriptors in a particular group or groups
- If possible, use the exact-name format when deleting the print descriptor.

When you delete the only remaining print descriptor in a print-descriptor group, you create an empty group. If you do not plan to reuse the group, you can specify that the group will also be deleted when the print descriptor is deleted.

For more information:

- For the PrdT functions, refer to 5-11, 5-32, 5-8, and 5-34.
- For the API verb functions, refer to 7-16 and 7-28.

Note: When working with a GLPrd, you should use the exact-name format.

Access Authority to Print Descriptor Groups

Controlling a user's access authority to print-descriptor groups may be useful to an organization because it restricts the set of print system components (for example, printers and printer drivers) and print job specifications (for example, types of media) that a user or a specific group of users can access. Access restrictions can be used to make printing simpler for users (because an organization can uniquely define a user's view of printing based on individual needs) as well as to allow the organization to control a user's access to printing functions.

Access authority to print-descriptor groups is available with existing system security functions:

MVS and VM RACF can be used in both MVS and VM. In VM, normal file mode restrictions can also be used to control the access to print-descriptor groups because these groups are files in VM.

OS/400 Object command language (CL) commands can be used to control access to print-descriptor groups because these groups are PDG objects.

Exchanging Print Descriptors between Systems

Just as print-descriptor names can have different formats, print descriptors themselves can have two formats:

- *System-specific format*, which allows you to store the print descriptor in a print-descriptor group
- *Interchange format*, which allows you to exchange print descriptors between systems as files.

Print descriptors in the interchange format cannot be stored in print-descriptor groups, and print descriptors in system-specific format cannot be exchanged. Conversely, a print descriptor must be in the system-specific format to store it in a group, and it must be changed to the interchange format to exchange it.

For more information on changing print-descriptor formats:

- For the PrdT functions, refer to 5-35 and 5-37.
- For the API verb functions, refer to 7-9 and 7-13.

Note: When you receive a print descriptor in interchange format on OS/400, the file that receives the interchange print descriptor:

- Must have the exact same record length as the print descriptor in interchange format.
- Must be a physical file with FILETYPE(*DATA) and LVLCHK(*NO).

Assigning Print-Descriptor Types

When you store a print descriptor, you can identify its type. You do this in one of two ways:

1. If you are storing a print descriptor with the PrdT tags, you can use the PRDID tag. For more information, refer to 5-29.
2. If you are saving a print descriptor with the PDSAVD (Save Descriptor) verb, you can use the *PrdId* field in the **PRDDEFN2** structure. For more information, refer to 7-31.

The print-descriptor types supplied with PrintManager can refer to each other hierarchically when referring to printers and other system components. They are:

- **PRD_GROUP_LIST**, a GLPrd. This special type of print descriptor defines the print-descriptor groups you are using and the search order for the print descriptors within the groups.
- **PRD_MEDIA**, a print descriptor that can describe the attributes of the physical media you use for printing. Application programs (such as formatting programs) and system devices can use "media" print descriptors to determine the capabilities and characteristics of the media you are using.
- **PRD_PRESENTATION_DEVICE**, a print descriptor that describes attributes of both a physical device and its associated driver. Specifying a **PRD_PRESENTATION_DEVICE** print descriptor implies "early binding" of a print job to a device and its driver; that is, the application chooses the physical device and driver. Print descriptors of this type can refer to device and device driver print descriptors.
- **PRD_PRESENTATION_SERVER** and **PRD_PRESENTATION_GROUP_SERVER**, which should be used to describe the collective attributes of all the devices or servers being served. These types of print descriptors can refer to **PRD_PRESENTATION_DEVICE** print descriptors. Specifying **PRD_PRESENTATION_SERVER** or **PRD_PRESENTATION_GROUP_SERVER** print descriptors implies "late binding" of a print job to a device; that is the "server" chooses the physical device.
- **PRD_DISPLAY_DEVICE**, **PRD_PLOTTER_DEVICE**, and **PRD_PRINTER_DEVICE**, print descriptors that describe attributes of a physical display, plotter, or printer. Typically, system components like actual device drivers (and their associated print descriptors) would refer to device print descriptors to determine the capabilities and default configurations of actual devices.
- **PRD_DISPLAY_DEVICE_DRIVER**, **PRD_PLOTTER_DEVICE_DRIVER**, or **PRD_PRINTER_DEVICE_DRIVER**, print descriptors that describe attributes of a display, plotter, or printer driver. These print descriptors can refer to print descriptors that describe physical devices to determine the capabilities and default configuration of these devices.
- **PRD_PRINT_PROFILE**, which is a print-descriptor type automatically assigned to the print profiles created by PrintManager PRF users. Print descriptors of this type contain only print option values (not validation information).

Note: The print-descriptor types described above are architected values. PrintManager provides no error checking to verify that the above values are used as described. Except for print descriptors of type **PRD_GROUP_LIST**, the contents of print descriptors are not validated by PrintManager.

System-Specific Print-Descriptor Information

The following sections:

- Describe system-specific methods for storing print descriptors
- Describe system-specific conventions for referencing GLPrds.

System-Specific Print-Descriptor Storage

The following sections describe how print-descriptor groups and print descriptors in interchange format are stored in system-specific files on each of the systems supported by PrintManager. For more information on the interchange format, refer to "Exchanging Print Descriptors between Systems" on page 3-10.

MVS System Storage

On MVS systems, print-descriptor groups and print descriptors in interchange format are stored as sequential data sets only.

VM System Storage

On VM systems, print-descriptor groups and print descriptors in interchange format are stored as files in the file name file type file mode (*fn ft fm*) format.

OS/400 System Storage

On OS/400 systems:

- Print-descriptor groups are stored as objects of type "PDG" whose format is:

  ```
  library/objectname
  ```

  If *library* is omitted, OS/400 will search for the object in the user's library list (*LIBL).[1] *objectname* is required. Because print-descriptor groups are OS/400 objects, you can use commands to manage them as described below.

- A print descriptor in system interchange format is a database file named in the following fully qualified format:

  ```
  library/file(member)
  ```

  If *library* is omitted, OS/400 searches for the database file in the library list (*LIBL). If *member* is not specified, *FIRST will be used as the default.

  Note: Always specify a fully qualified file name when exporting a print descriptor to system interchange format with the PDEXPD (Export Descriptor) verb.

There are two ways to create print descriptor groups on OS/400:

1. PrintManager will automatically create a group if you specify **PRD_AUTO_CREATE** when you create the first print descriptor in a group.
2. Use the CRTPDG command to create an empty group that contains no print descriptors.

Similarly, there are two ways to delete print descriptor groups:

1. PrintManager will automatically delete a group if you specify **PRD_AUTO_DELETE** when you delete the last print descriptor in a group.
2. Use the DLTPDG command to delete a print-descriptor group.

---

[1] If you are creating, deleting or updating a PDG and *library* is omitted, it will default to the user's current library (*CURLIB).

For more information on specifying **PRD_AUTO_CREATE** and **PRD_AUTO_DELETE**:

- For the PrdT functions, refer to 5-29 and 5-37.
- For the API verb functions, refer to 7-31 and 7-13.

System-Specific UserGLPrdNames

The following sections describe how to specify the UserGLPrdName for each operating system supported by PrintManager. PrintManager provides default UserGLPrdNames to minimize system setup. For more information, refer to "How to Set Up Print Descriptors for Your System" on page 3-14.

MVS UserGLPrdNames

On MVS systems a UserGLPrdName is specified as a character string in a group list pointer data set that must be allocated to the DD name EKIGLNME for each user. If you do not provide a group list pointer data set for each user, a PrintManager default data set is used.

The recommended methods for allocating a user's group list data set to the DD name EKIGLNME are as follows:

- In a user's TSO logon procedure
- In a CLIST invoked when the user logs on.

You can also explicitly use a DD JCL statement for batch jobs or the TSO ALLOCATE statement.

This data set is a sequential data set or a member of a partitioned data set with a record length of less than 256 bytes. The UserGLPrdName takes the format:

```
GLPrdGroup;GLPrdName
```

The parameters in this character string are:

GLPrdGroup
Data set name of the group that contains the user's GLPrd. This data set name must be either a fully or partially qualified MVS data set name. Fully qualified names must be enclosed in single quotes.

GLPrdName
Name of the GLPrd.

VM UserGLPrdNames

On VM systems, a UserGLPrdName is specified by an entry in the user's LASTING GLOBALV file. It is recommended that the LASTING GLOBALV entry be in upper case, which you can do with the GLOBALV command. If you want the entry to be in lower or mixed case, use the GLOBALV command to create the entry, then edit the LASTING GLOBALV file and make the desired change.

Using the GLOBALV command, you specify the following parameters:

groupname GROUPLISTPRD *GLPrdGroupFile;GLPrdName*

The parameters in this entry are:

groupname
Name of the group used to group GLOBALV GLPrd entries for users. This name must be EKIPRD.

GROUPLISTPRD
LASTING GLOBALV variable for a GLPrd entry.

GLPrdGroupFile
File ID of the group file that contains GLPrds in the *filename filetype* format (*fn ft*). The file mode (*fm*) is optional, and an "*" can be specified as a wild card.

GLPrdName
Name of the GLPrd.

If a LASTING GLOBALV file is not found for a user, one will be created. If a GLPrd entry (identified by the *groupname* EKIPRD) is not found in the LASTING GLOBALV file, a default entry will be defined.

OS/400 UserGLPrdNames

On OS/400, a UserGLPrdName can be stored in the user's PDG profile. You control UserGLPrdNames with the following commands:

Command	Use
CHGPDGPRF	Specify a UserGLPrdName. If you do not explicitly create a UserGLPrdName with the CHGPDGPRF command, a default UserGLPrdName is defined.
DSPPDGPRF	Interactively display a UserGLPrdName.
RTVPDGPRF	Retrieve the UserGLPrdName from within a program.

For more information on these commands, refer to *Application System/400 Programming: CL Reference*, SBOF-0481.

How to Set Up Print Descriptors for Your System

In each supported environment, PrintManager supplies a sample GLPrd and StdPrd group so that little or no setup is required to use print descriptors. You can also customize these samples, or create your own GLPrds and print-descriptor groups. The following sections tell how to use the PrintManager samples and how to customize them for your organization's needs.

Using the PrintManager Samples

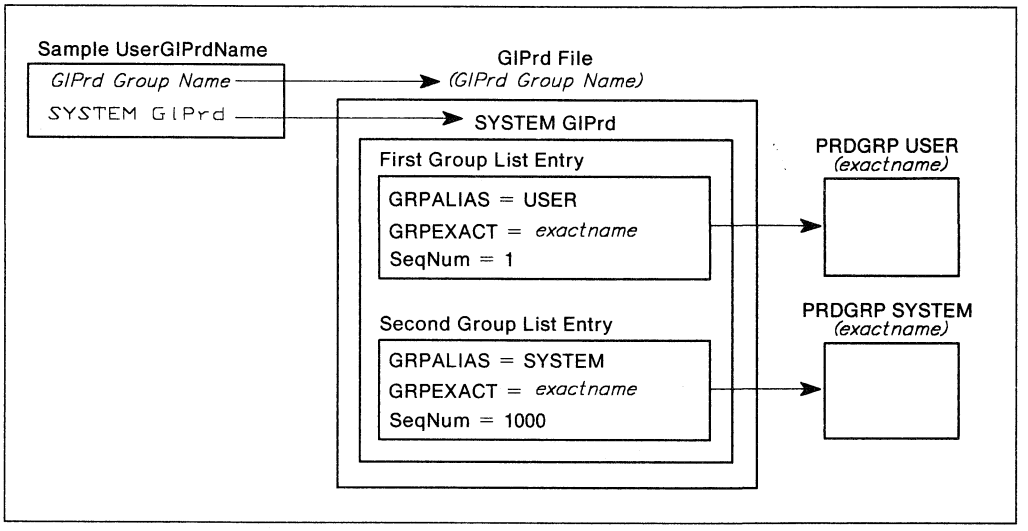


Figure 3-4. PrintManager API Samples

As Figure 3-4 shows, PrintManager supplies a sample UserGLPrdName, which names the sample GLPrd (*GLPrd Name*) and the group that contains it (*GLPrd Group Name*). The PrintManager-supplied name for *GLPrd Name* is **System Default Group List Print Descriptor**.

The PrintManager-supplied system-specific names for *GLPrd Group Name* are shown in Table 3-3.

Table 3-3. System-Specific Sample GLPrd Group Names

Operating System	Sample GLPrd Group Name
VM	$GLGRP$ EKIPM *
MVS	'EKI.$GLGRP$.EKIPM'
OS/400	QSYS/QCJGLGRP

Note: In MVS, the default group list data set which contains the PrintManager sample UserGLPrdName is EKI.$GLDS$.EKIPM. Your group list data set, however, may have been installed under a high-level qualifier different from EKI. The sample GLPrd group name may have also been changed.

The sample GLPrd has two entries, which reference two StdPrd groups:

- The first group has an alias name of USER and a sequence number of 1, so it is the first group in this user's search order. PrintManager supplies only the group name, not the group itself. This group will be automatically created for every user on your system if:
  - The group does not already exist
  - You specify **PRD_AUTO_CREATE** when storing the first print descriptor in the group. Therefore, you should specify **PRD_NOAUTO_CREATE** when adding subsequent print descriptors to the group.

  For more information on specifying **PRD_AUTO_CREATE** and **PRD_NOAUTO_CREATE**:
  - For the PrdT functions, refer to 5-29 and 5-37.

– For the API verb functions, refer to 7-31 and 7-13.

This group is intended to contain each user's unique print descriptors.

- The second group has an alias name of SYSTEM and a sequence number of 1000, so it is the second group in this user's search order. PrintManager supplies the SYSTEM group, which is intended to contain StdPrds to be available to multiple users.

The PrintManager-supplied system-specific names (*exactname*) for group USER are shown in Table 3-4.

Table 3-4. System-Specific Sample USER Print-Descriptor Group Names

Operating System	**System-Specific USER Print-Descriptor Group Name**
VM	$USRGRP$ EKIPM
MVS	$USRGRP$.EKIPM
OS/400	QCJUSRGRP

Note: For OS/400, if the library name is omitted, OS/400 searches for the print-descriptor group name in the library list (*LIBL).

The PrintManager-supplied system-specific names (*exactname*) for group SYSTEM are shown in Table 3-5.

Table 3-5. System-Specific Sample SYSTEM Print-Descriptor Group Names

Operating System	**System-Specific SYSTEM Print-Descriptor Group Name**
VM	$SYSGRP$ EKIPM *
MVS	'EKI.$SYSGRP$.EKIPM'
OS/400	QSYS/QCJSYSGRP

In the SYSTEM group, PrintManager supplies a set of StdPrds, which are sample printer definitions and profiles that can be used as a starter set by PRF users. PrintManager also supplies a set of page definitions that are used by the PRF. For more information, refer to *IBM SAA PrintManager Program Directory*.

Customizing the PrintManager Samples

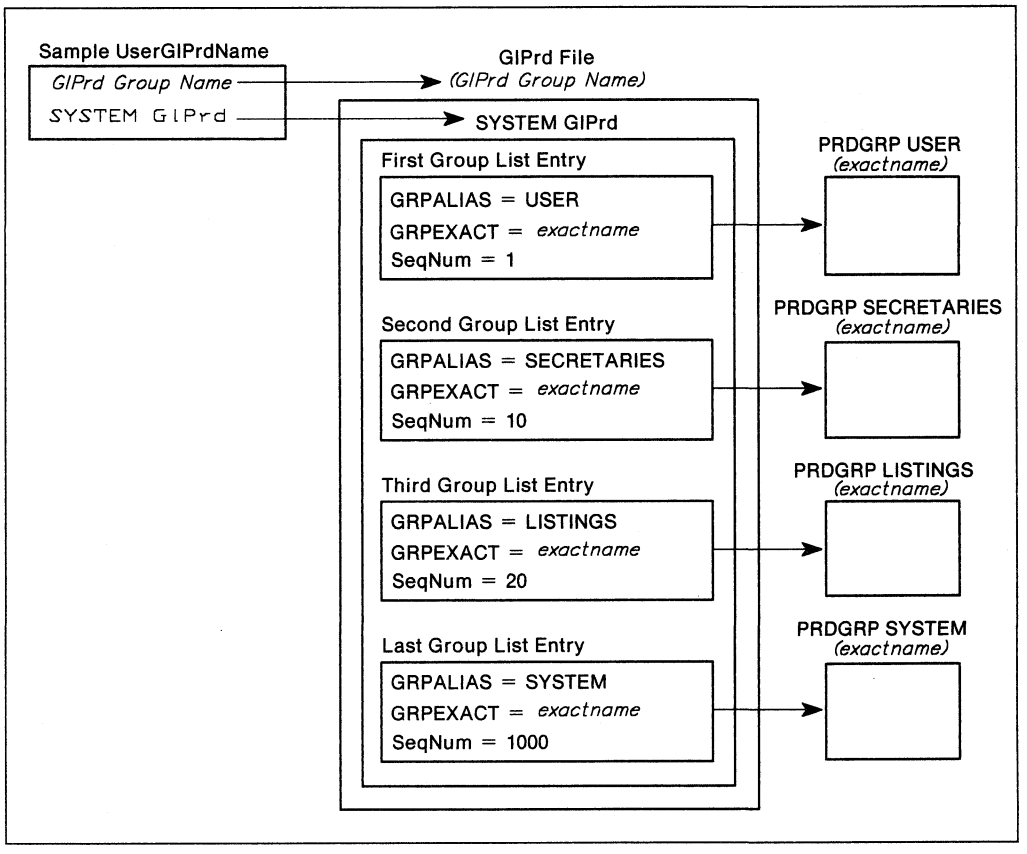


Figure 3-5. Customization of the PrintManager API Samples

Figure 3-5 shows an example of how you might customize the samples supplied with PrintManager to meet the needs of your organization. In this example:

- The first group in the GLPrd is still USER, with a sequence number of 1, so it is the first group in this user's search order. If necessary, this group can be created as described in "Using the PrintManager Samples" on page 3-15.
- The last group in the GLPrd is still SYSTEM, with a sequence number of 1000, so it is the last group in this user's search order. This group is supplied with PrintManager, as described in "Using the PrintManager Samples" on page 3-15.
- Two entries were added to the sample GLPrd. These entries reference two new groups with alias names SECRETARIES and LISTINGS. These groups could be used, for example, by a group of secretaries and a programming department. To simplify the secretaries' printing tasks, you could use RACF to restrict them from using both LISTINGS and SYSTEM. The programming department, on the other hand, might need to produce memos and reports, and might also want to use the PrintManager-supplied print descriptors in SYSTEM for some print jobs. The programming department, therefore, could have access to both LISTINGS and SYSTEM.

  All four groups are searched in the order specified by their sequence numbers. SYSTEM has a sequence number of 1000 to ensure that it is searched last and to allow other groups to be added to this system. Similarly, SECRETARIES and LISTINGS have sequence numbers of 10 and 20, respectively, to allow adding other groups to the search order without changing the relative search order of

these two groups. SECRETARIES and LISTINGS can both be created as described in "Creating a Print Descriptor" on page 6-3.

In the example shown in Figure 3-5 on page 3-17, you could either:

- Use the PrintManager-supplied names described in "Using the PrintManager Samples" on page 3-15.
- Change any of the names you want.

Figure 3-5 on page 3-17 shows one example of how to customize the PrintManager samples. Typically, you will want to customize the print descriptors in group SYSTEM for the specific needs of PRF users in your organization. For more information, refer to "Customizing the Samples for the VM Environment" and "Customizing the Samples for the MVS Environment" on page 3-31.

Note: If you create or modify print descriptors that are to be used with the PRF, ensure that these print descriptors have upper-case names. The PRF does not accept print descriptors that have lower or mixed-case names.

Customizing the Samples for the VM Environment

The VM alias name of the group containing the sample print descriptors is SYSTEM, and the default system-specific name (VM file name) of this group is $SYSGRP$ EKIPM *.

Note: This group may be installed under another file name on your system.

You can use PrdT commands to obtain information about the samples, as follows:

- To list the names of the print descriptors shipped with IBM SAA PrintManager in group SYSTEM, enter:

  ```
  PRD LIST PrdGrp_SysSpec_Name
  ```

  where *PrdGrp_SysSpec_Name* is the system-specific file name of group SYSTEM. The default file name for group SYSTEM is *$SYSGRP$ EKIPM* *, but this group may be installed under another file name on your system.

  The PRD LIST command produces the output file PRD OUT A, which contains the list of all the print descriptors in group SYSTEM. As listed in this output file (on the PRDID tag for each print descriptor), printer types have a print descriptor type of **PRD_PRESENTATION_DEVICE** and profiles have a print descriptor type of **PRD_PRINT_PROFILE**.

  Note: You can also use a PRF Values Window to display a list of the profiles that are currently available.

- To list the contents of a print descriptor in this group, enter:

  ```
  PRD QRY GRPALIAS=SYSTEM PRDNAME=Prd_Name
  ```

  where *Prd_Name* is the name of one of the print descriptors in group SYSTEM.

  The PRD QRY command also produces the output file PRD OUT A, which contains the contents of the print descriptor in PrdT tag format. You can rename PRD OUT A, edit the file to change its contents, then use the file as input to the PRD TOOL command to update the print descriptor or to create a new print descriptor.

- To list the contents of all print descriptors in group SYSTEM, enter:

  ```
  PRD DUMP PrdGrp_SysSpec_Name
  ```

where *PrdGrp_SysSpec_Name* is the system-specific file name of group SYSTEM.

The PRD DUMP command also produces the output file PRD OUT A, which contains the contents of group SYSTEM.

Printer types are used to define a set of options that reflect the printer characteristics and to define rules and valid values that reflect the printer capabilities. Profiles are used to provide a set of default values needed to submit a file for printing and do not contain rules and valid values.

The sample printer types provide general definitions for a range of logical printers, from basic definitions for line printers and IBM page printers through more complete definitions of IBM page printers. The sample profiles are designed to help PRF users easily print typical jobs for AFP printers. As described in the following sections, you can customize the samples by:

- Modifying the sample group list print descriptor (GLPrd) to reference a new print descriptor group. You can use this group to store any print descriptors that you create from the samples.
- Using the sample printer types to create printer types for your installation.
- Using the sample profiles to create profiles for your installation.

Default values can be set in either printer types or profiles. If no default (or value) is given, the job will be printed using the default defined by system services such as CMS and PSF. You should, therefore, set defaults in printer types when you want to enforce a default. For example, for printer types used to submit print jobs from VM to a printer on a remote system, the default for the **RSCSID** option should be set to the user ID of the RSCS service machine. You should *not* set defaults in printer types for options where you either:

- Want to allow the user to use a default set by system services
- Do not want to enforce a default. This is particularly important for PRF users, because defaults set in printer types cannot be blanked out.

Printer types are typically managed by the systems administrator using the PrdT, and they are stored in a common group available to all users. Profiles can be provided to all users in the same way. "Creating VM Printer Types" on page 3-24 and "Creating VM Profiles" on page 3-30 show examples of how to use the PrdT to customize the sample printer definitions and profiles.

In addition, however, individual users can create their own profiles (stored in the first group in the search order) using the PRF Save Profile function as described in *IBM SAA PrintManager User's Guide*. As the system administrator, you can also use the PRF to create profiles for groups of users. If a user saves a profile using the universal name format, the profile will be stored in the first print-descriptor group in the user's GLPrd (refer to "Storing a Print Descriptor" on page 3-7 for more information). However, if you create profiles for a group of users, you should use the exact name format to store the print descriptors in a group accessible to all users in the group. For example, you could store these print descriptors in the PrintManager-supplied SYSTEM group, or in the LOCAL group described in the following sections.

Note: The example in *IBM SAA PrintManager User's Guide* assumes that the user has access to the AFP CUTSHEET print descriptor as shipped with IBM SAA PrintManager. If you have deleted or modified this print descriptor, you

should modify the example and distribute the modified example to your users.

Modifying the VM Sample GLPrd: Before using the sample print descriptors to create print descriptors for your installation, you can first modify the sample GLPrd to add an entry to the print descriptor group list. This new entry references a new print descriptor group, which you can use to store print descriptors that you create. The sample print descriptors remain in group SYSTEM but will not be searched. You can modify the GLPrd as follows:

1. Use the PRD DUMP command to list the contents of the sample GLPrd. For example, to list the contents of the GLPrd as installed under its default VM file name, enter:

   ```
   PRD DUMP $GLGRP$ EKIPM
   ```

2. Edit the output file PRD OUT A, which contains PrdT tag format for the GLPrd as shown in Figure 3-6 on page 3-21.

```
/*-------------------------------------------*/
/*        Print Descriptor Contents          */
/*-------------------------------------------*/

:PRD. /* GRPEXACT=$GLGRP$ EKIPM C PRDNAME=System Default Group List Print
Descriptor */

/*-----------------------*/
/* Group List Information */
/*-----------------------*/

   :GRPLIST.

      :GROUP.1
         :FILEID.$USRGRP$ EKIPM
         :PROCESSFLAG.PRD_SEARCH
         :ALIAS.USER
         :DESCRIPTION.User Default Group
      :EGROUP.

      :GROUP.1000
         :FILEID.$SYSGRP$ EKIPM *
         :PROCESSFLAG.PRD_SEARCH
         :ALIAS.SYSTEM
         :DESCRIPTION.System Default Group
      :EGROUP.

   :EGRPLIST.

/*-----------------*/
/* Save Information */
/*-----------------*/

   :SAVE.GRPEXACT=$GLGRP$ EKIPM C PRDNAME=System Default Group List Print
Descriptor
      :DESCRIPTION.IBM Print Manager Group List PRD
      :PRDID.PRD_GROUP_LIST
      :CONTROLPRD.PRD_CREATE
      :CONTROLGRP.PRD_NOAUTO_CREATE
   :ESAVE. /* Level Stamp: 1991.07.17 14:43:50 110c8 */

:EPRD. /* GRPEXACT=$GLGRP$ EKIPM C PRDNAME=System Default Group List Print
Descriptor */
```

Figure 3-6. PrdT Tag Format of the VM Sample GLPrd

3. Create a list entry for the new group as shown in Figure 3-7 on page 3-23. This list entry specifies:
   - The group sequence number (500 on the GROUP tag)
   - The system-specific group name ($LCLGRP$ on the FILEID tag)
   - The group will be searched (**PRD_SEARCH** on the PROCESSFLAG tag)
   - The group alias name (LOCAL on the ALIAS tag)
   - A description of the group on the DESCRIPTION tag.

   The sequence number of group LOCAL is 500, so it will be searched after group USER. For example, if a PRF user created a profile called MYPROF, it would be

stored in the first group in the sequence (group USER). If the user then specified MYPROF for a PRF profile, PrintManager would search for MYPROF in group USER first. In addition, if the user displayed a list of available profiles on a PRF Values Window, MYPROF (and any other profiles in group USER) would be listed above the profiles in group LOCAL.

Also note that the PROCESSFLAG value for group SYSTEM is changed to **PRD_NOSEARCH**. This group is no longer included in the user's group search order, and the print descriptors in this group will *not* be listed on a PRF Values Window. Group SYSTEM and its print descriptors still exist, however, and you can still use these print descriptors as samples to customize your system (refer to "Creating VM Printer Types" on page 3-24).

As shown in the Save Information section, the value for the CONTROLPRD tag is now **PRD_UPDATE** to update the GLPrd, while the CONTROLGRP value is **PRD_NOAUTO_CREATE** (because the group that contains the GLPrd already exists).

```
/*-------------------------------------------*/
/*        Print Descriptor Contents          */
/*-------------------------------------------*/

:PRD. /* GRPEXACT=$GLGRP$ EKIPM C PRDNAME=System Default Group List Print
Descriptor */

/*-----------------------*/
/* Group List Information */
/*-----------------------*/

   :GRPLIST.

      :GROUP.1
         :FILEID.$USRGRP$ EKIPM
         :PROCESSFLAG.PRD_SEARCH
         :ALIAS.USER
         :DESCRIPTION.User Default Group
      :EGROUP.

      :GROUP.500
         :FILEID.$LCLGRP$ EKIPM *
         :PROCESSFLAG.PRD_SEARCH
         :ALIAS.LOCAL
         :DESCRIPTION.Local Default Group
      :EGROUP.


      :GROUP.1000
         :FILEID.$SYSGRP$ EKIPM *
         :PROCESSFLAG.PRD_NOSEARCH
         :ALIAS.SYSTEM
         :DESCRIPTION.System Default Group
      :EGROUP.

   :EGRPLIST.

/*-----------------*/
/* Save Information */
/*-----------------*/

   :SAVE.GRPEXACT=$GLGRP$ EKIPM C PRDNAME=System Default Group List Print
Descriptor
      :DESCRIPTION.IBM Print Manager Group List PRD
      :PRDID.PRD_GROUP_LIST
      :CONTROLPRD.PRD_UPDATE
      :CONTROLGRP.PRD_NOAUTO_CREATE
   :ESAVE. /* Level Stamp: 1991.07.17 14:45:50 110c8 */

:EPRD. /* GRPEXACT=$GLGRP$ EKIPM C PRDNAME=System Default Group List Print
Descriptor */
```

Figure 3-7. Modified PrdT Tag Format of the VM Sample GLPrd

4. Save PRD OUT A and rename it (for example, GLPRD PRDSRC A).
5. To create the updated GLPrd, enter:

   ```
   PRD TOOL GLPRD PRDSRC A
   ```

Note: The PRD DUMP and PRD QRY commands produce the output file PRD OUT A with the CONTROLGRP tag set to **PRD_NOAUTO_CREATE** and the CONTROLPRD tag set to **PRD_CREATE** to prevent you from accidentally updating an existing print descriptor. When you rename PRD OUT A and use it as input to the PRD TOOL command, specify the values for these tags as follows:

- For the CONTROLGRP tag:
  - Specify **PRD_AUTO_CREATE** when the group specified on the **GRPEXACT** or **GRPALIAS** parameter does not exist.
  - Specify **PRD_NOAUTO_CREATE** when the group specified on **GRPEXACT** or **GRPALIAS** parameter already exists, and you are just updating it.

    If you are saving multiple print descriptors in the same group and the group does not exist, the first SAVE tag must use **PRD_AUTO_CREATE**, and the following tags must use **PRD_NOAUTO_CREATE**.
- For the CONTROLPRD tag:
  - Specify **PRD_CREATE** when the print descriptor specified on the **PRDNAME** parameter does not exist in the group.
  - Specify **PRD_UPDATE** when the print descriptor specified on the **PRDNAME** parameter already exists in the group, and you are just updating it.
  - Specify **PRD_CREATE_OR_UPDATE** when the print descriptor specified on the **PRDNAME** parameter may or may not exist in the group, and it does not matter if an existing one is updated or a new one is created.

Creating VM Printer Types: Now that you have updated the GLPrd to reference group LOCAL, you can use the sample printer types to create printer types for each of your installation's printers. For example, you could customize the sample printer type AFP BASIC as follows:

1. To list the contents of this sample, enter:

   ```
   PRD QRY GRPALIAS=SYSTEM PRDNAME=AFP BASIC
   ```

2. Edit the output file PRD OUT A, which contains PrdT tag format of AFP BASIC as shown in Figure 3-8 on page 3-25.

```
/*-------------------------------------------*/
/*        Print Descriptor Contents          */
/*-------------------------------------------*/

:PRD. /* GRPEXACT=$SYSGRP$ EKIPM * PRDNAME=AFP BASIC */

/*--------------------------*/
/* Print Options Information */
/*--------------------------*/

   :SETOPT.CC
      :RULE.PRD_LIST
      :VVALUES.YES NO MACHINE
   :ESETOPT.

   :SETOPT.CLASS
      :DEFAULT.A
      :RULE.PRD_LIST
      :VVALUES.* A B C D E F G H I J K L M N O P Q R S T U V W X Y Z 0 1 2 3
4 5 6 7 8 9
   :ESETOPT.

   :SETOPT.COPIES
      :DEFAULT.1
      :RULE.PRD_RANGE
      :VVALUES.0 1 255
   :ESETOPT.

   :SETOPT.DATATYPE
      :RULE.PRD_LIST
      :VVALUES.LINE AFPDS MODCAP AFPDSLINE
   :ESETOPT.

   :SETOPT.DUPLEX
      :RULE.PRD_LIST
      :VVALUES.YES NO TUMBLE
   :ESETOPT.

   :SETOPT.FONT
      :RULE.PRD_STRING
      :VVALUES.2000
   :ESETOPT.

   :SETOPT.FORM
      :RULE.PRD_STRING
      :VVALUES.8
   :ESETOPT.

   :SETOPT.FORMDEF
      :RULE.PRD_STRING
      :VVALUES.2000
   :ESETOPT.
```

Figure 3-8 (Part 1 of 3). PrdT Tag Format of the VM Sample Printer Type AFP BASIC

```
:SETOPT.INBIN
   :RULE.PRD_LIST
   :VVALUES.1 2
:ESETOPT.

:SETOPT.LPI
   :RULE.PRD_LIST
   :VVALUES.6 8 10 12
:ESETOPT.

:SETOPT.OUTPUTID
   :RULE.PRD_STRING
   :VVALUES.17
:ESETOPT.

:SETOPT.OVERLAY
   :RULE.PRD_STRING
   :VVALUES.2000
:ESETOPT.

:SETOPT.PAGEDEF
   :RULE.PRD_STRING
   :VVALUES.2000
:ESETOPT.

:SETOPT.PAGESEG
   :RULE.PRD_STRING
   :VVALUES.2000
:ESETOPT.

:SETOPT.PRTDIRECTION
   :RULE.PRD_LIST
   :VVALUES.PORTRAIT LANDSCAPE PORTRAIT90 LANDSCAPE90
:ESETOPT.

:SETOPT.RSCSID
   :RULE.PRD_STRING
   :VVALUES.8
:ESETOPT.

:SETOPT.TRC
   :RULE.PRD_LIST
   :VVALUES.YES NO
:ESETOPT.
```

Figure 3-8 (Part 2 of 3). PrdT Tag Format of the VM Sample Printer Type AFP BASIC

```
/*------------------*/
/* Save Information */
/*------------------*/

   :SAVE.GRPEXACT=$SYSGRP$ EKIPM * PRDNAME=AFP BASIC
      :DESCRIPTION.Basic set of options for printing line data to AFP
printers
      :PRDID.PRD_PRESENTATION_DEVICE
      :CONTROLPRD.PRD_CREATE
      :CONTROLGRP.PRD_NOAUTO_CREATE
   :ESAVE. /* Level Stamp: 1991.07.18 09:30:34 110T3 */

:EPRD. /* GRPEXACT=$SYSGRP$ EKIPM * PRDNAME=AFP BASIC */
```

Figure 3-8 (Part 3 of 3). PrdT Tag Format of the VM Sample Printer Type AFP BASIC

3. Modify the contents of PRD OUT A as shown in Figure 3-9 on page 3-28. The changes are as follows:

   - The group name (and all associated comments) are changed from $SYSGRP$ to $LCLGRP$ (the system-specific name of the new group where this print descriptor will be saved).
   - The valid values for the **CLASS** option are modified for the printer classes used in the installation, and the default is set to A.
   - The rule for the **FORM** option is changed to **PRD_LIST**, and the valid values now reflect the forms used in the installation.
   - The rule for the **OUTPUTID** option is changed to **PRD_LIST**, the valid values now reflect the destinations of the printers available at the installation, and the default is set to BLDVMX.PRT001. As shown, valid values can span multiple lines.

     For the **RSCSID** option, the rule is changed to **PRD_LIST**, the only value is now NET (the user ID of the RSCS service machine), and the default is also NET. These changes to **RSCSID** enforce using the printers represented by the values for **OUTPUTID** option as remote printers.
   - The valid values for the **PRTDIRECTION** option are modified to reflect the values that are meaningful for cut sheet printers.
   - In the Save Information tags, the SAVE tag specifies the system-specific name of the group ($LCLGRP$), and the description is updated to describe the new printer type. The CONTROLPRD tag value is **PRD_CREATE** to create the new printer type, and the CONTROLGRP tag value is **PRD_AUTO_CREATE** to create the new group.

```
/*-------------------------------------------*/
/*        Print Descriptor Contents          */
/*-------------------------------------------*/

:PRD. /* GRPEXACT=$LCLGRP$ EKIPM * PRDNAME=AFP BASIC */

/*--------------------------*/
/* Print Options Information */
/*--------------------------*/

   :SETOPT.CC
      :RULE.PRD_LIST
      :VVALUES.YES NO MACHINE
   :ESETOPT.

   :SETOPT.CLASS
      :DEFAULT.A
      :RULE.PRD_LIST
      :VVALUES.A B C D
   :ESETOPT.

   :SETOPT.COPIES
      :DEFAULT.1
      :RULE.PRD_RANGE
      :VVALUES.0 1 255
   :ESETOPT.

   :SETOPT.DATATYPE
      :RULE.PRD_LIST
      :VVALUES.LINE AFPDS MODCAP AFPDSLINE
   :ESETOPT.

   :SETOPT.DUPLEX
      :RULE.PRD_LIST
      :VVALUES.YES NO TUMBLE
   :ESETOPT.

   :SETOPT.FONT
      :RULE.PRD_STRING
      :VVALUES.2000
   :ESETOPT.

   :SETOPT.FORM
      :RULE.PRD_LIST
      :VVALUES.STANDARD LETTER LEGAL
   :ESETOPT.

   :SETOPT.FORMDEF
      :RULE.PRD_STRING
      :VVALUES.2000
   :ESETOPT.
```

Figure 3-9 (Part 1 of 3). Modified PrdT Tag Format of AFP BASIC (VM Systems)

```
:SETOPT.INBIN
   :RULE.PRD_LIST
   :VVALUES.1 2
:ESETOPT.

:SETOPT.LPI
   :RULE.PRD_LIST
   :VVALUES.6 8 10 12
:ESETOPT.

:SETOPT.OUTPUTID
   :DEFAULT.BLDVMX.PRT001
   :RULE.PRD_LIST
   :VVALUES.BLDVMX.PRT001 BLDVMX.PRT002
            BLDVMX.PRT003 BLDVMX.PRT004
:ESETOPT.

:SETOPT.OVERLAY
   :RULE.PRD_STRING
   :VVALUES.2000
:ESETOPT.

:SETOPT.PAGEDEF
   :RULE.PRD_STRING
   :VVALUES.2000
:ESETOPT.

:SETOPT.PAGESEG
   :RULE.PRD_STRING
   :VVALUES.2000
:ESETOPT.

:SETOPT.PRTDIRECTION
   :RULE.PRD_LIST
   :VVALUES.PORTRAIT LANDSCAPE90
:ESETOPT.

:SETOPT.RSCSID
   :DEFAULT.NET
   :RULE.PRD_LIST
   :VVALUES.NET
:ESETOPT.

:SETOPT.TRC
   :RULE.PRD_LIST
   :VVALUES.YES NO
:ESETOPT.
```

Figure 3-9 (Part 2 of 3). Modified PrdT Tag Format of AFP BASIC (VM Systems)

```
/*------------------*/
/* Save Information */
/*------------------*/

   :SAVE.GRPEXACT=$LCLGRP$ EKIPM * PRDNAME=AFP BASIC
      :DESCRIPTION.Cut sheet printers on second floor
      :PRDID.PRD_PRESENTATION_DEVICE
      :CONTROLPRD.PRD_CREATE
      :CONTROLGRP.PRD_AUTO_CREATE
   :ESAVE. /* Level Stamp: 1991.07.18 09:30:34 110T3 */

:EPRD. /* GRPEXACT=$LCLGRP$ EKIPM * PRDNAME=AFP BASIC */
```

Figure 3-9 (Part 3 of 3). Modified PrdT Tag Format of AFP BASIC (VM Systems)

4. Save PRD OUT A and rename it (for example, FLOOR2 PRDSRC A).
5. To create group LOCAL (and to place the modified version of print descriptor AFP BASIC in that group), enter:

   ```
   PRD TOOL FLOOR2 PRDSRC A
   ```

Steps 1 through 5 are an example of how you can modify one of the sample printer types for your use, but you can use this procedure as a model to create any additional printer types you wish. Because group LOCAL now exists, for each additional print descriptor, specify **PRD_NOAUTO_CREATE** as the CONTROLGRP value.

In addition, when creating printer types, consider doing the following:

- To simplify print submission for your users, delete options that will not be used.
- You may want to change valid values for options with rules of **PRD_LIST** or **PRD_RANGE** to place controls on the use of these options at your installation. In addition, you may want to change rules and valid values for options with a rule of **PRD_STRING**.
- Many of the sample printer type contain system-specific print options for both MVS and VM. When creating printer types for your installation:
  - For printer types used to submit print jobs *from* MVS, delete the **DOCOWNER** and **RSCSID** options.
  - For printer types used to submit print jobs *from* VM, delete the **CKPTLINE**, **CKPTSEC**, **INDEX**, **LINDEX**, **LINECT**, **OUTMETHOD**, **PRTY**, **UCS**, and **WRITER** options.
  - When submitting print jobs from VM to a printer on a remote system, **RSCSID** as well as **OUTPUTID** must be specified. In addition, **RSCSID** and **DOCOWNER** are mutually exclusive. Therefore, you can specify either **RSCSID** or **DOCOWNER**, but should not specify both.

Creating VM Profiles: The sample profiles are designed to help PRF users easily print typical jobs for AFP printers. As described in "Customizing the Samples for the VM Environment" on page 3-18, you can use the PRD LIST command to get a list of these profiles. You can also use the PRD QRY or PRD DUMP commands to list the contents of a profile (or all in a group) in PrdT tag format. If you want to use any of these samples to create profiles for your installation, you can:

- Follow the same general procedure described in "Creating VM Printer Types" on page 3-24.

- Use the PRF Save Profile function to create new profiles from the samples. For more information, refer to *IBM SAA PrintManager User's Guide*.

Customizing the Samples for the MVS Environment

The MVS alias name of this group is SYSTEM, and the default system-specific name (MVS data set name) of this group is 'EKI.$SYSGRP$.EKIPM'.

Note: This group may be installed under another data set name or high level qualifier on your system.

You can use PrdT commands to obtain information about the samples, as follows:

- To list the names of the print descriptors shipped with IBM SAA PrintManager in group SYSTEM, enter:

  ```
  PRD LIST PrdGrp_SysSpec_Name
  ```

  where *PrdGrp_SysSpec_Name* is the system-specific data set name of group SYSTEM. The default data set name for group SYSTEM is 'EKI.$SYSGRP$.EKIPM', but this group may be installed under another data set name on your system.

  The PRD LIST command produces the output data set *userid*.PRD.OUT, which contains the list of all the print descriptors in group SYSTEM. As listed in this output data set (on the PRDID tag for each print descriptor), printer types have a print descriptor type of **PRD_PRESENTATION_DEVICE** and profiles have a print descriptor type of **PRD_PRINT_PROFILE**.

  Note: You can also use a PRF Values Window to display a list of the profiles that are currently available.

- To list the contents of a print descriptor in this group, enter:

  ```
  PRD QRY GRPEXACT=PrdGrp_SysSpec_Name PRDNAME=Prd_Name
  ```

  where *Prd_Name* is the name of one of the print descriptors and *PrdGrp_SysSpec_Name* is the is the system-specific data set name of group SYSTEM. The default data set name for group SYSTEM is 'EKI.$SYSGRP$.EKIPM', but this group may be installed under another data set name on your system.

  The PRD QRY command also produces the output data set *userid*.PRD.OUT, which contains the contents of the print descriptor in PrdT tag format. You can rename *userid*.PRD.OUT, edit the data set to change its contents, then use the data set as input to the PRD TOOL command to update the print descriptor or to create a new print descriptor.

- To list the contents of all print descriptors in group SYSTEM, enter:

  ```
  PRD DUMP PrdGrp_SysSpec_Name
  ```

  where *PrdGrp_SysSpec_Name* is the system-specific data set name of group SYSTEM.

  The PRD DUMP command also produces the output set *userid*.PRD.OUT, which contains the contents of group SYSTEM.

Printer types are used to define a set of options that reflect the printer characteristics and to define rules and valid values that reflect the printer capabilities. Profiles are used to provide a set of default values needed to submit a file for printing and do not contain rules and valid values.

The sample printer types provide general definitions for a range of logical printers, from basic definitions for line printers and IBM page printers through more complete definitions of IBM page printers. The sample profiles are designed to help PRF users easily print typical jobs for AFP printers. As described in the following sections, you can customize the samples by:

- Modifying the sample group list print descriptor (GLPrd) to reference a new print descriptor group. You can use this group to store any print descriptors that you create from the samples.
- Using the sample printer types to create printer types for your installation.
- Using the sample profiles to create profiles for your installation.

Default values can be set in either printer types or profiles. If no default (or value) is given, the job will be printed using the default defined by system services such as PSF. You should, therefore, set defaults in printer types when you want to enforce a default. You should *not* set defaults in printer types for options where you either:

- Want to allow the user to use a default set by system services
- Do not want to enforce a default. This is particularly important for PRF users, because defaults set in printer types cannot be blanked out.

Printer types are typically managed by the systems administrator using the PrdT, and they are stored in a common group available to all users. Profiles can be provided to all users in the same way. "Creating MVS Printer Types" on page 3-36 and "Creating MVS Profiles" on page 3-43 show examples of how to use the PrdT to customize the sample printer definitions and profiles.

In addition, however, individual users can create their own profiles (stored in the first group in the search order) using the PRF Save Profile function as described in *IBM SAA PrintManager User's Guide*. As the system administrator, you can also use the PRF to create profiles for groups of users. If a user saves a profile using the universal name format, the profile will be stored in the first print-descriptor group in the user's GLPrd (refer to "Storing a Print Descriptor" on page 3-7 for more information). However, if you create profiles for a group of users, you should use the exact name format to store the print descriptors in a group accessible to all users in the group. For example, you could store these print descriptors in the PrintManager-supplied SYSTEM group, or in the LOCAL group described in the following sections.

Note: The example in *IBM SAA PrintManager User's Guide* assumes that the user has access to the AFP CUTSHEET print descriptor as shipped with IBM SAA PrintManager. If you have deleted or modified this print descriptor, you should modify the example and distribute the modified example to your users.

Modifying the MVS Sample GLPrd: Before using the sample print descriptors to create print descriptors for your installation, you can first modify the sample GLPrd to add an entry to the print descriptor group list. This new entry references a new print descriptor group, which you can use to store print descriptors that you create. The sample print descriptors remain in group SYSTEM but will not be searched. You can modify the GLPrd as follows:

1. Use the PRD DUMP command to list the contents of the sample GLPrd. For example, to list the contents of the GLPrd as installed under its default MVS data set name, enter:

   ```
   PRD DUMP 'EKI.$GLGRP$.EKIPM'
   ```

2. Edit the output data set *userid*.PRD.OUT, which contains PrdT tag format for the GLPrd as shown in Figure 3-10 on page 3-33.

```
/*-------------------------------------------*/
/*        Print Descriptor Contents          */
/*-------------------------------------------*/

:PRD. /* GRPEXACT='EKI.$GLGRP$.EKIPM' PRDNAME=System Default Group List
Print Descriptor */

/*-----------------------*/
/* Group List Information */
/*-----------------------*/

   :GRPLIST.

      :GROUP.1
         :FILEID.$USRGRP$.EKIPM
         :PROCESSFLAG.PRD_SEARCH
         :ALIAS.USER
         :DESCRIPTION.User Default Group
      :EGROUP.

      :GROUP.1000
         :FILEID.'EKI.$SYSGRP$.EKIPM'
         :PROCESSFLAG.PRD_SEARCH
         :ALIAS.SYSTEM
         :DESCRIPTION.System Default Group
      :EGROUP.

   :EGRPLIST.

/*-----------------*/
/* Save Information */
/*-----------------*/

   :SAVE.GRPEXACT='EKI.$GLGRP$.EKIPM' PRDNAME=System Default Group List
Print Descriptor
      :DESCRIPTION.IBM Print Manager Group List PRD
      :PRDID.PRD_GROUP_LIST
      :CONTROLPRD.PRD_CREATE
      :CONTROLGRP.PRD_NOAUTO_CREATE
   :ESAVE. /* Level Stamp: 1991.07.17 14:43:50 110c8 */

:EPRD. /* GRPEXACT='EKI.$GLGRP$.EKIPM' PRDNAME=System Default Group List
Print Descriptor */
```

Figure 3-10. PrdT Tag Format of the MVS Sample GLPrd

3. Create a list entry for the new group as shown in Figure 3-11 on page 3-35. This list entry specifies:
   - The group sequence number (500 on the GROUP tag)
   - The system-specific group name ('EKI.$LCLGRP$.EKIPM' on the FILEID tag)
   - The group will be searched (**PRD_SEARCH** on the PROCESSFLAG tag)
   - The group alias name (LOCAL on the ALIAS tag)
   - A description of the group on the DESCRIPTION tag.

The sequence number of group LOCAL is 500, so it will be searched after group USER. For example, if a PRF user created a profile called MYPROF, it would be stored in the first group in the sequence (group USER). If the user then specified MYPROF for a PRF profile, PrintManager would search for MYPROF in group USER first. In addition, if the user displayed a list of available profiles on a PRF Values Window, MYPROF (and any other profiles in group USER) would be listed above the profiles in group LOCAL.

Also note that the PROCESSFLAG value for group SYSTEM is changed to **PRD_NOSEARCH**. This group is no longer included in the user's group search order, and the print descriptors in this group will *not* be listed on a PRF Values Window. Group SYSTEM and its print descriptors still exist, however, and you can still use these print descriptors as samples to customize your system (refer to "Creating MVS Printer Types" on page 3-36).

As shown in the Save Information section, the value for the CONTROLPRD tag is now **PRD_UPDATE** to update the GLPrd, while the CONTROLGRP value is **PRD_NOAUTO_CREATE** (because the group that contains the GLPrd already exists).

```
/*-----------------------------------------*/
/*        Print Descriptor Contents        */
/*-----------------------------------------*/

:PRD. /* GRPEXACT='EKI.$GLGRP$.EKIPM' PRDNAME=System Default Group List
Print Descriptor */

/*------------------------*/
/* Group List Information */
/*------------------------*/

   :GRPLIST.

      :GROUP.1
         :FILEID.$USRGRP$.EKIPM
         :PROCESSFLAG.PRD_SEARCH
         :ALIAS.USER
         :DESCRIPTION.User Default Group
      :EGROUP.

      :GROUP.500
         :FILEID.'EKI.$LCLGRP$.EKIPM'
         :PROCESSFLAG.PRD_SEARCH
         :ALIAS.LOCAL
         :DESCRIPTION.Local Default Group
      :EGROUP.


      :GROUP.1000
         :FILEID.'EKI.$SYSGRP$.EKIPM'
         :PROCESSFLAG.PRD_NOSEARCH
         :ALIAS.SYSTEM
         :DESCRIPTION.System Default Group
      :EGROUP.

   :EGRPLIST.

/*-----------------*/
/* Save Information */
/*-----------------*/

   :SAVE.GRPEXACT='EKI.$GLGRP$.EKIPM' PRDNAME=System Default Group List
Print Descriptor
      :DESCRIPTION.IBM Print Manager Group List PRD
      :PRDID.PRD_GROUP_LIST
      :CONTROLPRD.PRD_UPDATE
      :CONTROLGRP.PRD_NOAUTO_CREATE
   :ESAVE. /* Level Stamp: 1991.07.17 14:45:50 110c8 */

:EPRD. /* GRPEXACT='EKI.$GLGRP$.EKIPM' PRDNAME=System Default Group List
Print Descriptor */
```

Figure 3-11. Modified PrdT Tag Format of the MVS Sample GLPrd

4. Save *userid*.PRD.OUT and rename it (for example, *userid*.GLPRD.PRDSRC).
5. To create the updated GLPrd, enter:

   PRD TOOL '*userid*.GLPRD.PRDSRC'

Note: The PRD DUMP and PRD QRY commands produce the output data set *userid*.PRD.OUT with the CONTROLGRP tag set to **PRD_NOAUTO_CREATE** and the CONTROLPRD tag set to **PRD_CREATE** to prevent you from accidentally updating an existing print descriptor. When you rename *userid*.PRD.OUT and use it as input to the PRD TOOL command, specify the values for these tags as follows:

- For the CONTROLGRP tag:
  - Specify **PRD_AUTO_CREATE** when the group specified on the **GRPEXACT** or **GRPALIAS** parameter does not exist.
  - Specify **PRD_NOAUTO_CREATE** when the group specified on **GRPEXACT** or **GRPALIAS** parameter already exists, and you are just updating it.

    If you are saving multiple print descriptors in the same group and the group does not exist, the first SAVE tag must use **PRD_AUTO_CREATE**, and the following tags must use **PRD_NOAUTO_CREATE**.
- For the CONTROLPRD tag:
  - Specify **PRD_CREATE** when the print descriptor specified on the **PRDNAME** parameter does not exist in the group.
  - Specify **PRD_UPDATE** when the print descriptor specified on the **PRDNAME** parameter already exists in the group, and you are just updating it.
  - Specify **PRD_CREATE_OR_UPDATE** when the print descriptor specified on the **PRDNAME** parameter may or may not exist in the group, and it does not matter if an existing one is updated or a new one is created.

Creating MVS Printer Types: Now that you have updated the GLPrd to reference group LOCAL, you can use the sample printer types to create printer types for each of your installation's printers. For example, you could customize the sample printer type AFP BASIC as follows:

1. To list the contents of this sample, enter:

   ```
   PRD QRY GRPEXACT='EKI.$SYSGRP$.EKIPM' PRDNAME=AFP BASIC
   ```

2. Edit the output data set *userid*.PRD.OUT, which contains PrdT tag format of AFP BASIC as shown in Figure 3-12 on page 3-37.

```
/*-------------------------------------------*/
/*        Print Descriptor Contents          */
/*-------------------------------------------*/

:PRD. /* GRPEXACT='EKI.$SYSGRP$.EKIPM' PRDNAME=AFP BASIC */

/*--------------------------*/
/* Print Options Information */
/*--------------------------*/

   :SETOPT.CC
      :RULE.PRD_LIST
      :VVALUES.YES NO MACHINE
   :ESETOPT.

   :SETOPT.CLASS
      :DEFAULT.A
      :RULE.PRD_LIST
      :VVALUES.* A B C D E F G H I J K L M N O P Q R S T U V W X Y Z 0 1 2 3
4 5 6 7 8 9
   :ESETOPT.

   :SETOPT.COPIES
      :DEFAULT.1
      :RULE.PRD_RANGE
      :VVALUES.0 1 255
   :ESETOPT.

   :SETOPT.DATATYPE
      :RULE.PRD_LIST
      :VVALUES.LINE AFPDS MODCAP AFPDSLINE
   :ESETOPT.

   :SETOPT.DUPLEX
      :RULE.PRD_LIST
      :VVALUES.YES NO TUMBLE
   :ESETOPT.

   :SETOPT.FONT
      :RULE.PRD_STRING
      :VVALUES.2000
   :ESETOPT.

   :SETOPT.FORM
      :RULE.PRD_STRING
      :VVALUES.8
   :ESETOPT.

   :SETOPT.FORMDEF
      :RULE.PRD_STRING
      :VVALUES.2000
   :ESETOPT.
```

Figure 3-12 (Part 1 of 3). PrdT Tag Format of the MVS Sample Printer Type AFP BASIC

```
:SETOPT.INBIN
   :RULE.PRD_LIST
   :VVALUES.1 2
:ESETOPT.

:SETOPT.LPI
   :RULE.PRD_LIST
   :VVALUES.6 8 10 12
:ESETOPT.

:SETOPT.OUTPUTID
   :RULE.PRD_STRING
   :VVALUES.17
:ESETOPT.

:SETOPT.OVERLAY
   :RULE.PRD_STRING
   :VVALUES.2000
:ESETOPT.

:SETOPT.PAGEDEF
   :RULE.PRD_STRING
   :VVALUES.2000
:ESETOPT.

:SETOPT.PAGESEG
   :RULE.PRD_STRING
   :VVALUES.2000
:ESETOPT.

:SETOPT.PRTDIRECTION
   :RULE.PRD_LIST
   :VVALUES.PORTRAIT LANDSCAPE PORTRAIT90 LANDSCAPE90
:ESETOPT.

:SETOPT.RSCSID
   :RULE.PRD_STRING
   :VVALUES.8
:ESETOPT.

:SETOPT.TRC
   :RULE.PRD_LIST
   :VVALUES.YES NO
:ESETOPT.
```

Figure 3-12 (Part 2 of 3). PrdT Tag Format of the MVS Sample Printer Type AFP BASIC

```
/*------------------*/
/* Save Information */
/*------------------*/

   :SAVE.GRPEXACT='EKI.$SYSGRP$.EKIPM' PRDNAME=AFP BASIC
      :DESCRIPTION.Basic set of options for printing line data to AFP
printers
      :PRDID.PRD_PRESENTATION_DEVICE
      :CONTROLPRD.PRD_CREATE
      :CONTROLGRP.PRD_NOAUTO_CREATE
   :ESAVE. /* Level Stamp: 1991.07.18 09:30:34 110T3 */

:EPRD. /* GRPEXACT='EKI.$SYSGRP$.EKIPM' PRDNAME=AFP BASIC */
```

Figure 3-12 (Part 3 of 3). PrdT Tag Format of the MVS Sample Printer Type AFP BASIC

3. Modify the contents of *userid*.PRD.OUT as shown in Figure 3-13 on page 3-40. The changes are as follows:

   - The group name (and all associated comments) are changed from $SYSGRP$ to $LCLGRP$ (the system-specific name of the new group where this print descriptor will be saved).
   - The valid values for the **CLASS** option are modified for the printer classes used in the installation, and the default is set to A.
   - The rule for the **FORM** option is changed to **PRD_LIST**, and the valid values now reflect the forms used in the installation.
   - The rule for the **OUTPUTID** option is changed to **PRD_LIST**, the valid values now reflect the destinations of the printers available at the installation, and the default is set to BLDMVSX.PRT001. As shown, valid values can span multiple lines.

     The **RSCSID** option has been deleted, because **RSCSID** is not meaningful in MVS (the environment from which print jobs will be submitted).
   - The valid values for the **PRTDIRECTION** option are modified to reflect the values that are meaningful for cut sheet printers.
   - In the Save Information tags, the SAVE tag specifies the system-specific name of the group ($LCLGRP$), and the description is updated to describe the new printer type. The CONTROLPRD tag value is **PRD_CREATE** to create the new printer type, and the CONTROLGRP tag value is **PRD_AUTO_CREATE** to create the new group.

```
/*-------------------------------------------*/
/*        Print Descriptor Contents          */
/*-------------------------------------------*/

:PRD. /* GRPEXACT='EKI.$LCLGRP$.EKIPM' PRDNAME=AFP BASIC */

/*--------------------------*/
/* Print Options Information */
/*--------------------------*/

   :SETOPT.CC
      :RULE.PRD_LIST
      :VVALUES.YES NO MACHINE
   :ESETOPT.

   :SETOPT.CLASS
      :DEFAULT.A
      :RULE.PRD_LIST
      :VVALUES.A B C D
   :ESETOPT.

   :SETOPT.COPIES
      :DEFAULT.1
      :RULE.PRD_RANGE
      :VVALUES.0 1 255
   :ESETOPT.

   :SETOPT.DATATYPE
      :RULE.PRD_LIST
      :VVALUES.LINE AFPDS MODCAP AFPDSLINE
   :ESETOPT.

   :SETOPT.DUPLEX
      :RULE.PRD_LIST
      :VVALUES.YES NO TUMBLE
   :ESETOPT.

   :SETOPT.FONT
      :RULE.PRD_STRING
      :VVALUES.2000
   :ESETOPT.

   :SETOPT.FORM
      :RULE.PRD_LIST
      :VVALUES.STANDARD LETTER LEGAL
   :ESETOPT.

   :SETOPT.FORMDEF
      :RULE.PRD_STRING
      :VVALUES.2000
   :ESETOPT.
```

Figure 3-13 (Part 1 of 3). Modified PrdT Tag Format of AFP BASIC (MVS Systems)

```
:SETOPT.INBIN
   :RULE.PRD_LIST
   :VVALUES.1 2
:ESETOPT.

:SETOPT.LPI
   :RULE.PRD_LIST
   :VVALUES.6 8 10 12
:ESETOPT.

:SETOPT.OUTPUTID
   :DEFAULT.BLDMVSX.PRT001
   :RULE.PRD_LIST
   :VVALUES.BLDMVSX.PRT001 BLDMVSX.PRT002
            BLDMVSX.PRT003 BLDMVSX.PRT004
:ESETOPT.

:SETOPT.OVERLAY
   :RULE.PRD_STRING
   :VVALUES.2000
:ESETOPT.

:SETOPT.PAGEDEF
   :RULE.PRD_STRING
   :VVALUES.2000
:ESETOPT.

:SETOPT.PAGESEG
   :RULE.PRD_STRING
   :VVALUES.2000
:ESETOPT.

:SETOPT.PRTDIRECTION
   :RULE.PRD_LIST
   :VVALUES.PORTRAIT LANDSCAPE90
:ESETOPT.

:SETOPT.TRC
   :RULE.PRD_LIST
   :VVALUES.YES NO
:ESETOPT.
```

Figure 3-13 (Part 2 of 3). Modified PrdT Tag Format of AFP BASIC (MVS Systems)

```
/*------------------*/
/* Save Information */
/*------------------*/

   :SAVE.GRPEXACT='EKI.$LCLGRP$.EKIPM' PRDNAME=AFP BASIC
      :DESCRIPTION.Cut sheet printers on second floor
      :PRDID.PRD_PRESENTATION_DEVICE
      :CONTROLPRD.PRD_CREATE
      :CONTROLGRP.PRD_AUTO_CREATE
   :ESAVE. /* Level Stamp: 1991.07.18 09:30:34 110T3 */

:EPRD. /* GRPEXACT='EKI.$LCLGRP$.EKIPM' PRDNAME=AFP BASIC */
```

Figure 3-13 (Part 3 of 3). Modified PrdT Tag Format of AFP BASIC (MVS Systems)

4. Save *userid*.PRD.OUT and rename it (for example, *userid*.FLOOR2.PRDSRC).
5. To create group LOCAL (and to place the modified version of print descriptor AFP BASIC in that group), enter:

   PRD TOOL '*userid*.FLOOR2.PRDSRC'

Steps 1 through 5 are an example of how you can modify one of the sample printer types for your use, but you can use this procedure as a model to create any additional printer types you wish. Because group LOCAL now exists, for each additional print descriptor, specify **PRD_NOAUTO_CREATE** as the CONTROLGRP value.

In addition, when creating printer types, consider doing the following:

- To simplify print submission for your users, delete options that will not be used.
- You may want to change valid values for options with rules of **PRD_LIST** or **PRD_RANGE** to place controls on the use of these options at your installation. In addition, you may want to change rules and valid values for options with a rule of **PRD_STRING**.
- Many of the sample printer type contain system-specific print options for both MVS and VM. When creating printer types for your installation:
  - For printer types used to submit print jobs *from* MVS, delete the **DOCOWNER** and **RSCSID** options.
  - For printer types used to submit print jobs *from* VM, delete the **CKPTLINE**, **CKPTSEC**, **INDEX**, **LINDEX**, **LINECT**, **OUTMETHOD**, **PRTY**, **UCS**, and **WRITER** options.
  - When submitting print jobs from VM to a printer on a remote system, **RSCSID** as well as **OUTPUTID** must be specified. In addition, **RSCSID** and **DOCOWNER** are mutually exclusive. Therefore, you can specify either **RSCSID** or **DOCOWNER**, but should not specify both.

Creating MVS Profiles: The sample profiles are designed to help PRF users easily print typical jobs for AFP printers. As described in "Customizing the Samples for the MVS Environment" on page 3-31, you can use the PRD LIST command to get a list of these profiles. You can also use the PRD QRY or PRD DUMP commands to list the contents of a profile (or all in a group) in PrdT tag format. If you want to use any of these samples to create profiles for your installation, you can:

- Follow the same general procedure described in "Creating MVS Printer Types" on page 3-36.
- Use the PRF Save Profile function to create new profiles from the samples. For more information, refer to *IBM SAA PrintManager User's Guide*.

Chapter 4. Overview of the Print Descriptor Tool

You can use the Print Descriptor Tool (PrdT) to create and maintain print descriptors for your organization's printing needs. The PrdT consists of the PrdT commands and the PrdT tags, which are similar in format to Generalized Markup Language (GML) tags. This chapter provides an overview of the PrdT. For detailed use and reference information for the PrdT, refer to Chapter 5, Print Descriptor Tool Reference. For a PrdT tag example, refer to Appendix B, Example of a PRD TOOL Input File.

Print Descriptor Tool Commands

Table 4-1 lists the PrdT commands and their functions.

Table 4-1. Print Descriptor Tool Commands

PrdT Command	**Function**
PRD TOOL	Process PRD TOOL input files (that contain PrdT tags).
PRD LIST	Get a list of the print descriptors in a print-descriptor group or in all groups available to a user.
PRD QRY	List the contents of a print descriptor.
PRD DUMP	List the contents of all print descriptors in a group.
PRD COPY	Make a copy of a print descriptor.
PRD DEL	Delete a print descriptor from a print-descriptor group.

As Table 4-1 shows, you use the PRD TOOL command to process PRD TOOL input files. PRD TOOL input files contain PrdT tags, which are used to create new print descriptors, update existing print descriptors, or manage print descriptors and print descriptor groups.

You use the other PrdT commands to manage print descriptors and print-descriptor groups. These commands do *not* use a tag file as input, but have command parameters that allow you to specify the function you want to do.

Note: All PrdT commands consist of three parts: PRD, a required blank, followed by the operator portion of the command. For example, to enter the PRD DEL command, you must type PRD, followed by a blank, then DEL before typing in the name of the file to be deleted.

Print Descriptor Tool Tags

Figure 4-1 shows the complete set of PrdT tags as they could be entered in a PRD TOOL input file.

```
:PRD.

        /*
        The PRD and EPRD tags open and close the PRD TOOL session.
        As shown, the other session tags can be entered between
        the PRD and EPRD tags.
        The non-session tags must be entered outside
        the PRD and EPRD tags.
                                                */

        :GETPRD.PrdName
        :SETOPT.OptionName
                :RULE.Rule
                :VVALUES.Vv
                :DEFAULT.Def
        :ESETOPT.
        :DELOPTION.OptionName
        :REFLIST.
                :REFERENCE.SeqNum
                        :PRDNAME.PrdName
                        :PROCESSFLAG.ProcessFlag
                :EREFERENCE.
                :DELREF.SeqNum
        :EREFLIST.
        :PRDBUILD.
        :GRPLIST.
                :GROUP.SeqNum
                        :FILEID.PrdGrp_SysSpec_Name
                        :PROCESSFLAG.ProcessFlag
                        :ALIAS.PrdGrp_Alias_Name
                        :DESCRIPTION.Description
                :EGROUP.
                :DELGRP.SeqNum
        :EGRPLIST.
        :SAVE.PrdName
                :DESCRIPTION.Description
                :PRDID.PrdId
                :CONTROLPRD.ControlPrd
                :CONTROLGRP.ControlGrp
        :ESAVE.
:EPRD.
```

Figure 4-1 (Part 1 of 2). The Print Descriptor Tool Tags

```
:LISTPRDS.PrdGrp_SysSpec_Name
:QUERY.PrdName
:DESTROY.PrdName
        :CONTROLGRP.ControlGrp
:EDESTROY.
:EXPORT.PrdName
        :FILEID.Prd_File_Name
        :CONTROLPRD.ControlPrd
:EEXPORT.
:IMPORT.Prd_File_Name
        :PRDNAME.PrdName
        :CONTROLPRD.ControlPrd
        :CONTROLGRP.ControlGrp
:EIMPORT.
```

Figure 4-1 (Part 2 of 2). The Print Descriptor Tool Tags

As Figure 4-1 on page 4-2 shows, PrdT tags consist of *session tags* and *non-session* tags. In a PRD TOOL input file, session tags *must* be entered between the PRD and EPRD tags (which define a PRD TOOL session), while non-session tags *cannot* be entered between the PRD and EPRD tags.

Figure 4-1 on page 4-2 also shows that the PrdT tags are organized as *tag functions.* A tag function is a group of associated tags used to do a single function. For example, the tags used to set or change a print-descriptor reference are the REFLIST, REFERENCE, PRDNAME, PROCESSFLAG, EREFERENCE, and EREFLIST tags.

Table 4-2 lists the session-tag functions and the tags used for each function.

Table 4-2. Print Descriptor Tool Session-Tag Functions

PrdT Session-Tag Function	**PrdT Tags**
Get an existing print descriptor.	GETPRD
Set or change a print option.	SETOPT, RULE, VVALUES, DEFAULT, ESETOPT
Delete a print option.	DELOPTION
Set or change a print-descriptor reference.	REFLIST, REFERENCE, PRDNAME, PROCESSFLAG, EREFERENCE, EREFLIST
Delete a print-descriptor reference.	REFLIST, DELREF, EREFLIST
Build a composite print descriptor.	PRDBUILD
Set or change a group-list entry.	GRPLIST, GROUP, FILEID, PROCESSFLAG, ALIAS, DESCRIPTION, EGROUP, EGRPLIST
Delete a group-list entry.	GRPLIST, DELGRP, EGRPLIST
Store a print descriptor.	SAVE, DESCRIPTION, PRDID, CONTROLPRD, CONTROLGRP, ESAVE

Table 4-3 lists the non-session tag functions and the tags used for each function.

Table 4-3. Print Descriptor Tool Non-Session-Tag Functions

PrdT Non-Session-Tag Function	**PrdT Tags**
List print descriptors.	LISTPRDS
List the contents of a print descriptor.	QUERY
Delete a print descriptor.	DESTROY, CONTROLGRP, EDESTROY
Translate a print descriptor to interchange format.	EXPORT, FILEID, CONTROLPRD, EEXPORT
Translate a print descriptor to system-specific format.	IMPORT, PRDNAME, CONTROLPRD, CONTROLGRP, EIMPORT

As described above, session tags must be entered between PRD and EPRD tags while non-session tags cannot be entered between the PRD and EPRD tags. You can, however, code session and non-session-tag functions within the same PRD TOOL input file. For example, you might want to list the print descriptors in a group, then update one of the print descriptors you listed. To do so in a single input file, you would:

- Use the LISTPRDS non-session tag to list the print descriptors in the group.
- Use the PRD tag to open a PRD TOOL session.
- Use the GETPRD tag to get a print descriptor for updating.
- Make the desired changes to the print descriptor. For example, you might change or delete print options.
- Store the updated print descriptor.
- Use the EPRD tag to close the PRD TOOL session.
- Use the LISTPRDS tag to ensure that the print descriptor was correctly stored.

Chapter 5. Print Descriptor Tool Reference

This chapter provides use and reference information for the Print Descriptor Tool (PrdT) commands and tags. For an overview of the PrdT, refer to Chapter 4.

Print Descriptor Tool Command Reference

The following sections provide detailed information on the PrdT commands, including:

- How to use the commands
- The format used to describe these commands
- Command reference information.

Using the Print Descriptor Tool Commands

The following sections:

- Tell how to invoke the PrdT commands
- Describe the output files from these commands
- Tell how to handle errors caused while using the commands.

Invoking the Print Descriptor Tool Commands

You enter a PrdT command from the CMS or TSO/E command line. You enter the PRD TOOL command as PRD TOOL *filename*, where *filename* is the file or data set name of the PRD TOOL input file. For example, in VM: `PRD TOOL PRD1 PRDSRC`. For more information on input files, see "Creating PRD TOOL Input Files" on page 5-14.

You enter the other PrdT commands as a command and its parameters. For example, to list the contents of a print descriptor with the universal name PRD3812A, enter `PRD QRY **PRDNAME**=PRD3812A`.

Notes:

1. The commands in this section translate all print-descriptor names to uppercase. Therefore, you can enter the print-descriptor name in upper, lower, or mixed case, but the stored name of the print descriptor will be in upper case.
2. In MVS, the total length of a command and its parameters cannot exceed 100 characters.

Print Descriptor Tool Command Output Files

The PRD LIST, PRD QRY, and PRD DUMP commands produce an output file with the file attributes shown in Table 5-1 on page 5-2. The PRD TOOL command can also produce the same type of output file when used with an input file that contains a LISTPRDS or QUERY tag.

Table 5-1. Print Descriptor Tool Command Output File Attributes			
Operating System	**File Name**	**Record Format**	**Maximum Record Length**
VM	PRD OUT A	variable (recfm = V)	variable
MVS	*userid*.PRD.OUT	variable (recfm = VB)	255

If the output file is pre-allocated in MVS, it must have the file attributes shown in Table 5-1. It is not necessary, however, to pre-allocate the file.

Notes:

1. The PrdT output file is overwritten each time you invoke a command that creates the file. Therefore, if you want to save a particular version of the output file, you must rename it.
2. The output file contains tags PRD TOOL input file format. The output file also has the same file attributes as a PRD TOOL input file. Therefore, the output file can be used as a PRD TOOL input file, but it must be renamed before doing so.

Print Descriptor Tool Command Errors

Command errors are indicated by PrdT error messages. Refer to "Print Descriptor Tool Messages" on page E-1 for more information on these messages.

Format of the Command Descriptions

Each of the command descriptions in "Command Reference" on page 5-4 contains these sections:

Command name The name of the command.

Supported system checklist
The operating systems supported by the command.

Function A brief description of the command function.

Syntax Syntax of the command. This syntax is represented as in the example shown in Figure 5-1 on page 5-3.

Parameters Descriptions of each command parameter. Command parameters are either:

- Variables (for example, the *PrdGrp_SysSpec_Name* parameter of the PRD DUMP command, which is shown in Figure 5-4 on page 5-10)
- Composed of a keyword, operator, and variable (for example, the **GRPALIAS** = *Source_PrdGrp_Alias_Name* parameter of the PRD COPY command, which is shown in Figure 5-1 on page 5-3).

Usage Use information for the command.

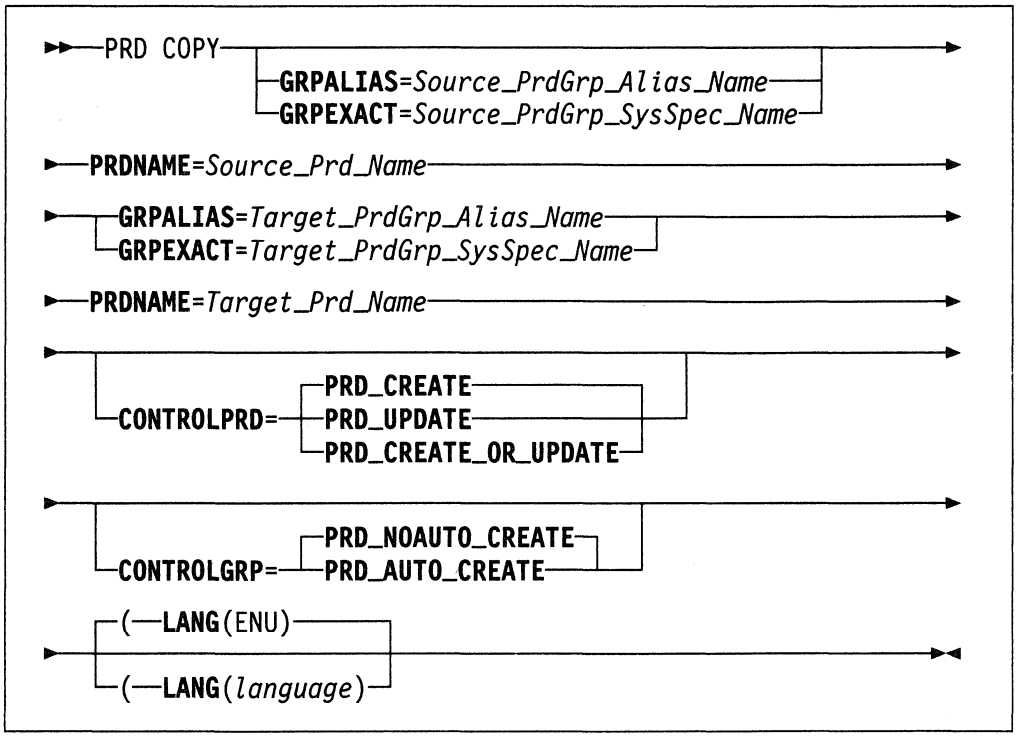


Figure 5-1. Example: Syntax of the PRD COPY Command

As shown in Figure 5-1, command syntax is represented as follows:

- Arrows are used to show the command and parameter sequences.
- Single parameters are shown on a single line. Required parameters are shown on the center line and optional parameters are shown below the center line.
- For choices among groups of parameters, the parameters are shown on parallel lines. Optional choice groups are shown below the center line. If a required choice group has a default, it is shown above the center line.

As this example shows:

- The source print-descriptor group name is optional; if chosen, it must be entered first. You can enter either an alias name or an exact name for the group.
- The source print descriptor name is required, and must be entered next.
- The target print-descriptor group name is required, and must be entered next. You can enter either an alias name or an exact name for the group.
- The source print descriptor name is required, and must be entered next.
- The **CONTROLPRD** parameter is optional; if chosen, it must be entered next. The default value is **PRD_CREATE**.
- The **CONTROLGRP** parameter is optional; if chosen, it must be entered next. The default value is **PRD_NOAUTO_CREATE**
- The **LANG** parameter is optional; if chosen, it must be entered next and is preceded by a required left parentheses. The PrdT default value is ENU.

Command Reference

This section provides reference information on the PRD COPY, PRD DEL, PRD DUMP, PRD LIST, PRD QRY, and PRD TOOL commands.

PRD COPY

MVS	VM	OS/400	OS/2
X	X		

Function Make a copy of a print descriptor.

Syntax

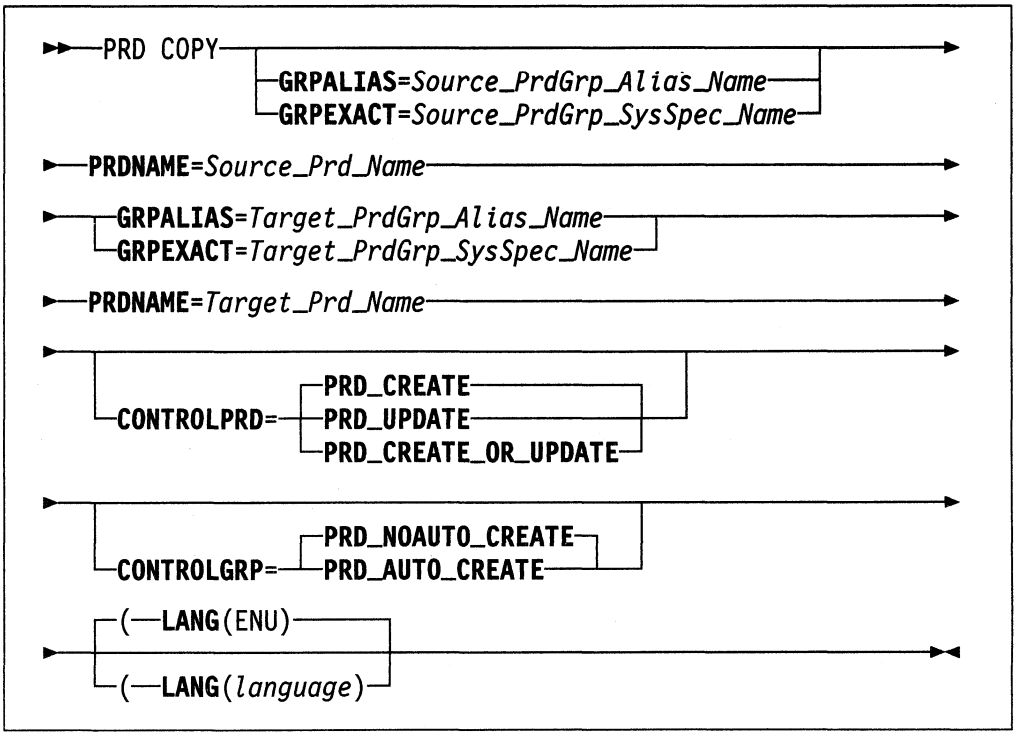


Figure 5-2. Syntax of the PRD COPY Command

Parameters

Source Print Descriptor Name

GRPALIAS = *Source_PrdGrp_Alias_Name*
The alias name of the source print-descriptor group.

GRPEXACT = *Source_PrdGrp_SysSpec_Name*
The exact name of the source print-descriptor group.

PRDNAME = *Source_Prd_Name*
The source print-descriptor name.

Target Print Descriptor Name

GRPALIAS = *Target_PrdGrp_Alias_Name*
The alias name of the target print-descriptor group.

GRPEXACT = *Target_PrdGrp_SysSpec_Name*
The exact name of the target print-descriptor group.

PRDNAME = *Target_Prd_Name*
The target print-descriptor name.

CONTROLPRD
Specifies action for the target print descriptor as follows:

PRD_CREATE Create a new print descriptor. An error occurs if the specified print descriptor already exists.

PRD_UPDATE Update an existing print descriptor. An error occurs if the specified print descriptor does not exist.

PRD_CREATE_OR_UPDATE
Either create a new print descriptor or update an existing print descriptor.

The default value is **PRD_CREATE**.

CONTROLGRP

Specifies the action for the target print-descriptor group as follows:

PRD_AUTO_CREATE Create a new print-descriptor group in which to store the print descriptor. An error occurs if the group already exists.

PRD_NOAUTO_CREATE
Store the print descriptor in an existing group. An error occurs if the group does not exist.

The default value is **PRD_NOAUTO_CREATE**.

LANG(*language***)**
Specifies the language used to display PrdT messages. Values for this option are standard three-letter abbreviations. For example, ENU (American English), FRS (Swiss French), ESP (Spanish), and so forth. The **LANG** parameter is optional, and the PrdT default value is ENU.

Your location may have installed two or more languages or modified the languages supplied. See your system administrator for specific languages supported at your location and their abbreviations.

Usage: Use the PRD COPY command to copy a source print descriptor to a target print descriptor. When using the PRD COPY command, consider the following guidelines:

- *To copy a print descriptor to a new print descriptor in a new print-descriptor group.* Specify **PRD_CREATE** as the **CONTROLPRD** value to create the new print descriptor (and to ensure that the print descriptor does not already exist). Specify **PRD_AUTO_CREATE** as the **CONTROLGRP** value to create the new print-descriptor group (and to ensure that the group does not already exist). Finally, use the Set or Change a Group-List Entry function (refer to page 5-26) to add the new group name to the group-list print descriptor so that the print descriptor is included in the search order.
- *To copy a print descriptor to a new print descriptor in an existing print-descriptor group.* Specify **PRD_CREATE** as the **CONTROLPRD** value to create the new print descriptor (and to ensure that the print descriptor does not already exist). Specify **PRD_NOAUTO_CREATE** as the **CONTROLGRP** value to store the new print descriptor in the existing print-descriptor group (and to ensure that the group exists and that a new group will not be created). If you are making a copy of a print descriptor within the same group, ensure that you use the

PRDNAME = *Target_Prd_Name* parameter to specify a different print-descriptor name for the target print descriptor

- *To copy a print descriptor to an existing print descriptor.* Specify **PRD_UPDATE** as the **CONTROLPRD** value to update (overwrite) the print descriptor (and to ensure that the print descriptor exists and that a new one will not be created). Specify **PRD_NOAUTO_CREATE** as the **CONTROLGRP** value to store the existing print descriptor in the existing print-descriptor group (and to ensure that the group exists and that a new group will not be created). If you are making a copy of a print descriptor within the same group, ensure that you use the **PRDNAME** = *Target_Prd_Name* parameter to specify a different print-descriptor name for the target print descriptor
- The **CONTROLPRD** and **CONTROLGRP** keywords are optional. If they are not specified, however, the defaults will be used.

PRD DEL

MVS	VM	OS/400	OS/2
X	X		

Function Delete a print descriptor from a print-descriptor group.

Syntax

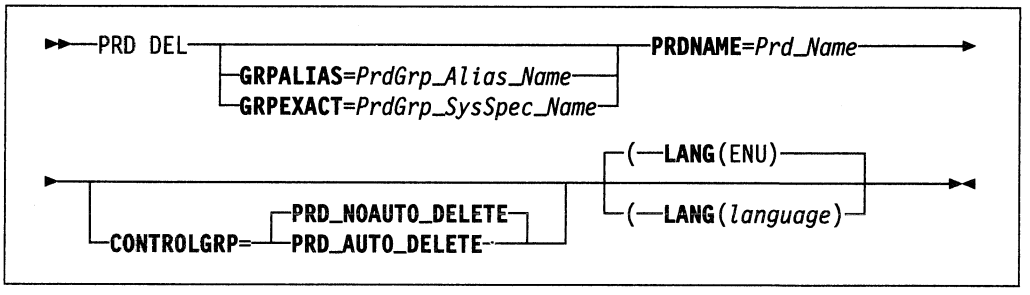


Figure 5-3. Syntax of the PRD DEL Command

Parameters

GRPALIAS
The alias name of the print-descriptor group.

GRPEXACT
The exact name of the print-descriptor group.

PRDNAME
The print-descriptor name.

Note: You may want to specify the **GRPEXACT** keyword to ensure that you delete the correct print descriptor.

CONTROLGRP
Specifies the action when the print descriptor to be deleted is the only print descriptor in the group, as follows:

PRD_AUTO_DELETE If deleting the print descriptor produces an empty print-descriptor group, delete the group.

PRD_NOAUTO_DELETE
Do not delete an empty group.

The default value is **PRD_NOAUTO_DELETE**.

LANG(*language*)
Specifies the language used to display PrdT messages. Values for this option are standard three-letter abbreviations. For example, ENU (American English), FRS (Swiss French), ESP (Spanish), and so forth. The **LANG** parameter is optional, and the PrdT default value is ENU.

Your location may have installed two or more languages or modified the languages supplied. See your system administrator for specific languages supported at your location and their abbreviations.

Usage: Use the PRD DEL command to delete a print descriptor from a print-descriptor group. PrintManager searches for the print descriptor to be deleted according to the parameters you specify on the PRD DEL command. To ensure you delete the print descriptor you want:

- Use the List Print Descriptors tag function or the PRD LIST command to list the print descriptors in a specific group or groups
- If possible, specify the **GRPEXACT** keyword.

The **CONTROLGRP** keyword is optional. If it is not specified, however, the default will be used.

PRD DUMP

MVS	VM	OS/400	OS/2
X	X		

Function List the contents of all print descriptors in a group.

Syntax

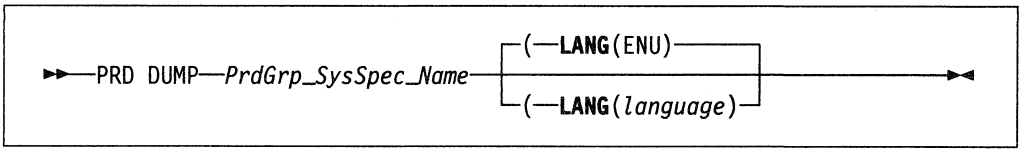


Figure 5-4. Syntax of the PRD DUMP Command

Parameters

PrdGrp_SysSpec_Name
Specifies the system-specific name of the group containing the print descriptors whose contents are to be listed.

LANG(*language***)**
Specifies the language used to display PrdT messages. Values for this option are standard three-letter abbreviations. For example, ENU (American English), FRS (Swiss French), ESP (Spanish), and so forth. The **LANG** parameter is optional, and the PrdT default value is ENU.

Your location may have installed two or more languages or modified the languages supplied. See your system administrator for specific languages supported at your location and their abbreviations.

Usage: Use the PRD DUMP command to create an output file containing the contents of the all print descriptors in the group specified on the *PrdGrp_SysSpec_Name* variable. The output file contains the print descriptor contents in PrdT-tag format. You can, therefore, rename the output file, edit it, and use the renamed file as a PRD TOOL input file for the functions described in "Print Descriptor Tool Session-Tag Functions" on page 5-16. For more information, refer to "Print Descriptor Tool Command Output Files" on page 5-1.

The PRD DUMP command will list the contents of all print descriptors in a group regardless of the case (mixed, upper, or lower) of the names of the print descriptors.

PRD LIST

MVS	VM	OS/400	OS/2
X	X		

Function Get a list of the print descriptors in a print-descriptor group or in all groups available to a user.

Syntax

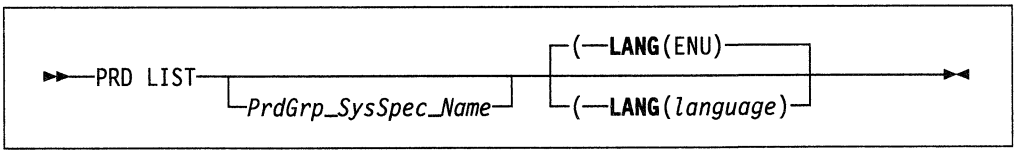


Figure 5-5. Syntax of the PRD LIST Command

Parameters

PrdGrp_SysSpec_Name
Specifies the system-specific name of the print-descriptor group whose print-descriptor names will be listed. If you do not enter a group name, all print descriptors in all groups in the user's group-list print descriptor are listed.

LANG(*language***)**
Specifies the language used to display PrdT messages. Values for this option are standard three-letter abbreviations. For example, ENU (American English), FRS (Swiss French), ESP (Spanish), and so forth. The **LANG** parameter is optional, and the PrdT default value is ENU.

Your location may have installed two or more languages or modified the languages supplied. See your system administrator for specific languages supported at your location and their abbreviations.

Usage: Use the PRD LIST command to create an output file containing a list of the names of the print descriptors in a group or in all groups in a user's group-list print descriptor. For more information, refer to "Print Descriptor Tool Command Output Files" on page 5-1.

The PRD LIST command will list the names of all print descriptors in the specified group regardless of the case (mixed, upper, or lower) of the names of the print descriptors.

PRD QRY

MVS	VM	OS/400	OS/2
X	X		

Function List the contents of a print descriptor.

Syntax

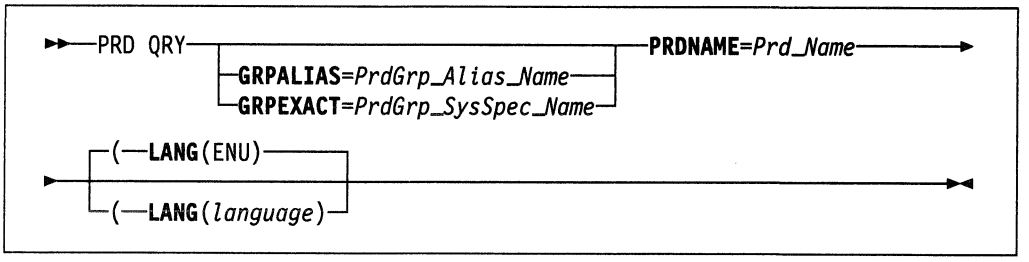


Figure 5-6. Syntax of the PRD QRY Command

Parameters

GRPALIAS
The alias name of the print-descriptor group.

GRPEXACT
The exact name of the print-descriptor group.

PRDNAME
The print-descriptor name.

LANG(*language*)
Specifies the language used to display PrdT messages. Values for this option are standard three-letter abbreviations. For example, ENU (American English), FRS (Swiss French), ESP (Spanish), and so forth. The **LANG** parameter is optional, and the PrdT default value is ENU.

Your location may have installed two or more languages or modified the languages supplied. See your system administrator for specific languages supported at your location and their abbreviations.

Usage: Use the PRD QRY command to create an output file which lists the contents of the queried print descriptor in PrdT-tag format. Since the output file is in PrdT-tag format, if it is renamed it can be used as input to the PRD TOOL command to update the print descriptor. For more information, refer to "Print Descriptor Tool Command Output Files" on page 5-1.

PRD TOOL

MVS	VM	OS/400	OS/2
X	X		

Function Process a PRD TOOL input file.

Syntax

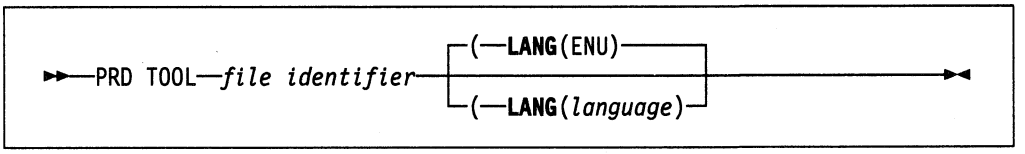


Figure 5-7. Syntax of the PRD TOOL Command

Parameters

file identifier
Specifies the file identifier of the PRD TOOL input file. In MVS, specify an MVS data set. In VM, specify a VM file.

LANG(*language*)
Specifies the language used to display PrdT messages. Values for this option are standard three-letter abbreviations. For example, ENU (American English), FRS (Swiss French), ESP (Spanish), and so forth. The **LANG** parameter is optional, and the PrdT default value is ENU.

Your location may have installed two or more languages or modified the languages supplied. See your system administrator for specific languages supported at your location and their abbreviations.

Usage: Use the PRD TOOL to process a PRD TOOL input file. The input file cannot have the same name as the output files produced by PrdT commands. For more information, see "PRD TOOL Input Files" on page 5-14, and "Print Descriptor Tool Command Output Files" on page 5-1.

Print Descriptor Tool Tag Reference

The following sections:

- Describe PRD TOOL input files
- Show the format used to describe the PrdT tags
- Provide reference information on the PrdT tag functions.

PRD TOOL Input Files

The following sections tell how to:

- Create PRD TOOL input files
- Find errors in a PRD TOOL input file.

Creating PRD TOOL Input Files

You use the PRD TOOL command to process PRD TOOL input files, which contain PrdT tags. For more information on the PRD TOOL command, refer to "Using the Print Descriptor Tool Commands" on page 5-1 and "PRD TOOL" on page 5-13.

You can create PRD TOOL input files with any editor that does not imbed formatting characters. PRD TOOL input files can have any valid system name *except* the name of the output file from the PRD LIST, PRD QRY, and PRD DUMP commands or the List Print Descriptors and List the Contents of a Print Descriptor tag functions. You may, however, want to rename these output files and use them as PRD TOOL input files. Refer to "Print Descriptor Tool Command Output Files" on page 5-1 for more information.

PRD TOOL input files must have the file attributes shown in Table 5-2.

Table 5-2. PRD TOOL Input File Attributes

Operating System	Record Format	Maximum Record Length
VM	variable (recfm = V)	variable
MVS	variable (recfm = VB)	255

PRD TOOL input files consist of PrdT tags and optional comments. Each PrdT tag begins with a colon (:) character and ends with a period (.). If you do not use a colon to begin a tag and a period to end the tag, you will get unpredictable results. The backslash (\) character can be used to enter literal characters in a PRD TOOL input file. For example, if you want to enter a colon (:) which is *not* the beginning character of a PrdT tag, enter the \: characters.

You may begin a tag anywhere on a line. For example, tags may be indented for ease of coding and reading PRD TOOL input files as shown in Figure 5-8 on page 5-15.

```
:PRD.
        :GETPRD.PrdName
        :DELOPTION.OptionName
        :SAVE.PrdName
                :DESCRIPTION.Description
                :CONTROLPRD.ControlPrd
                :CONTROLGRP.ControlGrp
        :ESAVE.
:EPRD.
```

Figure 5-8. PRD TOOL Input File Syntax Example

The PrdT tags described in "Print Descriptor Tool Tag Reference" on page 5-14 are shown in indented format. As Figure 5-8 also shows, if a tag has an associated tag value (for example, :GETPRD.*PrdName*), there is no space between the tag and the value. You can also code more than one tag per line in a file, although it is not recommended.

Comments can be placed anywhere inside the tag file (*except* where tag input is expected) and can span lines. The /* characters must precede the comment, and the */ characters must end the comment.

Finding Errors in PRD TOOL Input Files

Even though there may be multiple errors in a PRD TOOL input file, only one error will be reported at a time. Errors are reported by error messages, which provide the line number in the file where the error occurred. After an error is corrected, the next error (if any) will be reported until all errors are corrected. Refer to "Print Descriptor Tool Messages" on page E-1 for more information on these messages.

Format of the Tag Function Descriptions

Each PrdT tag function description contains these sections:

Function name The name of the tag function. A PrdT tag function is a group of associated tags used to do a single function--for example, delete a print descriptor from a group.

Supported system checklist The operating systems supported by the function.

Function A brief description of the tag function.

Syntax Syntax of each tag included in the function.

Tag Descriptions Descriptions of each tag included in the function.

Usage Programming use information for the function.

Where appropriate, the coding rules (required tags and valid tag sequences) are represented as in the example shown in Figure 5-9 on page 5-16.

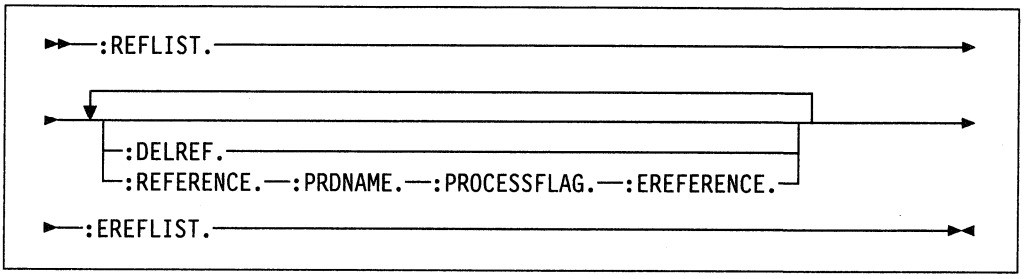


Figure 5-9. Example: Coding Rules for the Print-Descriptor Reference Tags

As shown in Figure 5-9, the PrdT tag coding rules are represented as follows:

- Arrows are used to indicate the valid tag sequences, including whether an item (an item is a tag or group of tags) or a choice of items can be repeated.
- Required items are shown on the center line, while optional items are shown below the center line.
- Choices between items are shown on parallel lines.

As this example shows:

- The REFLIST tag is required and must be coded first.
- Optionally, *either* the DELREF tag or the REFERENCE-EREFERENCE tag group can be chosen next. If chosen, however, all tags in the REFERENCE-EREFERENCE group are required. As the arrows show, the choice of these two optional items can be repeated.
- The EREFLIST tag is required and must be coded last.

Print Descriptor Tool Session-Tag Functions

This section provides reference information on the PrdT session-tag functions. The session-tag functions are those functions that can be entered between the PRD and EPRD tags in a PRD TOOL input file.

Open and Close a PRD TOOL Session

MVS	VM	OS/400	OS/2
X	X		

Function Open and close a PRD TOOL session.

Syntax

```
:PRD.

        /*
        The PRD and EPRD tags open and close the PRD TOOL session.
        As shown, the other session tags can be entered between
        the PRD and EPRD tags.
                                                */

        :GETPRD.PrdName
        :SETOPT.OptionName
                :RULE.Rule
                :VVALUES.Vv
                :DEFAULT.Def
        :ESETOPT.
        :DELOPTION.OptionName
        :REFLIST.
                :REFERENCE.SeqNum
                        :PRDNAME.PrdName
                        :PROCESSFLAG.ProcessFlag
                :EREFERENCE.
                :DELREF.SeqNum
        :EREFLIST.
        :PRDBUILD.
        :GRPLIST.
                :GROUP.SeqNum
                        :FILEID.PrdGrp_SysSpec_Name
                        :PROCESSFLAG.ProcessFlag
                        :ALIAS.PrdGrp_Alias_Name
                        :DESCRIPTION.Description
                :EGROUP.
                :DELGRP.SeqNum
        :EGRPLIST.
        :SAVE.PrdName
                :DESCRIPTION.Description
                :PRDID.PrdId
                :CONTROLPRD.ControlPrd
                :CONTROLGRP.ControlGrp
        :ESAVE.
:EPRD.
```

Figure 5-10. Tag Syntax to Open and Close a PRD TOOL Session

Tag Descriptions

:PRD.
Open a PRD TOOL session.

:EPRD.
Close a PRD TOOL session.

Usage: Use the PRD and EPRD tags to open and close a PRD TOOL session. As shown in Figure 5-10 on page 5-17, you can enter any other PRD TOOL session tags within the session (between the PRD and EPRD tags). Refer to the rest of "Print Descriptor Tool Session-Tag Functions" on page 5-16 for more information on the other session tags and their functions.

Get a Print Descriptor

MVS	VM	OS/400	OS/2
X	X		

Function Get an existing print descriptor.

Syntax

```
:GETPRD.PrdName
```

Figure 5-11. Tag Syntax to Get a Print Descriptor

Tag Descriptions

:GETPRD.*PrdName*

PrdName specifies the name of an existing print descriptor (any valid name format; refer to "Print-Descriptor Name Formats" on page 3-4).

Usage: Use the GETPRD tag to get an existing print descriptor for editing in a PRD TOOL session.

Set or Change a Print Option

MVS	VM	OS/400	OS/2
X	X		

Function Set or change a print option.

Syntax

```
:SETOPT.OptionName
          :RULE.Rule
          :VVALUES.Vv
          :DEFAULT.Def
:ESETOPT.
```

Figure 5-12. Tag Syntax to Set or Change a Print Option

Tag Descriptions

:SETOPT.*OptionName*
OptionName specifies the name of the print option you want to set or change.

:RULE.*Rule*
Rule specifies the print-option validation rule. The PrintManager supplied values are **PRD_NOVALIDATION**, **PRD_STRING**, **PRD_RANGE**, and **PRD_LIST**. For more information, see "Print Option Validation" on page 9-4.

:VVALUES.*Vv*
Vv specifies the valid values for the print option.

:DEFAULT.*Def*
Def specifies the default value for the print option.

:ESETOPT.
End tag for the SETOPT tag.

Usage: Use the tags shown in Figure 5-12 to set or change a print option. The RULE, VVALUES, and DEFAULT tags are optional and can be entered in any order between the SETOPT and ESETOPT tags. However, if you do not specify a rule, default value, or valid values when setting a print option, PrintManager sets the default and valid values to null and the rule to **PRD_NOVALIDATION**.

The print options you set in a print descriptor can be used to validate print options specified in a PrintManager Interface application. For general information on print options (including print-option validation), refer to Chapter 9, Print Options. For information on the syntax, validation rule, and allowable values for specific print options, refer to Appendix A, PrintManager Print Options.

Delete a Print Option

MVS	VM	OS/400	OS/2
X	X		

Function Delete a print option.

Syntax

```
:DELOPTION.OptionName
```

Figure 5-13. Tag Syntax to Delete a Print Option

Tag Descriptions

:DELOPTION.*OptionName*
OptionName specifies the name of the print option you want to delete.

Usage: Use the DELOPTION tag to delete a print option. DELOPTION can be repeated to delete multiple options.

For general information on print options, refer to Chapter 9, Print Options. For information on the syntax, validation rule, and allowable values for specific print options, refer to Appendix A, PrintManager Print Options.

Set or Change a Print-Descriptor Reference

MVS	VM	OS/400	OS/2
X	X		

Function Set or change a print-descriptor reference.

Syntax

```
:REFLIST.
          :REFERENCE.SeqNum
                        :PRDNAME.PrdName
                        :PROCESSFLAG.ProcessFlag
          :EREFERENCE.
:EREFLIST.
```

Figure 5-14. Tag Syntax to Set or Change a Print-Descriptor Reference

Tag Descriptions

:REFLIST.
Begin tag to set or change a print-descriptor reference.

:REFERENCE.*SeqNum*
SeqNum specifies the sequence number of the print-descriptor reference that you want to set or change.

:PRDNAME.*PrdName*
PrdName specifies the name of the referenced print descriptor in any valid name format (refer to "Print-Descriptor Name Formats" on page 3-4). *PrdName*, however, must uniquely identify the desired print descriptor. You may, therefore, want to use the exact-name or relative-name formats for these reference names.

:PROCESSFLAG.*ProcessFlag*
ProcessFlag specifies if the print-descriptor reference will be used to create a composite print descriptor. Set *ProcessFlag* as follows:

PRD_BUILD Use this print-descriptor reference to create a composite print descriptor

PRD_NOBUILD Do not use this reference.

:EREFERENCE.
End tag for the REFERENCE tag.

:EREFLIST.
End tag for the REFLIST tag.

Usage: Use the tags shown in Figure 5-14 to set or change print-descriptor references. When you have a print descriptor that contains references in a PRD TOOL session, you can then use the PRDBUILD tag to build a composite print descriptor.

The sequence number (specified as the value on the REFERENCE tag) determines the print-descriptor reference order and the way a composite print descriptor is created, while the PROCESSFLAG tag specifies if a print-descriptor reference is used when creating a composite print descriptor. Refer to "Build a Composite Print

Descriptor" on page 5-25 for more information on how the PRDBUILD tag creates a composite print descriptor.

Refer to Figure 5-15, which describes the required tags and valid tag sequences for the tags shown in Figure 5-14 on page 5-22. As Figure 5-15 shows, you can set, change, or delete print-descriptor references within the same pair of REFLIST and EREFLIST tags.

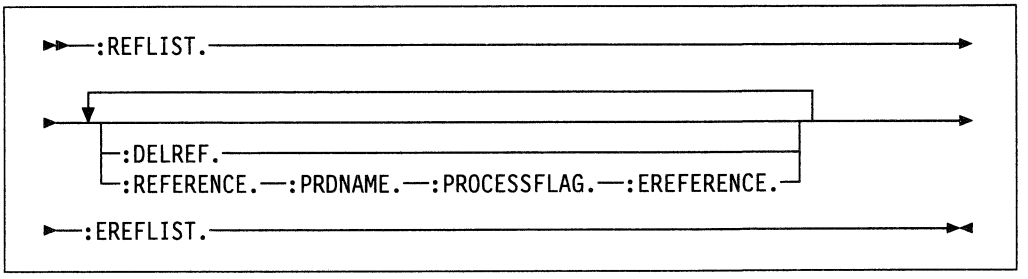


Figure 5-15. Coding Rules for the Set or Change a Print-Descriptor Reference Tags

Delete a Print-Descriptor Reference

MVS	VM	OS/400	OS/2
X	X		

Function Delete a print-descriptor reference.

Syntax

```
:REFLIST.
            :DELREF.SeqNum
:EREFLIST.
```

Figure 5-16. Tag Syntax to Delete a Print-Descriptor Reference

Tag Descriptions

:REFLIST.
Begin tag to delete a print-descriptor reference.

:DELREF.*SeqNum*
SeqNum specifies the sequence number of the print-descriptor reference that you want to delete.

:EREFLIST.
End tag for the REFLIST tag.

Usage: Use the tags shown in Figure 5-16 to delete print-descriptor references. When you have a print descriptor that contains references in a PRD TOOL session, you can then use the PRDBUILD tag to build a composite print descriptor. Refer to "Build a Composite Print Descriptor" on page 5-25 for more information on how the PRDBUILD tag creates a composite print descriptor.

Refer to Figure 5-17, which describes the required tags and valid tag sequences for the tags shown in Figure 5-16. As Figure 5-17 shows, you can set, change, or delete print-descriptor references within the same pair of REFLIST and EREFLIST tags.

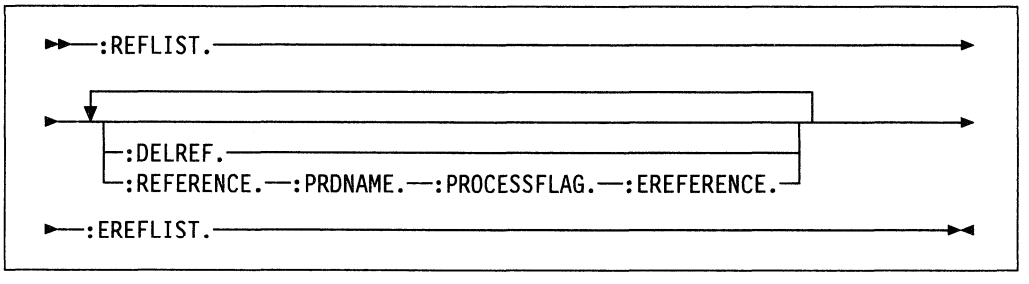


Figure 5-17. Coding Rules for the Delete a Print-Descriptor Reference Tags

Build a Composite Print Descriptor

MVS	VM	OS/400	OS/2
X	X		

Function Build a composite print descriptor.

Syntax

```
:PRDBUILD.
```

Figure 5-18. Tag Syntax to Build a Composite Print Descriptor

Tag Descriptions

:PRDBUILD.
Create a composite print descriptor.

Usage: Use the PRDBUILD tag to create a composite print descriptor from a print descriptor with references in a PRD TOOL session. A composite print descriptor is built by getting each of the referenced print descriptors in sequence order (defined by the value on the REFERENCE tag) and merging them with the current session references (if any exist). The Set or Change a Print-Descriptor Reference function is used to define references in a print descriptor.

Print descriptors are referenced in ascending order of their sequence number (defined by the value on the REFERENCE tag). The *ProcessFlag* values for the PROCESSFLAG tag specify if PRDBUILD will use the print-descriptor reference to create a composite print descriptor. If you specify **PRD_BUILD**, the print descriptor is referenced and its print options are used to create the composite print descriptor. PrintManager then marks these references as **PRD_USED** in the composite print descriptor, and they are not used in later builds for this PRD TOOL session.

Print-option values and validation rules are resolved as described in "Print Option Validation" on page 9-4 and "How Print-Option Information Is Merged" on page 9-7.

Set or Change a Group-List Entry

MVS	VM	OS/400	OS/2
X	X		

Function Set or change a group-list entry.

Syntax

```
:GRPLIST.
        :GROUP.SeqNum
                :FILEID.PrdGrp_SysSpec_Name
                :PROCESSFLAG.ProcessFlag
                :ALIAS.PrdGrp_Alias_Name
                :DESCRIPTION.Description
        :EGROUP.
:EGRPLIST.
```

Figure 5-19. Tag Syntax to Set or Change a Group-List Entry

Tag Descriptions

:GRPLIST.
Begin tag to set or change a group-list entry.

:GROUP.*SeqNum*
SeqNum specifies the sequence number of the group-list entry that you want to set or change.

:FILEID.*PrdGrp_SysSpec_Name*
PrdGrp_SysSpec_Name specifies the system-specific name of the group.

:PROCESSFLAG.*ProcessFlag*
ProcessFlag specifies if the print-descriptor group will be included in the group search order. Set *ProcessFlag* as follows:

PRD_SEARCH	Specifies that this group will be searched when referencing a print descriptor specified in the universal name format (that is, this group will be included in the group search order).
PRD_NOSEARCH	Specifies that this group will not be searched when referencing a print descriptor specified in the universal name format.

Refer to "Searching for a Print Descriptor" on page 3-6 for information on how PrintManager searches for print descriptors.

:ALIAS.*PrdGrp_Alias_Name*
PrdGrp_Alias_Name specifies the alias name for the group.

:DESCRIPTION.*Description*
Description is a description of the print-descriptor group.

:EGROUP.
End tag for the GROUP tag.

:EGRPLIST.
End tag for the GRPLIST tag.

Usage: Use the tags shown in Figure 5-19 to set or change a group-list entry in a group-list print descriptor. Group-list entries specify the print-descriptor groups available to a user, and the search order of these groups.

You can use the GROUP tag to set or change the group search order. Use the PROCESSFLAG tag to specify if the group will be included in the search order. The FILEID tag must uniquely identify the desired group, so you must use the system-specific group name. Use the ALIAS tag to set or change an alias name for the group, and use the DESCRIPTION tag to create a description of the group.

Refer to Figure 5-20, which describes the required tags and valid tag sequences for the tags shown in Figure 5-19 on page 5-26. As Figure 5-20 shows, you can set, change, or delete group-list entries within the same pair of GRPLIST and EGRPLIST tags.

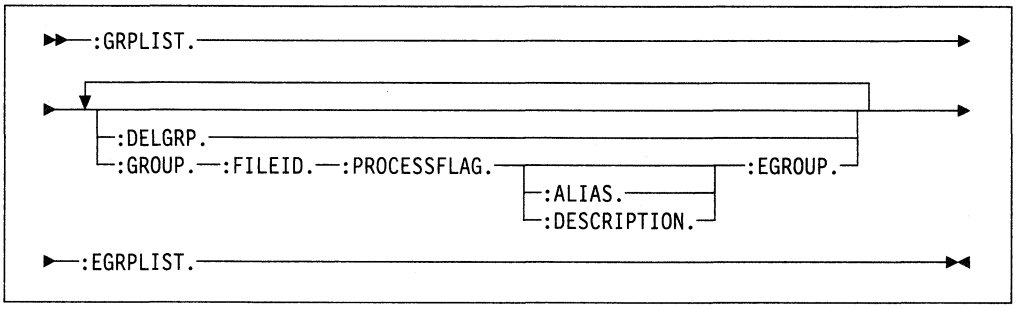


Figure 5-20. Coding Rules for the Set or Change a Group-List Entry Tags

Delete a Group-List Entry

MVS	VM	OS/400	OS/2
X	X		

Function Delete a group-list entry.

Syntax

```
:GRPLIST.
            :DELGRP.SeqNum
:EGRPLIST.
```

Figure 5-21. Tag Syntax to Delete a Group-List Entry

Tag Descriptions

:GRPLIST.
Begin tag to delete a group-list entry.

:DELGRP.*SeqNum*
SeqNum specifies the sequence number of the group-list entry that you want to delete.

:EGRPLIST.
End tag for the GRPLIST tag.

Usage: Use the tags shown in Figure 5-21 to delete a group-list entry in a group-list print descriptor. Group-list entries specify the print-descriptor groups available to a user, and the search order of these groups. You can use the DELGRP tag to delete a group-list entry.

Refer to Figure 5-22, which describes the required tags and valid tag sequences for the tags shown in Figure 5-21. As Figure 5-22 shows, you can set, change, or delete group-list entries within the same pair of GRPLIST and EGRPLIST tags.

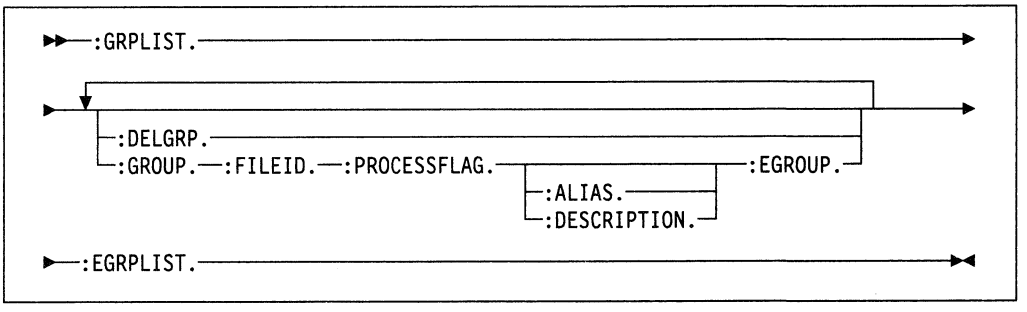


Figure 5-22. Coding Rules for the Delete a Group-List Entry Tags

Store a Print Descriptor

MVS	VM	OS/400	OS/2
X	X		

Function Store a print descriptor.

Syntax

```
:SAVE.PrdName
        :DESCRIPTION.Description
        :PRDID.PrdId
        :CONTROLPRD.ControlPrd
        :CONTROLGRP.ControlGrp
:ESAVE.
```

Figure 5-23. Tag Syntax to Store a Print Descriptor

Tag Descriptions

:SAVE.*PrdName*
PrdName specifies the name of the print descriptor to be stored (any valid name format; refer to "Print-Descriptor Name Formats" on page 3-4).

:DESCRIPTION.*Description*
Description is a description of the print descriptor.

:PRDID.*PrdId*
PrdId specifies the print descriptor type. The default value is **PRD_PRESENTATION_DEVICE**. Refer to "Assigning Print-Descriptor Types" on page 3-11 for more information on print descriptor types and a description of the values supplied with PrintManager.

:CONTROLPRD.*ControlPrd*
ControlPrd specifies the action for the print descriptor as follows:

PRD_CREATE Create a new print descriptor. An error occurs if the specified print descriptor already exists.

PRD_UPDATE Update an existing print descriptor. An error occurs if the specified print descriptor does not exist.

PRD_CREATE_OR_UPDATE
Either create a new print descriptor or update an existing print descriptor.

The default value is **PRD_CREATE**.

:CONTROLGRP.*ControlGrp*
ControlGrp specifies the action for the print-descriptor group as follows:

PRD_AUTO_CREATE Create a new print-descriptor group in which to store the print descriptor. An error occurs if the group already exists.

PRD_NOAUTO_CREATE
Store the print descriptor in an existing group. An error occurs if the group does not exist.

The default value is **PRD_NOAUTO_CREATE**.

:ESAVE.
End tag for the SAVE tag.

Usage: Use the tags shown in Figure 5-23 on page 5-29 to store a print descriptor in a print-descriptor group. You can:

- *Store a new print descriptor in a new print-descriptor group.* Specify **PRD_CREATE** as the value on the CONTROLPRD tag to create the print descriptor (and to ensure that the print descriptor does not already exist). Specify **PRD_AUTO_CREATE** as the value on the CONTROLGRP tag to create the new print-descriptor group (and to ensure that the group does not already exist). Finally, use the Set or Change a Group-List Entry function (refer to page 5-26) to add the new group name to the group-list print descriptor so that the print descriptor is included in the search order.
- *Store a new print descriptor in an existing print-descriptor group.* Specify **PRD_CREATE** as the value on the CONTROLPRD tag to create the print descriptor (and to ensure that the print descriptor does not already exist). Specify **PRD_NOAUTO_CREATE** as the value on the CONTROLGRP tag to store the new print descriptor in an existing print-descriptor group (and to ensure that the group exists and that a new group will not be created).
- *Store an existing print descriptor in an existing print-descriptor group.* If you do not specify the name of the print descriptor, the name in the current PRD TOOL session will be used. Specify **PRD_UPDATE** as the value on the CONTROLPRD tag to update the print descriptor (and to ensure that the print descriptor exists and that a new one will not be created). Specify **PRD_NOAUTO_CREATE** as the value on the CONTROLGRP tag to store the existing print descriptor in an existing print-descriptor group (and to ensure that the group exists and that a new group will not be created).

For more information on how the API stores print descriptors, refer to "Storing a Print Descriptor" on page 3-7.

The DESCRIPTION, PRDID, CONTROLPRD, and CONTROLGRP tags are optional and can be entered in any order between the SAVE and ESAVE tags. tags. If you do not specify a value for the CONTROLPRD and CONTROLGRP tags, the defaults are used.

Print Descriptor Tool Non-Session-Tag Functions

This section provides reference information on the PrdT non-session-tag functions. The non-session-tag functions are those functions that can be entered in a PRD TOOL input file but cannot be entered between the PRD and EPRD tags.

List Print Descriptors

MVS	VM	OS/400	OS/2
X	X		

Get a list of the print descriptors in a print-descriptor group or in all groups available to a user.

Function

Syntax

```
:LISTPRDS.PrdGrp_SysSpec_Name
```

Figure 5-24. Tag Syntax to List Print Descriptors

Tag Descriptions

:LISTPRDS.*PrdGrp_SysSpec_Name*

PrdGrp_SysSpec_Name specifies the system-specific name of the print-descriptor group whose print-descriptor names will be listed. If you do not enter a group name, all print descriptors in all groups in the user's group-list print descriptor are listed.

Usage: Use the LISTPRDS tag to create an output file containing a list of the names of the print descriptors in a group or in all groups in a user's group-list print descriptor.

List the Contents of a Print Descriptor

MVS	VM	OS/400	OS/2
X	X		

Function List the contents of a print descriptor.

Syntax

```
:QUERY.PrdName
```

Figure 5-25. Tag Syntax to List the Contents of a Print Descriptor

Tag Descriptions

:QUERY.*PrdName*

PrdName specifies the print descriptor whose contents will be listed (any valid name format; refer to "Print-Descriptor Name Formats" on page 3-4).

Usage: Use the QUERY tag to create an output file which lists the contents of the queried print descriptor in PrdT-tag format. Since the output file is in PrdT-tag format, if it is renamed it can be used as input to the PRD TOOL command to update the print descriptor.

Delete a Print Descriptor

MVS	VM	OS/400	OS/2
X	X		

Function Delete a print descriptor from a print-descriptor group.

Syntax

```
:DESTROY.PrdName
              :CONTROLGRP.ControlGrp
:EDESTROY.
```

Figure 5-26. Tag Syntax to Delete a Print Descriptor

Tag Descriptions

:DESTROY.*PrdName*
PrdName specifies the name of the print descriptor to be deleted in any valid name format (refer to "Print-Descriptor Name Formats" on page 3-4). You may, however, want to use the exact-name format to ensure that you delete the correct print descriptor.

:CONTROLGRP.*ControlGrp*
Specifies the action when the print descriptor to be deleted is the only print descriptor in the group, as follows:

PRD_AUTO_DELETE If deleting the print descriptor produces an empty print-descriptor group, delete the group.

PRD_NOAUTO_DELETE
Do not delete an empty group.

The default value is **PRD_NOAUTO_DELETE**.

:EDESTROY.
End tag for the DESTROY tag.

Usage: Use the Delete a Print Descriptor function to delete a print descriptor from a print-descriptor group. PrintManager searches for the print descriptor by its name format (refer to "Searching for a Print Descriptor" on page 3-6 for more information). To ensure you delete the print descriptor you want:

- Use the List Print Descriptors tag function or the PRD LIST command to list the print descriptors in a specific group or groups
- If possible, use the exact-name format on the DESTROY tag.

The CONTROLGRP tag is optional. If it is not specified, however, the default will be used.

Translate a Print Descriptor to Interchange Format

MVS	VM	OS/400	OS/2
X	X		

Function Change a print descriptor from system-specific format to interchange format.

Syntax

```
:EXPORT.PrdName
        :FILEID.Prd_File_Name
        :CONTROLPRD.ControlPrd
:EEXPORT.
```

Figure 5-27. Tag Syntax to Translate a Print Descriptor to Interchange Format

Tag Descriptions

:EXPORT.*PrdName*

PrdName specifies the name of the print descriptor in a print-descriptor group that is to be translated to the interchange format (any valid name format; refer to "Print-Descriptor Name Formats" on page 3-4).

:FILEID.*Prd_File_Name*

Prd_File_Name specifies the file name where PrintManager stores the translated print descriptor in interchange format (must be specified in system-specific file name format).

:CONTROLPRD.*ControlPrd*

ControlPrd specifies how PrintManager stores the translated print descriptor in interchange format, as follows:

PRD_CREATE Store the print descriptor in interchange format in a file. An error occurs if the print descriptor is already stored in the named file.

PRD_UPDATE Store the print descriptor in an existing file. An error occurs if the print descriptor is not stored in the named file.

PRD_CREATE_OR_UPDATE
Store the print descriptor either in a new file or in an existing file.

The default value is **PRD_CREATE**.

:EEXPORT.

End tag for the EXPORT tag.

Usage: When print descriptors are exchanged between operating systems, they must be placed in interchange format. The Translate a Print Descriptor to Interchange Format function translates a print descriptor from system-specific format to interchange format and stores the print descriptor in a file. The Translate a Print Descriptor to System-Specific Format function changes a print descriptor from interchange format to system-specific format and stores the print descriptor in the specified print-descriptor group.

When using the Translate a Print Descriptor to Interchange Format function, consider the following guidelines:

- To store a print descriptor in interchange format in a new file, specify **PRD_CREATE** as the value on the CONTROLPRD tag to store the print descriptor (and to ensure that the file does not already exist).
- To store a print descriptor in interchange format in an existing file, specify **PRD_UPDATE** as the value on the CONTROLPRD tag to update the print descriptor (and to ensure that the file exists and that a new file will not be created).
- The CONTROLPRD tag is optional. If it is not specified, however, the default will be used.
- The Translate a Print Descriptor to Interchange Format function does not change the contents of a print descriptor, merely the way it is formatted and stored.

Translate a Print Descriptor to System-Specific Format

MVS	VM	OS/400	OS/2
X	X		

Function Change a print descriptor from interchange format to system-specific format.

Syntax

```
:IMPORT.Prd_File_Name
        :PRDNAME.PrdName
        :CONTROLPRD.ControlPrd
        :CONTROLGRP.ControlGrp
:EIMPORT.
```

Figure 5-28. Tag Syntax to Translate a Print Descriptor to System-Specific Format

Tag Descriptions

:IMPORT.*Prd_File_Name*
Prd_File_Name specifies the system-specific file name of the print descriptor in interchange format that is to be translated.

:PRDNAME.*PrdName*
PrdName specifies the name of the print descriptor to be placed in a print-descriptor group after it is translated to system-specific format (any valid name format; refer to "Print-Descriptor Name Formats" on page 3-4).

:CONTROLPRD.*ControlPrd*
ControlPrd specifies the action for the print descriptor as follows:

PRD_CREATE Create a new print descriptor. An error occurs if the specified print descriptor already exists.

PRD_UPDATE Update an existing print descriptor. An error occurs if the specified print descriptor does not exist.

PRD_CREATE_OR_UPDATE
Either create a new print descriptor or update an existing print descriptor.

The default value is **PRD_CREATE**.

:CONTROLGRP.*ControlGrp*
ControlGrp specifies the action for the print-descriptor group as follows:

PRD_AUTO_CREATE Create a new print-descriptor group in which to store the print descriptor. An error occurs if the group already exists.

PRD_NOAUTO_CREATE
Store the print descriptor in an existing group. An error occurs if the group does not exist.

The default value is **PRD_NOAUTO_CREATE**.

:EIMPORT.
End tag for the IMPORT tag.

Usage: When print descriptors are exchanged between operating systems, they must be placed in interchange format. The Translate a Print Descriptor to Interchange Format function translates a print descriptor from system-specific format to interchange format and stores the print descriptor in a file. The Translate a Print Descriptor to System-Specific Format function changes a print descriptor from interchange format to system-specific format and stores the print descriptor in the specified print-descriptor group.

When using the Translate a Print Descriptor to System-Specific Format function, consider the following guidelines:

- *To store a translated print descriptor in a new print-descriptor group.* Specify **PRD_CREATE** as the value on the CONTROLPRD tag to create the print descriptor (and to ensure that the print descriptor does not already exist). Specify **PRD_AUTO_CREATE** as the value on the CONTROLGRP tag to create the new print-descriptor group. Finally, use the Set or Change a Group-List Entry function (refer to page 5-26) to add the new group name to the group-list print descriptor so that the print descriptor is included in the search order.
- *To store a translated print descriptor in an existing print-descriptor group.* Specify **PRD_CREATE** as the value on the CONTROLPRD tag to create the print descriptor (and to ensure that the print descriptor does not already exist). Specify **PRD_NOAUTO_CREATE** as the value on the CONTROLGRP tag to store the translated print descriptor in an existing print-descriptor group (and to ensure that the group exists and that a new group will not be created).
- *To update a print descriptor in an existing print-descriptor group.* Specify **PRD_UPDATE** as the value on the CONTROLPRD tag to update the print descriptor (and to ensure that the print descriptor exists and that a new one will not be created). Specify **PRD_NOAUTO_CREATE** as the value on the CONTROLGRP tag to store the print descriptor in an existing print-descriptor group (and to ensure that the group exists and that a new group will not be created).
- The Translate a Print Descriptor to System-Specific Format tags must be coded in the order shown in Figure 5-28 on page 5-37. The CONTROLPRD and CONTROLGRP tags are optional. If they are not specified, however, the defaults will be used.
- The Translate a Print Descriptor to System-Specific Format function does not change the contents of a print descriptor, merely the way it is formatted and stored.

Chapter 6. Using the API Verbs

You can use the PrintManager Application Programming Interface (API) verbs to create and maintain print descriptors for your organization's printing needs. This chapter:

- Tells how to manage print descriptors with the API verbs.
- Tells how to handle errors that occur when using the API verbs.

Note: For a programming example, refer to Appendix D, Example of a C Language Print-Descriptor Edit Session.

Managing Print Descriptors with the API Verbs

Creating a print descriptor is done by writing an application program that invokes the API verbs (functions). You call API verbs in your application to establish a print-descriptor edit session that you use to create and update print descriptors. For example, to create a new print descriptor for a standard memo, you use SPRINIT (Initialize PrintManager) to initialize PrintManager, PDOPEN (Open Session) to open the edit session, PDSOPT (Set Print Option) to define page formatting, media, and output controls, PDSAVD (Save Descriptor) to save the print descriptor, PDCLS (Close Session) to close the edit session, and SPRTERM (Terminate PrintManager) to terminate PrintManager.

You can also use an edit session to update existing print descriptors. If you wanted to change your definition of a standard memo, you use SPRINIT (Initialize PrintManager) to initialize PrintManager, PDOPEN (Open Session) to open the edit session, PDGETD (Get Descriptor) to get your existing print descriptor for memos, PDLOPT (List Print Options) and PDQOPT (Query Print Option) to review the original print option values, and PDSOPT (Set Print Option) to change any values you want. You would then use PDSAVD (Save Descriptor) to save the updated print descriptor, PDCLS (Close Session) to close the session, and SPRTERM (Terminate PrintManager) to terminate PrintManager.

You can also use the API verbs to maintain print descriptors (for example, PDLDEF (List Descriptors), PDRDEF (Delete Descriptor), PDIMPD (Import Descriptor), and PDEXPD (Export Descriptor)). Table 6-1 on page 6-2 groups the API verbs by related function and shows which verbs can be used outside an edit session.

Table 6-1 (Page 1 of 2). PrintManager API Verb Function Groups

API Verb	Function Group	Function Description	Outside Edit Session
SPRINIT (Initialize PrintManager)	Initialize PrintManager.	Initialize PrintManager.	YES
PDOPEN (Open Session)	Open a print-descriptor edit session.	Open a print-descriptor edit session.	YES
PDGETD (Get Descriptor)	Get a print descriptor.	Get an existing print descriptor or reinitialize the edit session.	NO
PDLOPT (List Print Options)	Manage print options.	List current print options for an edit session.	NO
PDQOPT (Query Print Option)	Manage print options.	List current values for a print option.	NO
PDSOPT (Set Print Option)	Manage print options.	Set or change a print option.	NO
PDREMO (Delete Print Option)	Manage print options.	Delete a print option.	NO
PDLREF (Query Reference List)	Manage print-descriptor references.	Get current print-descriptor reference information.	NO
PDSREF (Set Reference List)	Manage print-descriptor references.	Set, update, or delete print-descriptor references.	NO
PDBLDD (Build Descriptor)	Create composite print descriptors.	Create a composite print descriptor.	NO
PDLGRP (Query Group List)	Manage print-descriptor group lists.	Get current print-descriptor group list information.	NO
PDSGRP (Set Group List)	Manage print-descriptor group lists.	Set, update, or delete entries from a GLPrd.	NO
PDSAVD (Save Descriptor)	Store a print descriptor.	Store a print descriptor in a print-descriptor group.	NO
PDCLS (Close Session)	Close a print-descriptor edit session.	Close a print-descriptor edit session.	NO
PDLDEF (List Descriptors)	Manage print descriptors.	Get a list of print descriptors.	YES
PDIMPD (Import Descriptor)	Exchange a print descriptor between systems.	Change print descriptor from interchange format to system-specific format.	YES

Table 6-1 (Page 2 of 2). PrintManager API Verb Function Groups

API Verb	Function Group	Function Description	Outside Edit Session
PDEXPD (Export Descriptor)	Exchange a print descriptor between systems.	Change print descriptor from system-specific format to interchange format.	YES
PDRDEF (Delete Descriptor)	Manage print descriptors.	Delete a print descriptor from a print-descriptor group.	YES
SPRGERI (Get Error Information)	Get error data.	Get error data structure from the last error.	YES
SPRGEEM (Get Error Message)	Get error data.	Get additional error information from the last error.	YES
SPRFERI (Free Error Information)	Get error data.	Free storage required for last error data structure.	YES
SPRTERM (Terminate PrintManager)	Terminate PrintManager.	Terminate PrintManager.	YES

Creating a Print Descriptor

The following steps show a typical edit session used to create a print descriptor. Some steps (such as opening the session, storing the print descriptor, and closing the session) are required, whereas others (such as specifying the contents of the print descriptor, setting print-descriptor references, and getting error data) are optional and will vary according to your needs. To create a print descriptor, follow these steps:

1. Use the SPRINIT (Initialize PrintManager) verb to initialize PrintManager.
2. Use the PDOPEN (Open Session) verb to open the edit session.
3. Specify the contents of the print descriptor. Refer to "Contents of Standard Print Descriptors (StdPrds)" on page 3-2 and "Contents of Group-List Print Descriptors (GLPrds)" on page 3-3 for more information.
4. If any errors occur, use the error verbs (SPRGERI (Get Error Information), SPRGEEM (Get Error Message), and SPRFERI (Free Error Information)) to diagnose the problem.
5. Next, use the PDSAVD (Save Descriptor) verb to name and store the print descriptor. The name format determines how you can store and search for the print descriptor (refer to "Print-Descriptor Name Formats" on page 3-4 and "Searching for, Storing, and Deleting Print Descriptors" on page 3-6 for more information).
6. Use the PDCLS (Close Session) verb to close the edit session.
7. Use the SPRTERM (Terminate PrintManager) verb to terminate PrintManager.
8. Run the API application. For more information on C applications, refer to Appendix C, PrintManager API Verb C Language Applications.

Updating a Print Descriptor

Updating a print descriptor includes the basic steps you would use in creating a print descriptor—opening the session, getting a print descriptor, adding or changing contents, doing error diagnosis (if necessary), saving the print descriptor, and closing the session. You will vary the steps slightly in updating a print descriptor, however, particularly in how you might change its contents. Let's look at the basic steps in an edit session to update a print descriptor:

1. Use the SPRINIT (Initialize PrintManager) verb to initialize PrintManager.
2. Use the PDOPEN (Open Session) verb to open the edit session.
3. Use the PDGETD (Get Descriptor) verb to get an existing print descriptor. You need to specify a print-descriptor name for the *PrdName* field. As discussed in "Searching for a Print Descriptor" on page 3-6, PrintManager follows certain rules in getting a print descriptor for you to edit. *Before* trying to get a print descriptor to update, you may want to use the PDLDEF (List Descriptors) verb to list the print descriptors in a particular group to ensure that you retrieve the desired print descriptor.
4. Change the contents of the print descriptor. Refer to "Contents of Standard Print Descriptors (StdPrds)" on page 3-2 and "Contents of Group-List Print Descriptors (GLPrds)" on page 3-3 for more information.
5. If any errors occur, use the error verbs (SPRGERI (Get Error Information), SPRGEEM (Get Error Message), and SPRFERI (Free Error Information)) to diagnose the problem.
6. Next, use the PDSAVD (Save Descriptor) verb to save the contents of the edit session. You may want to store the updated print descriptor under another name or change the group that contains it (refer to "Storing a Print Descriptor" on page 3-7 for more information).
7. Use the PDCLS (Close Session) verb to end the edit session.
8. Use the SPRTERM (Terminate PrintManager) verb to terminate PrintManager.
9. Run the updated API application to put the edit changes into effect. For more information on C applications, refer to Appendix C, PrintManager API Verb C Language Applications.

Handling API Verb Errors

The following sections tell how to handle errors that occur when using the API verbs. Handling these errors consists of:

- Using the PrintManager error verbs to find the cause of errors that occur while using the other API verbs.
- Using the PrintManager trace facility to gather information about problems with PrintManager (if requested by your IBM Support Center representative).
- Recovering from API verb application program abends in MVS and VM.

Using PrintManager Error Verbs

Except for the SPRINIT (Initialize PrintManager) and SPRTERM (Terminate PrintManager) verbs, if an API verb returns a FALSE or 0 return value, you can use the PrintManager error verbs to obtain information about the error as follows:

- Use the SPRGERI (Get Error Information) verb to get the error information structure (**ERRINFO**) associated with the last API error. The error information

structure contains the error id. The error id is a four byte value in which the first two bytes are the severity code of the error and the last two bytes are the error code.

A severity code of 4 indicates a warning condition, while a severity code of 8 indicates an error condition. A warning condition means that the requested function completed successfully, and you may continue, although results may not be as expected. For example, an error code of X'412A' has a severity code of 4, and indicates that truncation occurred when a valid value was set in a print descriptor. An error condition means that an error occurred, and the requested function did not complete successfully.

- For some errors, additional error information can be obtained by using the error information structure (**ERRINFO**) as input to the SPRGEEM (Get Error Message) verb. The SPRGEEM (Get Error Message) verb returns additional error information in the **ERRMSG** structure. The **ERRMSG** structure contains the message identifier and a value which is a string containing additional error information.

  Note: Appendix E, Print Descriptor Tool Messages and API Verb Error Codes lists API verb error codes and provides general explanations of associated API errors, a list of API verbs that may have issued the error, user responses, and a list of additional messages (if any) that are issued.

- After the error information structure (**ERRINFO**) from SPRGERI (Get Error Information) is no longer needed, use the SPRFERI (Free Error Information) verb to free storage obtained for this structure.

Note: Storage is not actually freed until either:

- A successful call to the SPRTERM (Terminate PrintManager) verb is made
- Another error is received.

Using the Trace Facility

After you have used the error verbs, if you suspect a problem with PrintManager, you should contact your IBM Support Center representative, who may ask you to run a trace to gather more information. This section tells how to:

- Create a trace setup file
- Start a trace
- Stop a trace.

Creating a Trace Setup File

You specify trace options in a *trace setup file* that you create (refer to "Starting a Trace" on page 6-6).

Notes:

1. If you are using a file editor that can insert line numbers in a file, ensure that you do *not* create a trace setup file with line numbers. A file with line numbers cannot be used as a trace setup file.
2. Keywords must be in upper case or the SPRINIT (Initialize PrintManager) verb will fail.
3. Values must also be in upper case, or they are ignored.

In the trace setup file, you code trace options in the **KEYWORD = VALUE** format and in the order shown in the following list (default values are underscored):

PRTMGR_TRACE_FLAG
Specifies whether tracing is enabled.

TRUE
Tracing is enabled.

FALSE
Tracing is disabled.

PRTMGR_TRACE_FILE = *filename*
Filename specifies the name of the file where trace output will be sent. This keyword and file name is required to successfully complete a trace. In MVS, the output file name is a sequential data set name; in VM it is a file name; and in OS/400 it is a database file name.

If the output file does not exist it will be created. In MVS, the dataset is created with your high level qualifier. In VM, the file name and file type are required. In OS/400, the file name is required.

Note: Ensure that you have write access to the VM minidisk, MVS high level qualifer of the data set, or OS/400 library for the output file. Otherwise, a SPRINIT (Initialize PrintManager) error will occur.

PRTMGR_WRAP
Specifies whether the trace will wrap.

WRAP
Trace will wrap.

NOWRAP
Trace will not wrap (tracing stops when the trace output file is full).

PRTMGR_FILE_SIZE = *size*
Specifies the maximum size (in bytes) of the trace entries before wrapping occurs. If you specify a value of less than 32767, PrintManager will change the value to 32767, which is the default value for all system environments.

PRTMGR_TRACE_CLASS
Specifies the PrintManager verbs to be traced. This keyword (and a value) is required to successfully complete a trace and there is no default.

PRTMGR
Specifies tracing of all the PrintManager Interface verbs.

PRD
Specifies tracing of all the API verbs.

Starting a Trace

To start a trace, you must create a trace setup file (refer to "Creating a Trace Setup File" on page 6-5). The SPRINIT (Initialize PrintManager) verb will then read the trace file and start the trace. If no setup file exists or if PrintManager cannot reference the file, tracing will not occur. The following sections tell how to name and specify a trace setup file.

Starting MVS and VM Traces: On VM and MVS systems, the trace setup file must be associated with the ddname **EKITRACE**. In MVS, you can use JCL or the TSO ALLOCATE command to associate a data set name with this ddname. In VM, you can use the CMS FILEDEF command.

Starting OS/400 Traces: On OS/400 systems, the trace setup file must be specified via a database file with the name **EKITRACE**. To actually start the trace, however, it is recommended that you override the file name **EKITRACE** with an actual database file name with the OVRDBF command. If the override file does not exist, it will be created with a record length of 200 bytes. If you create the override file, it is recommended that the override file have a record length of 200 bytes for a complete trace. If you do not override the file name, then the **EKITRACE** data base file must exist in a library in your library list.

Stopping a Trace

The following sections describe the system-specific methods you use to stop a trace.

Stopping an MVS Trace: On MVS systems, you can stop a trace by either removing the **EKITRACE** ddname or by using the TSO FREE command.

Stopping a VM Trace: On VM system, you stop a trace with the CMS FILEDEF CLEAR command.

Stopping an OS/400 Trace: On OS/400 systems, you can stop a trace with the DLTOVR command which deletes the override of the file name **EKITRACE**. If you do not override the file name, then delete the **EKITRACE** file or remove it from your library list.

Recovering from API Application Program Abends in MVS and VM

If an API application abends while updating a print descriptor, if necessary, you may be able to use a backup group produced by PrintManager to recover your original print-descriptor group.

Abends can be caused by:

- Insufficient disk storage for a print descriptor
- Cancelling the API application
- An unrecoverable I/O error
- A system abend.

Before you save an updated print descriptor, PrintManager attempts to make a backup copy of the original print-descriptor group, using the naming conventions shown in Table 6-2.

Table 6-2. API Backup Group Naming Conventions

Operating System	**Original Group Name**	**Backup Group Name**
VM	*GroupName* GROUP	*GroupName* $EKITMP$ A
MVS	*userid.GroupName*.GROUP	*userid*.$EKITMP$

If the application abended because of insufficient disk storage, obtain additional disk storage. Insufficient disk storage is indicated by a **Disk** *x(nnn)* **is full** message on VM or a X'B37', X'D37', or X'E37' abend in MVS.

If an abend occurs, do the following:

1. Check for a backup group. If a backup group exists, go to step 2. If one does not exist, retry the application.

- If the application completes successfully, no further action is necessary.
- If the application abends, repeat step 1.

2. If a backup group exists, rename the backup group (you cannot retry the application if a backup group exists), then retry the application.
   - If the application completes successfully, no further action is necessary.
   - If the retry abends again, use the abend code to determine the cause of a system problem.
   - If the retry failed due to a **PMERR_PRD_INV_GROUP** error, copy the backup group to your original group. Retry the application and repeat step 1.

Chapter 7. API Verb Reference

This chapter contains reference information for the PrintManager API verbs and consists of the following sections:

- "Format of the API Verb Descriptions," which shows the format used to describe the API verbs.
- "API Verb Data Types" on page 7-2, which describes the general data types and data structures used by the API verbs. These data types and structures are shown in a *metalanguage* that corresponds to the verb format used in this chapter. For a description of the actual data types, refer to "Data Types for C Language" on page C-5.
- The descriptions of each API verb.

To supplement the information in this chapter, refer to the following:

- For information on using the API and its verbs (including API edit sessions, print-descriptor contents, and print-descriptor names), refer to Chapter 6, Using the API Verbs.
- For a programming example, refer to Appendix D, Example of a C Language Print-Descriptor Edit Session.
- For verb error data, refer to Appendix E, Print Descriptor Tool Messages and API Verb Error Codes.
- For information on PrintManager print options, refer to Chapter 9, Print Options, and Appendix A, PrintManager Print Options.

Format of the API Verb Descriptions

Each API verb description contains these sections:

Verb name The PrintManager programming verb name and the natural language verb name (in parentheses).

Supported system checklist
The operating systems supported by the verb.

Function A brief description of the verb's function.

Syntax The verb syntax, which is shown in a *metalanguage* using the PrintManager programming verb name. For information on the actual verb calls, refer to Appendix C, PrintManager API Verb C Language Applications.

Parameters The parameters and their general data types. The parameter description also tells whether the parameter is an input, output, or input/output parameter and tells how to use the parameter.

Usage Programming use information for the verb.

Return values The name of the return parameter of the verb and its return values or TRUE/FALSE indicators if a Boolean function.

Environment restrictions
Operating-system-specific information for the verb.

API Verb Data Types

This section describes the general data types for the API verbs described in this chapter.

Data Type	Description
BOOL	An integer with zero or nonzero values
BUFFER	Data buffer
BYTE	A byte of data
CHAR	A single-byte character
COUNT2	A 2-byte count
COUNT2B	A 2-byte count of bytes
ERRINFO	The data structure used with SPRGERI (Get Error Information) to return error information with the following fields:

Field (data type)	Description
fixederrinfo (COUNT2B)	Length of fixed data in this structure
error (ERRORID)	A four byte value in which the first two bytes are the severity and the last two bytes are the error code.
detaillevel (COUNT2)	Reserved
messages (OFFSET2B)	Reserved
binarydata (OFFSET2B)	Reserved

ERRMSG	The data structure used with SPRGEEM (Get Error Message) to return additional error information with the following fields:

Field (data type)	Description
identifer (ULONG)	Error identifier
value (STRL)	Descriptive error information (up to 256 characters)

ERRORID	A four byte value in which the first two bytes are the severity and the last two bytes are the error code.
HAB	Handle to a PrintManager anchor block
HPRD	A print-descriptor edit session identifier
LHANDLE	Pointer for the handle for PrintManager verbs.
OFFSET2B	2-byte offset in bytes
OPTDEFN1	A print option definition structure with the following fields:

Field (data type)	Description
Name (STRL)	Print option name (up to 31 characters)
DefType (ULONG)	Type of data contained in the *Def* field. The following are valid values:
	PRTMGR_BINARY *Def* is a binary value, with its length specified in *DefLength*

PRTMGR_STRL *Def* is a string, with its length implied in the string. *DefLength* is ignored; however, on successful return from the PDQOPT (Query Print Option) verb, this field is updated to the actual default length.

DefLength (ULONG) Length of the data contained in *Def*

Def (BUFFER) Default value for the print option

OPTDEFN2 A print option definition structure with the following fields:

Field (data type)	Description
Name (STRL)	Print option name (up to 31 characters)
DefType (ULONG)	The type of data contained in the *Def* field. The following are valid values:

PRTMGR_BINARY *Def* is a binary value, with its length specified in *DefLength*

PRTMGR_STRL *Def* is a string, with its length implied in the string. *DefLength* is ignored; however, on successful return from the PDQOPT (Query Print Option) verb, this field is updated to the actual default length.

DefLength (ULONG)	Length of the data contained in *Def*
Def (BUFFER)	Default value for the print option
Rule (ULONG)	The validation rule for the print option. For more information on validation rules, refer to "Print Option Validation" on page 9-4.
Vv (STRL)	The valid values specified for the print option. For more information on valid values, refer to "Print Option Defaults, Validation, and Merging" on page 9-4.

PRDDEFN2 A print-descriptor definition structure with the following fields:

Field (data type)	Description
PrdName (STRL)	Print-descriptor name
Description (PRDDESCRIPTION)	Description of the print descriptor
PrdId (ULONG)	The type of print descriptor
LevelStamp (PRDLEVELSTAMP)	The date and time a print descriptor was created or last updated

PRDDESCRIPTION
Description of a print descriptor or print-descriptor group (up to 63 characters).

PRDGRP1 A print-descriptor group structure with the following fields:

Field (data type)	Description
SeqNum (ULONG)	A sequence number that uniquely identifies a print-descriptor group in the group list. This sequence number also implies the print-descriptor group search order.
ProcessFlag (ULONG)	A group search process indicator. Supported values are:

PRD_SEARCH	Specifies that this group will be searched or listed when referencing a print descriptor (that is, this group will be included in the group search order).
PRD_NOSEARCH	Specifies that this group will not be searched or listed when referencing a print descriptor.

GrpAlias (STRL)	A print-descriptor group name in the relative format
GrpName (STRL)	A print-descriptor group name in the system-specific format
Description (PRDDESCRIPTION)	Description of the print-descriptor group

PRDLEVELSTAMP

A time-date stamp (up to 31 characters) in the following format:

Chars	Contents
0 – 3	Year
4	'.'
5 – 6	Month
7	'.'
8 – 9	Day
10	Blank
11 – 12	Hours (HH)
13	':'
14 – 15	Minutes
16	':'
17 – 18	Seconds
19	Blank
20 – 22	Code level

	23 – 24	Random number (unsigned character)
	25 – 31	Reserved

PRDREF1 A print-descriptor reference structure with the following fields:

Field (data type)	Description
SeqNum (ULONG)	A sequence number that uniquely identifies the relative order of the print-descriptor references.
ProcessFlag (ULONG)	A print-descriptor build process indicator. Supported values are:

PRD_BUILD	PDBLDD (Build Descriptor) will use print-descriptor references to build a composite print descriptor from the print options in this print descriptor.
PRD_NOBUILD	Do not use references to build a composite print descriptor
PRD_USED	After a composite print descriptor has been built, the PDBLDD (Build Descriptor) verb marks references as **PRD_USED** in the composite print descriptor, and they are not used in later builds for the current edit session.

RefName (STRL)	A print-descriptor name in any of the valid name formats

SDF A self-defining field (structure) that is used with the PDLOPT (List Print Options) verb (refer to "PDLOPT (List Print Options) PrdListOptions" on page 7-21) and the PDLDEF (List Descriptors) verb (refer to "PDLDEF (List Descriptors) PrdListDescriptors" on page 7-16). This structure is a triplet with the following fields:

Field (data type)	Description
Length (ULONG)	The SDF length in bytes.
Type (ULONG)	The SDF type.
Data (BYTE)	Variable data that is exclusively used and maintained by the API

STRL A string with an implicit length count

STRLARR An array of character strings with implied lengths. The length of this array is specified by an accompanying COUNT parameter when passed to a function.

ULONG An unsigned 4-byte integer value (32 bits)

USHORT An unsigned 2-byte integer value (16 bits)

PDBLDD (Build Descriptor)
PrdBuildDescriptor

MVS	VM	OS/400	OS/2
X	X	X	

Function
Create a composite print descriptor.

Syntax

```
PDBLDD (hprd, success)
```

Figure 7-1. Format of the PDBLDD (Build Descriptor) Verb

Parameters
hprd (HPRD)
(Input). Identifies a valid print-descriptor edit session.

Usage
Use the PDBLDD (Build Descriptor) verb to create a composite print descriptor, which is built by getting each of the referenced print descriptors in sequence order (determined by the *SeqNum* value) and merging them with the current edit session. The PDSREF (Set Reference List) verb is used to define references in a print descriptor.

As described in the information on the PDLREF (Query Reference List) and PDSREF (Set Reference List) verbs, print descriptors are referenced in ascending order of their sequence number (defined in the *SeqNum* field). The **PRD_BUILD** and **PRD_NOBUILD** values on the *ProcessFlag* field specify if PDBLDD (Build Descriptor) will use the print-descriptor reference to create a composite print descriptor. If you specify **PRD_BUILD**, the print descriptor is referenced and its print options are used to create the composite print descriptor. The PDBLDD (Build Descriptor) verb then marks these references as **PRD_USED** in the composite print descriptor, and they are not used in later builds for this edit session.

Print-option values and validation rules are resolved as described in "Print Option Validation" on page 9-4 and "How Print-Option Information Is Merged" on page 9-7.

For the *hprd* parameter, ensure that you enter a valid edit-session identifier. Otherwise, in the 370 environment PrintManager will abend with an abend code of X'0245' and in OS/400 a value of 0 will be returned.

Return Values

success **(BOOL)**

A nonzero value means the composite print descriptor was created successfully, whereas a value of 0 means an error has occurred. If a build error occurs, one or more additional error messages can be retrieved with the SPRGEEM (Get Error Message) verb. Each will indicate a possible cause of the error. The PDBLDD (Build Descriptor) verb will build a composite print descriptor, even if errors occur. For example, if a referenced print descriptor cannot be successfully retrieved, the composite will be built from all other referenced print descriptors. In general, PDBLDD (Build Descriptor) will produce a composite print descriptor as complete as possible with no errors.

Environment Restrictions

None.

PDCLS (Close Session)
PrdClose

MVS	VM	OS/400	OS/2
X	X	X	

Function
End a print-descriptor edit session.

Syntax

PDCLS (hprd, *success*)

Figure 7-2. Format of the PDCLS (Close Session) Verb

Parameters
hprd (HPRD)
(Input). Identifies a valid print-descriptor edit session.

Usage
Use the PDCLS (Close Session) verb to end a print-descriptor edit session and to release processor storage. For the *hprd* parameter, ensure that you enter a valid print-descriptor edit-session identifier (as returned by a successful issue of the PDOPEN (Open Session) verb); otherwise, PrintManager deletes all data in the edit session, marks *hprd* as invalid, and releases all storage to the operating system.

For the *hprd* parameter, ensure that you enter a valid edit-session identifier. Otherwise, in the 370 environment PrintManager will abend with an abend code of X'0245' and in OS/400 a value of 0 will be returned.

Return Values
success **(BOOL)**
If the close operation fails, PDCLS (Close Session) returns a value of 0, deletes all data in the edit session, and marks *hprd* as invalid.

If you enter a valid edit session identifier for the *hprd* parameter, PDCLS (Close Session) returns a nonzero value, the edit session ends successfully, and all processor storage is released.

Environment Restrictions
None.

PDEXPD (Export Descriptor)
PrdExportDescriptor

MVS	VM	OS/400	OS/2
X	X	X	

Function
Change a print descriptor from system-specific format to interchange format.

Syntax

```
PDEXPD (hab, PrdName, ExportName, ControlPrd, success)
```

Figure 7-3. Format of the PDEXPD (Export Descriptor) Verb

Parameters
hab (HAB)
(Input). The anchor block handle returned from a successful call to SPRINIT (Initialize PrintManager).

PrdName (STRL)
(Input). Specifies the name of the print descriptor in a print-descriptor group that is to be translated to the interchange format (any valid name format; refer to "Print-Descriptor Name Formats" on page 3-4).

ExportName (STRL)
(Input). Specifies the file name of the translated print descriptor in interchange format (refer to "Print-Descriptor Storage" on page 3-5 and "Exchanging Print Descriptors between Systems" on page 3-10).

ControlPrd (ULONG)
(Input). Specifies how PrintManager stores the translated print descriptor in interchange format, as follows:

PRD_CREATE Store the print descriptor in interchange format in a file. An error occurs if the print descriptor is already stored in the named file.

PRD_UPDATE Store the print descriptor in an existing file. An error occurs if the print descriptor is not stored in the named file.

PRD_CREATE_OR_UPDATE
Store the print descriptor either in a new file or in an existing file.

Usage
When print descriptors are exchanged between operating systems, they must be placed in interchange format. The PDEXPD (Export Descriptor) verb translates a print descriptor from system-specific format to interchange format and stores the print descriptor in a file. The PDIMPD (Import Descriptor) verb changes a print descriptor from interchange format to system-specific format and stores the print descriptor in the specified print-descriptor group.

When using the PDEXPD (Export Descriptor) verb, consider the following guidelines:

- To store a new print descriptor, specify **PRD_CREATE** on the *ControlPrd* parameter to create the print descriptor (and to ensure that the print descriptor does not already exist).
- To store an existing print descriptor, specify **PRD_UPDATE** on the *ControlPrd* parameter to update the print descriptor (and to ensure that the print descriptor exists and that a new print descriptor will not be created).
- PDEXPD (Export Descriptor) does not change the contents of a print descriptor, merely the way it is formatted and stored.
- You can issue PDEXPD (Export Descriptor) outside an edit session.

For the *hab* parameter, ensure that you enter the anchor-block handle returned by the SPRINIT (Initialize PrintManager) verb. Otherwise, in the 370 environment PrintManager will abend with an abend code of X'0245' and in OS/400 a value of 0 will be returned.

Return Values

success **(BOOL)**

A nonzero value means the format was changed and the print descriptor was stored successfully, whereas a value of 0 means an error has occurred.

Environment Restrictions

None.

PDGETD (Get Descriptor)
PrdGetDescriptor

MVS	VM	OS/400	OS/2
X	X	X	

Function
Get an existing print descriptor or reinitialize the edit session.

Syntax

```
PDGETD (hprd, PrdName, Level, Length,
        Buffer, LengthNeeded, Control, success)
```

Figure 7-4. Format of the PDGETD (Get Descriptor) Verb

Parameters
hprd (HPRD)
(Input). Identifies a valid print-descriptor edit session.

PrdName (STRL)
(Input). The name of the print descriptor. To get an existing print descriptor, enter the print-descriptor name in any of the valid name formats (refer to "Print-Descriptor Name Formats" on page 3-4). To reinitialize the edit session with no print options, no print-descriptor references, and no group-list entries, enter a null value.

Level (ULONG)
(Input). Specifies the format of the data in the buffer as follows:

0 Do not return information to the buffer.

1 Return the print-descriptor name in the exact-name format.

2 Return a **PRDDEFN2** structure with the *PrdName* field in the exact-name format.

Length (ULONG)
(Input). Specifies the buffer size.

Buffer (BUFFER)
(Output). Print-descriptor information as specified in the *Level* parameter. Depending on the print descriptor and the type of information you request, PDGETD (Get Descriptor) will update *LengthNeeded* as described below.

LengthNeeded (ULONG)
(Output). Length needed to hold information about the print descriptor. *LengthNeeded* can contain the following values:

- If you specify a null value for *Buffer* and specify *Length* as 0, *LengthNeeded* will be updated to the buffer size needed for the information.
- If you do not specify sufficient buffer space, *LengthNeeded* will be updated to the buffer space needed for the information.
- If *Length* is sufficient to hold all the information, *LengthNeeded* is updated to the size of the data contained in the buffer.

Control (ULONG)
(Input). This parameter is reserved and should have a value of 0.

Usage

Use the PDGETD (Get Descriptor) verb either to initialize an edit session or to get an existing print descriptor for editing.

For the *hprd* parameter, ensure that you enter a valid edit-session identifier. Otherwise, in the 370 environment PrintManager will abend with an abend code of X'0245' and in OS/400 a value of 0 will be returned.

Return Values

success **(BOOL)**
A nonzero value means the print descriptor was found or the edit session was successfully initialized. A value of 0 means an error has occurred.

Environment Restrictions

None.

PDIMPD (Import Descriptor)
PrdImportDescriptor

MVS	VM	OS/400	OS/2
X	X	X	

Function
Change a print descriptor from interchange format to system-specific format.

Syntax

```
PDIMPD (hab, ImportName, PrdName,
        ControlPrd, ControlGrp, success)
```

Figure 7-5. Format of the PDIMPD (Import Descriptor) Verb

Parameters
hab (HAB)
(Input). The anchor block handle returned from a successful call to SPRINIT (Initialize PrintManager).

ImportName (STRL)
(Input). Specifies the file name of the print descriptor in interchange format to be translated (refer to "Print-Descriptor Storage" on page 3-5 and "Exchanging Print Descriptors between Systems" on page 3-10).

PrdName (STRL)
(Input). Specifies the name of the print descriptor to be placed in a print-descriptor group after it is translated to system-specific format (any valid name format; refer to "Print-Descriptor Name Formats" on page 3-4).

ControlPrd (ULONG)
(Input). Specifies how PrintManager saves the print-descriptor information, as follows:

PRD_CREATE Create a new print descriptor. An error occurs if the named print descriptor already exists.

PRD_UPDATE Update an existing print descriptor. An error occurs if the named print descriptor does not exist.

PRD_CREATE_OR_UPDATE
Either create a new print descriptor or update an existing print descriptor.

ControlGrp (ULONG)
(Input). Specifies the print-descriptor group action as follows:

PRD_AUTO_CREATE Create a new print-descriptor group in which to store the print descriptor. An error occurs if the group already exists.

PRD_NOAUTO_CREATE
Store the print descriptor in an existing group. An error occurs if the group does not exist.

Usage

When print descriptors are interchanged between operating systems, they must be placed in interchange format. The PDEXPD (Export Descriptor) verb translates a print descriptor from system-specific format to interchange format and stores the print descriptor in a file. The PDIMPD (Import Descriptor) verb changes a print descriptor from interchange format to system-specific format and stores the print descriptor in the referenced print-descriptor group.

When using the PDIMPD (Import Descriptor) verb, consider the following guidelines:

- *To store a new print descriptor in a new print-descriptor group,* specify **PRD_CREATE** on the *ControlPrd* parameter to create the print descriptor (and to ensure that the print descriptor does not already exist). Also, specify **PRD_AUTO_CREATE** on the *ControlGrp* parameter to create the new print-descriptor group. Finally, use the PDSGRP (Set Group List) verb to add the new group name to the GLPrd so that the print descriptor is included in the search order.
- *To store a new print descriptor in an existing print-descriptor group,* specify **PRD_CREATE** on the *ControlPrd* parameter to create the print descriptor. Specify **PRD_NOAUTO_CREATE** on the *ControlGrp* parameter to store the new print descriptor in an existing print-descriptor group (and to ensure that the group exists).
- *To store an existing print descriptor in an existing print-descriptor group,* specify **PRD_UPDATE** on the *ControlPrd* parameter to update the print descriptor (and to ensure that the print descriptor exists and that a new print descriptor will not be created). Specify **PRD_NOAUTO_CREATE** on the *ControlGrp* parameter to specify an existing print-descriptor group (and to ensure that the print-descriptor group already exists and that a new group will not be created).
- For more information on how the API stores print descriptors, refer to "Storing a Print Descriptor" on page 3-7.
- PDIMPD (Import Descriptor) does not change the contents of a print descriptor, merely the way it is formatted and stored.
- You can issue PDIMPD (Import Descriptor) outside an edit session.

When you receive a print descriptor in interchange format on OS/400, the file that receives the interchange print descriptor:

- Must have the exact same record length as the print descriptor in interchange format.
- Must be a physical file with FILETYPE(*DATA) and LVLCHK(*NO).

For the *hab* parameter, ensure that you enter the anchor-block handle returned by the SPRINIT (Initialize PrintManager) verb. Otherwise, in the 370 environment PrintManager will abend with an abend code of X'0245' and in OS/400 a value of 0 will be returned.

Return Values

success **(BOOL)**

A nonzero value means the format was changed and the print descriptor was stored successfully, whereas a value of 0 means an error has occurred.

Environment Restrictions

None.

PDLDEF (List Descriptors)
PrdListDescriptors

MVS	VM	OS/400	OS/2
X	X	X	

Function

Get a list of print descriptors from a specific print-descriptor group or a list of all print descriptors available to a user. All available print descriptors are those print descriptors in all groups in a user's GLPrd where *ProcessFlag* is set to **PRD_SEARCH**.

Syntax

```
PDLDEF (hab, PrdGroupName, Level, Length, Buffer, LengthNeeded,
        ItemsReturned, ItemsRemaining, Continue, success)
```

Figure 7-6. Format of the PDLDEF (List Descriptors) Verb

Parameters

hab (HAB)
(Input). The anchor block handle returned from a successful call to SPRINIT (Initialize PrintManager).

PrdGroupName **(STRL)**
(Input). Specifies the name of a print-descriptor group. If you want to list all print descriptors in a group, enter the system-specific group name. If you want to list all print descriptors available to you, enter a null value.

Level (ULONG)
(Input). Specifies the format of the data in the buffer as follows:

1 Return an array of print-descriptor names in the exact-name format.

2 Return an array of **PRDDEFN2** structures specifying the print descriptors retrieved. The *PrdName* fields will provide print-descriptor names in the exact-name format.

Length (ULONG)
(Input). Specifies the buffer size (refer to "Usage").

Buffer (BUFFER)
(Output). Returns a list of the print descriptors in the specified group. If you did not specify sufficient buffer space as indicated on the *Length* parameter, *Buffer* contains a partial list, and *LengthNeeded* indicates a value for the additional space needed to complete the list.

LengthNeeded (ULONG)
(Output). If the buffer is large enough to contain all print-descriptor names in the requested list, *LengthNeeded* indicates the actual size of the list returned. If not, *LengthNeeded* indicates a value for the buffer space needed to complete the list.

ItemsReturned (ULONG)
(Output). Indicates the number of print-descriptor names in the buffer.

ItemsRemaining (ULONG)
(Output). For a partial list, indicates the number of print-descriptor names not yet listed. Otherwise, *ItemsRemaining* is 0.

Continue (SDF)
(Input/Output). Used to get a list of options in a series of calls (refer to "Usage"). If you are not using this field for a series of calls, specify it as null.

Usage

Use the PDLDEF (List Descriptors) verb to list the names of the print descriptors in a group (or list all print descriptors available to you). If you do not know the buffer size required for the list, do one of the following:

- If storage is not constrained or if the list is small, issue PDLDEF (List Descriptors) in two calls. First, specify *Length* as 0 and *Buffer* as null. *LengthNeeded* will indicate a value for the buffer space needed for the list.

  For the second call, allocate the buffer space required (as returned in *LengthNeeded* on the first call) and specify the required buffer space in *Length*. The entire list will be returned in the buffer.

- If storage is constrained or if the list is large, issue PDLDEF (List Descriptors) in a series of calls. First, allocate sufficient buffer space (at least enough to contain one print-descriptor name in the list). For the first call specify *Length* as the amount of storage available in the buffer you are using, and allocate storage for the *Continue* parameter equal to the value of the **PRTMGR_GETFIRST_LENGTH** constant. When allocating storage, do *not* use the *sizeof* C function on the SDF structure to determine the size of the *Continue* parameter. In the *Continue* parameter set the *Type* field to **PRTMGR_GET_FIRST** and the *Length* field to **PRTMGR_GETFIRST_LENGTH**.

  Continue the series of calls (and continue the list) by passing back the *Continue* parameter unmodified. When *ItemsRemaining* is 0, you will have received all list entries.

Notes:

1. Any changes to the print-descriptor groups during a series of calls using *Continue* will not be reflected in the partial lists returned.
2. In the **PRDGRP1** structure, if the *ProcessFlag* field is set to **PRD_NOSEARCH**, print descriptors will not be listed for that group when listing all print descriptors available to you (when *PrdGroupName* has a null value).

You can issue PDLDEF (List Descriptors) outside an edit session.

For the *hab* parameter, ensure that you enter the anchor-block handle returned by the SPRINIT (Initialize PrintManager) verb. Otherwise, in the 370 environment PrintManager will abend with an abend code of X'0245' and in OS/400 a value of 0 will be returned.

Return Values

success **(BOOL)**

A nonzero value means the list request was successful, whereas a value of 0 means an error has occurred.

Environment Restrictions

None.

PDLGRP (Query Group List)
PrdQueryGrpList

MVS	VM	OS/400	OS/2
X	X	X	

Function
List the current print-descriptor group lists for an edit session.

Syntax

```
PDLGRP (hprd, Level, Length, Buffer,
        LengthNeeded, ItemsReturned, success)
```

Figure 7-7. Format of the PDLGRP (Query Group List) Verb

Parameters
hprd (HPRD)
(Input). Identifies a valid print-descriptor edit session.

Level (ULONG)
(Input). Specifies the format of the data in the buffer. Use a value of 1 to specify in *Buffer* an array of **PRDGRP1** structures containing print-descriptor group lists.

Length (ULONG)
(Input). Specifies the buffer size (refer to "Usage").

Buffer (BUFFER)
(Input/Output). Returns an array of **PRDGRP1** structures. If you did not specify sufficient buffer space as indicated on the *Length* parameter, *Buffer* contains a partial array, and *LengthNeeded* indicates a value for the additional space needed to complete the array. Print-descriptor groups are listed in ascending order of their sequence number (*SeqNum*).

LengthNeeded (ULONG)
(Output). If the buffer is large enough to contain an entire array, *LengthNeeded* indicates the actual size of the array. If not, *LengthNeeded* indicates a value for the buffer space needed to complete the array.

ItemsReturned (ULONG)
(Output). Contains the number of elements in the array.

Usage
Use the PDLGRP (Query Group List) verb to list the current print-descriptor group lists for an edit session. You can issue PDLGRP (Query Group List) in two calls. First, specify *Length* as 0 and *Buffer* as null. *LengthNeeded* will indicate a value for the buffer space needed for the entire array.

For the second call, allocate the buffer space required (as returned in *LengthNeeded* on the first call) and specify the required buffer space in *Length*. The entire array will be returned in the buffer.

After listing print-descriptor groups, you can enter or change values for fields in the **PRDGRP1** structures. Then use the PDSGRP (Set Group List) verb to set the group list information.

The *SeqNum* value determines the print-descriptor group search order, so you can use the *SeqNum* to set or change the group search order. The *GrpName* fields in the **PRDGRP1** structures identify print-descriptor groups and must uniquely identify the desired group. You must use the system-specific name for these groups. A *group-list print descriptor (GLPrd)* is used to contain print-descriptor group lists.

For the *hprd* parameter, ensure that you enter a valid edit-session identifier. Otherwise, in the 370 environment PrintManager will abend with an abend code of X'0245' and in OS/400 a value of 0 will be returned.

Return Values

success **(BOOL)**

A nonzero value means the list request was successful, whereas a value of 0 means an error has occurred.

Environment Restrictions

None.

PDLOPT (List Print Options)
PrdListOptions

MVS	VM	OS/400	OS/2
X	X	X	

Function
List current print options for an edit session.

Syntax

```
PDLOPT (hprd, Level, Length, Buffer, LengthNeeded, ItemsReturned,
        ItemsRemaining, Continue, success)
```

Figure 7-8. Format of the PDLOPT (List Print Options) Verb

Parameters
hprd (HPRD)
(Input). Identifies a valid print-descriptor edit session.

Level (ULONG)
(Input). Specifies the format of the data in the buffer. A value of 1 specifies that a list of print option names is returned in the buffer as an array of character strings.

Length (ULONG)
(Input). Specifies the buffer size (refer to "Usage").

Buffer (BUFFER)
(Output). Returns a list of print option names. If you did not specify sufficient buffer space as indicated on the *Length* parameter, *Buffer* contains a partial list, and *LengthNeeded* indicates a value for the additional space needed to complete the list.

LengthNeeded (ULONG)
(Output). If the buffer is large enough to contain all option names in the requested list, *LengthNeeded* indicates the actual size of the list returned. If not, *LengthNeeded* indicates a value for the buffer space needed to complete the list.

ItemsReturned (ULONG)
(Output). Indicates the number of print option names returned in the buffer.

ItemsRemaining (ULONG)
(Output). For a partial list, indicates the number of option names not yet listed. Otherwise, *ItemsRemaining* is 0.

Continue (SDF)
(Input/Output). Used to get a list of options in a series of calls (refer to "Usage"). If you are not using this field for a series of calls, specify it as null.

Usage

Use the PDLOPT (List Print Options) verb to list the names of the current print options in an edit session. If you do not know the buffer size required for the list, do one of the following:

- If storage is not constrained or if the list is small, issue PDLOPT (List Print Options) in two calls. First, specify *Length* as 0 and *Buffer* as null. *LengthNeeded* will indicate a value for the buffer space needed for the list.

  For the second call, allocate the buffer space required (as returned in *LengthNeeded* on the first call) and specify the required buffer space in *Length*. The entire list will be returned in the buffer.

- If storage is constrained or if the list is large, issue PDLOPT (List Print Options) in a series of calls. First, allocate sufficient buffer space (at least enough to contain one option name in the list). For the first call specify *Length* as the amount of storage available in the buffer you are using, and allocate storage for the *Continue* parameter equal to the value of the **PRTMGR_GETFIRST_LENGTH** constant. When allocating storage, do *not* use the *sizeof* C function on the SDF structure to determine the size of the *Continue* parameter. In the *Continue* parameter set the *Type* field to **PRTMGR_GET_FIRST** and the *Length* field to **PRTMGR_GETFIRST_LENGTH**.

  Continue the series of calls (and continue the list) by passing back the *Continue* parameter unmodified. When *ItemsRemaining* is 0, you will have received all list entries.

Notes:

1. When *Length* is set to 0 and *Buffer* is set to null, if the *LengthNeeded* value returned by PDLOPT (List Print Options) is 0 and the return code is non-zero, there are no print options defined for the current print-descriptor edit session. In this case, a second call will not return any meaningful information.
2. Any changes to the print options during a series of calls using *Continue* will not be reflected in the partial lists returned.

For the *hprd* parameter, ensure that you enter a valid edit-session identifier. Otherwise, in the 370 environment PrintManager will abend with an abend code of X'0245' and in OS/400 a value of 0 will be returned.

Return Values

success **(BOOL)**

A nonzero value means the list request was successful, whereas a value of 0 means an error has occurred.

Environment Restrictions

None.

PDLREF (Query Reference List)
PrdQueryRefList

MVS	VM	OS/400	OS/2
X	X	X	

Function
List current print-descriptor reference information.

Syntax

```
PDLREF (hprd, Level, Length, Buffer,
        LengthNeeded, ItemsReturned, success)
```

Figure 7-9. Format of the PDLREF (Query Reference List) Verb

Parameters
hprd (HPRD)
(Input). Identifies a valid print-descriptor edit session.

Level (ULONG)
(Input). Specifies the format of the data in the buffer. Use a value of 1 to return in *Buffer* an array of **PRDREF1** structures containing print-descriptor references.

Length (ULONG)
(Input). Specifies the buffer size (refer to "Usage").

Buffer (BUFFER)
(Input/Output). Contains an array of **PRDREF1** structures. If you did not specify sufficient buffer space as indicated on the *Length* parameter, *Buffer* contains a partial array, and *LengthNeeded* indicates a value for the additional space needed. Print-descriptor references are returned in ascending order of their sequence number (*SeqNum*).

LengthNeeded (ULONG)
(Output). If the buffer is large enough to contain an entire array, *LengthNeeded* indicates the actual size of the array. If not, *LengthNeeded* indicates a value for the buffer space needed to complete the array.

ItemsReturned (ULONG)
(Output). Contains the number of elements in the array.

Usage
Use the PDLREF (Query Reference List) verb to list the current print-descriptor references for an edit session. You can issue PDLREF (Query Reference List) in two calls. First, specify *Length* as 0 and *Buffer* as null. *LengthNeeded* will return a value for the buffer size needed for the entire array.

For the second call, allocate the buffer space required (as returned in *LengthNeeded* on the first call) and specify the required buffer space in *Length*. The entire array will be returned in the buffer.

PrintManager allows only one level of print-descriptor reference. That is, a referenced print descriptor cannot reference another print descriptor.

For the *hprd* parameter, ensure that you enter a valid edit-session identifier. Otherwise, in the 370 environment PrintManager will abend with an abend code of X'0245' and in OS/400 a value of 0 will be returned.

Return Values

success **(BOOL)**

A nonzero value means the reference information was returned successfully, whereas a value of 0 means an error has occurred.

Environment Restrictions

None.

PDOPEN (Open Session)
PrdOpen

MVS	VM	OS/400	OS/2
X	X	X	

Function
Open a print-descriptor edit session.

Syntax

```
PDOPEN (hab, HPRD)
```

Figure 7-10. Format of the PDOPEN (Open Session) Verb

Parameters
hab (HAB)
(Input). The anchor block handle returned from a successful call to SPRINIT (Initialize PrintManager).

Usage
Use the PDOPEN (Open Session) verb to open a print-descriptor edit session and to allocate processor storage. When you issue the PDOPEN (Open Session) verb, the session contains no print options, no print-descriptor references, and no group-list entries.

For the *hab* parameter, ensure that you enter the anchor-block handle returned by the SPRINIT (Initialize PrintManager) verb. Otherwise, in the 370 environment PrintManager will abend with an abend code of X'0245' and in OS/400 a value of 0 will be returned.

Return Values
HPRD (HPRD)
The PDOPEN (Open Session) verb should return session identifier (*HPRD*). A return value of 0, however, indicates insufficient storage to open the edit session. Ensure that you save the *HPRD* value that the PDOPEN (Open Session) verb returns. You will need to specify this value on other API verbs used in a print descriptor edit session.

Environment Restrictions
None.

PDQOPT (Query Print Option)
PrdQueryOption

MVS	VM	OS/400	OS/2
X	X	X	

Function
List current values for a print option.

Syntax

```
PDQOPT (hprd, Level, Length, Buffer, LengthNeeded, success)
```

Figure 7-11. Format of the PDQOPT (Query Print Option) Verb

Parameters
hprd (HPRD)
(Input). Identifies a valid print-descriptor edit session.

Level (ULONG)
(Input). Specifies the format of the data in the buffer. On input, specify the print-option name, which can be up to 31 characters. Any additional characters will be truncated. On output, the *Buffer* fields depend on the level you specify, as follows:

1 Return an **OPTDEFN1** structure with values only for the *Def*, *DefType*, and *DefLength* fields.

2 Return an **OPTDEFN2** structure with values for the *Def*, *DefType*, *DefLength*, *Vv*, and *Rule* fields.

Length (ULONG)
(Input). Specifies the buffer size (refer to "Usage").

Buffer (BUFFER)
(Input/Output). Returns a print-option information structure as specified in the *Level* parameter. If you did not specify sufficient buffer space as indicated on the *Length* parameter, *Buffer* does not contain any information, and *LengthNeeded* indicates a value for the space needed.

Information passed in the **OPTDEFN1** and **OPTDEFN2** structures will be modified when returned.

LengthNeeded (ULONG)
(Output). If the buffer is large enough to contain the print-option information, *LengthNeeded* indicates the actual size of the information returned. If *Length* does not specify sufficient buffer space for the requested information, *LengthNeeded* indicates a value for the buffer space needed for the specified print-option information. If you specify *Length* as 0 and *Buffer* as null, *LengthNeeded* indicates a value for the buffer size needed for the largest print option in the print descriptor according to the value specified on the *Level* parameter.

Usage

Use the PDQOPT (Query Print Option) verb to list the current values for a print option. You can issue PDQOPT (Query Print Option) in two calls. First, specify *Length* as 0 and *Buffer* as null, *LengthNeeded* will indicate a value for the buffer size needed for the largest print option in the print descriptor.

For the second call, allocate the buffer space required (as returned in *LengthNeeded* on the first call) and indicate the amount of buffer space in *Length*. The option information will be returned in the buffer. Because *LengthNeeded* reflects the size of the buffer needed for the largest print option in the print descriptor, this buffer is sufficient for any of the current options.

Note: When *Length* is set to 0 and *Buffer* is set to null, if the *LengthNeeded* value returned by PDQOPT (Query Print Option) is 0 and the return code is non-zero, there are no print options defined for the current print-descriptor edit session. In this case, a second call will not return any meaningful information.

For the *hprd* parameter, ensure that you enter a valid edit-session identifier. Otherwise, in the 370 environment PrintManager will abend with an abend code of X'0245' and in OS/400 a value of 0 will be returned.

Return Values

success **(BOOL)**

A nonzero value means the request was successful, whereas a value of 0 means an error has occurred.

Environment Restrictions

None.

PDRDEF (Delete Descriptor)
PrdDestroyDescriptor

MVS	VM	OS/400	OS/2
X	X	X	

Function
Delete a print descriptor from a print-descriptor group.

Syntax

```
PDRDEF (hab, PrdName, ControlGrp, success)
```

Figure 7-12. Format of the PDRDEF (Delete Descriptor) Verb

Parameters
hab (HAB)
(Input). The anchor block handle returned from a successful call to SPRINIT (Initialize PrintManager).

PrdName (STRL)
(Input). Specifies the name of the print descriptor to be deleted (any valid name format; refer to "Print-Descriptor Name Formats" on page 3-4).

ControlGrp (ULONG)
(Input). Specifies the action when the print descriptor to be deleted is the only print descriptor in the group, as follows:

PRD_AUTO_DELETE If deleting the print descriptor produces an empty print-descriptor group, delete the group.

PRD_NOAUTO_DELETE
Do not delete an empty group.

Usage
Use the PDRDEF (Delete Descriptor) verb to delete a print descriptor from a print-descriptor group. PrintManager searches for the print descriptor by its name format (refer to "Searching for a Print Descriptor" on page 3-6 for more information). To ensure you delete the print descriptor you want:

- Use the PDLDEF (List Descriptors) verb to list the print descriptors in a particular group or groups
- If possible, use the exact-name format on the PDRDEF (Delete Descriptor) verb.

You can issue the PDRDEF (Delete Descriptor) verb outside an edit session.

For the *hab* parameter, ensure that you enter the anchor-block handle returned by the SPRINIT (Initialize PrintManager) verb. Otherwise, in the 370 environment PrintManager will abend with an abend code of X'0245' and in OS/400 a value of 0 will be returned.

Return Values

success **(BOOL)**

A nonzero value means the print descriptor was deleted from the group, whereas a value of 0 means an error has occurred.

Environment Restrictions

None.

PDREMO (Delete Print Option)
PrdDeleteOption

MVS	VM	OS/400	OS/2
X	X	X	

Function
Delete a print option, its default value, rule, and valid values from the current edit session.

Syntax

```
PDREMO (hprd, OptionName, success)
```

Figure 7-13. Format of the PDREMO (Delete Print Option) Verb

Parameters
hprd (HPRD)
(Input). Identifies a valid print-descriptor edit session.

OptionName (STRL)
(Input). Identifies the print option to delete. This option-name string can be up to 31 characters. Any additional characters are truncated.

Usage
Use the PDREMO (Delete Print Option) verb to delete a print option from the current edit session. You should first use the PDLOPT (List Print Options) verb to list all current print options before deleting options you do not want.

For the *hprd* parameter, ensure that you enter a valid edit-session identifier. Otherwise, in the 370 environment PrintManager will abend with an abend code of X'0245' and in OS/400 a value of 0 will be returned.

Return Values
success **(BOOL)**
A nonzero value means the print option was deleted successfully, whereas a value of 0 means an error has occurred.

Environment Restrictions
None.

PDSAVD (Save Descriptor)
PrdSaveDescriptor

MVS	VM	OS/400	OS/2
X	X	X	

Function
Save the current edit session as a print descriptor.

Syntax

```
PDSAVD (hprd, Level, Length, Buffer,
        ControlPrd, ControlGrp, success)
```

Figure 7-14. Format of the PDSAVD (Save Descriptor) Verb

Parameters
hprd (HPRD)
(Input). Identifies a valid print-descriptor edit session.

Level (ULONG)
(Input). Specifies the format of the data in the buffer as follows:

0 Ignore any data in *Buffer*. Save the current edit session and store the print descriptor by its current name and group.

1 Store the print descriptor by the name you specify in *Buffer*. The name can be in any valid format (refer to "Print-Descriptor Name Formats" on page 3-4).

2 Store the print descriptor by the following **PRDDEFN2** fields you specify in *Buffer*:

- *PrdName*: Print-descriptor name in any valid name format (refer to "Print-Descriptor Name Formats" on page 3-4).
- *Description*: Print descriptor description.
- *PrdId*: Print-Descriptor type. Refer to "Assigning Print-Descriptor Types" on page 3-11 for more information on this field and a description of the values supplied with PrintManager.

PrintManager sets or updates the *LevelStamp* field.

Length (ULONG)
(Input). Specifies the buffer size.

Buffer (BUFFER)
(Input). Print-descriptor information as specified in the *Level* parameter.

ControlPrd (ULONG)
(Input). Specifies how PrintManager saves the print-descriptor information, as follows:

PRD_CREATE Create a new print descriptor. An error occurs if the named print descriptor already exists.

PRD_UPDATE Update an existing print descriptor. An error occurs if the named print descriptor does not exist.

PRD_CREATE_OR_UPDATE
Either create a new print descriptor or update an existing print descriptor.

ControlGrp (ULONG)
(Input). Specifies the print-descriptor group action as follows:

PRD_AUTO_CREATE Create a new print-descriptor group in which to store the print descriptor. An error occurs if the group already exists.

PRD_NOAUTO_CREATE
Store the print descriptor in an existing group. An error occurs if the group does not exist.

Usage

Use the PDSAVD (Save Descriptor) verb to store a print descriptor in a print-descriptor group. You can:

- *Store a new print descriptor in a new print-descriptor group.* In this case, specify either 1 or 2 on the *Level* parameter. You can enter more information about a new print descriptor by specifying Level 2. Specify **PRD_CREATE** on the *ControlPrd* parameter to create the print descriptor (and to ensure that the print descriptor does not already exist). Also, specify **PRD_AUTO_CREATE** on the *ControlGrp* parameter to create the new print-descriptor group. Finally, use the PDSGRP (Set Group List) verb to add the new group name to the GLPrd so that the print descriptor is included in the search order.
- *Store a new print descriptor in an existing print-descriptor group.* Specify either 1 or 2 on the *Level* parameter, and specify **PRD_CREATE** on the *ControlPrd* parameter to create the print descriptor. Specify **PRD_NOAUTO_CREATE** on the *ControlGrp* parameter to store the new print descriptor in an existing print-descriptor group (and to ensure that the group exists).
- *Store an existing print descriptor in an existing print-descriptor group.* If you do not specify the name of the print descriptor, the name in the current edit session will be used. Specify **PRD_UPDATE** on the *ControlPrd* parameter to update the print descriptor (and to ensure that the print descriptor exists and that a new one will not be created). Specify **PRD_NOAUTO_CREATE** on the *ControlGrp* parameter to specify an existing print-descriptor group (and to ensure that the print-descriptor group already exists and that a new group will not be created).
- For more information on how the API stores print descriptors, refer to "Storing a Print Descriptor" on page 3-7.

For the *hprd* parameter, ensure that you enter a valid edit-session identifier. Otherwise, in the 370 environment PrintManager will abend with an abend code of X'0245' and in OS/400 a value of 0 will be returned.

Return Values

success **(BOOL)**

A nonzero value means the print descriptor was stored successfully, whereas a value of 0 means an error has occurred.

Environment Restrictions

None.

PDSGRP (Set Group List)
PrdSetGrpList

MVS	VM	OS/400	OS/2
X	X	X	

Function
Set, update, or delete entries from a print-descriptor group list.

Syntax

```
PDSGRP (hprd, Level, Length, Buffer, TotalItems, success)
```

Figure 7-15. Format of the PDSGRP (Set Group List) Verb

Parameters
hprd (HPRD)
(Input). Identifies a valid print-descriptor edit session.

Level (ULONG)
(Input). Specifies the format of the data in the buffer. Use a value of 1 to specify an array of **PRDGRP1** structures containing print-descriptor group lists.

Length (ULONG)
(Input). Specifies the buffer size.

Buffer (BUFFER)
(Input). Contains an array of **PRDGRP1** structures. Print-descriptor group-list entries are listed in ascending order of their sequence number (*SeqNum*).

TotalItems (ULONG)
(Input). The number of groups contained in the buffer.

Usage
Use the PDSGRP (Set Group List) verb to set, update, or delete print-descriptor group list information in a GLPrd. You may first want to use the PDLGRP (Query Group List) verb to list print-descriptor group information and to set or change values for fields in the **PRDGRP1** structures. Then use the PDSGRP (Set Group List) verb to set the print-descriptor group information. To delete an entry from the group list, specify its group name as null, blank, or 0 length.

The *SeqNum* value determines the print-descriptor group search order, so you can use the *SeqNum* to set or change the group search order. If there is no sequence number, you must specify one in the *SeqNum* field.

The *GrpName* fields in the **PRDGRP1** structures identify print-descriptor groups, and must uniquely identify the desired group. You must use the system-specific names for these groups.

A group referenced by a group-list entry need not actually exist. For example, a group may not have been defined yet, or it may have been deleted. Therefore, you need not update the group list whenever a group is deleted or before it is defined.

Set the *ProcessFlag* field in the **PRDGRP1** structure to specify if the print-descriptor group will be included in the group search order. Set the *ProcessFlag* field as follows:

PRD_SEARCH Specifies that this group will be searched when referencing a print descriptor specified in the universal name format (that is, this group will be included in the group search order).

PRD_NOSEARCH Specifies that this group will not be searched when referencing a print descriptor specified in the universal name format.

Refer to "Searching for a Print Descriptor" on page 3-6 for information on how PrintManager searches for print descriptors.

For the *hprd* parameter, ensure that you enter a valid edit-session identifier. Otherwise, in the 370 environment PrintManager will abend with an abend code of X'0245' and in OS/400 a value of 0 will be returned.

Return Values

success **(BOOL)**

A nonzero value means the group list information was set, updated, or deleted successfully, whereas a value of 0 means an error has occurred.

Environment Restrictions

None.

PDSOPT (Set Print Option)
PrdSetOption

MVS	VM	OS/400	OS/2
X	X	X	

Function
Set or change a print option.

Syntax

PDSOPT (hprd, Level, Length, Buffer, *success*)

Figure 7-16. Format of the PDSOPT (Set Print Option) Verb

Parameters
hprd (HPRD)
(Input). Identifies a valid print-descriptor edit session.

Level (ULONG)
(Input). Specifies the format of the data in the buffer. Specify the print-option name, which can be up to 31 characters. Any additional characters will be truncated. On input, format of the *Buffer* parameter depends on the level you specify, as follows:

1 An **OPTDEFN1** structure with values for the *Def*, *DefType*, and *DefLength* fields.

2 An **OPTDEFN2** structure with values for the *Def*, *DefType*, *DefLength*, *Vv*, and *Rule* fields.

Length (ULONG)
(Input). Specifies the buffer size.

Buffer (BUFFER)
(Input). The print-option information structure as specified in the *Level* parameter.

Usage
The print options you set in a print descriptor can be used to validate print options specified in a PrintManager Interface application. For general information on print options (including print-option validation), refer to Chapter 9, Print Options. For information on the syntax, validation rule, and allowable values for specific print options, refer to Appendix A, PrintManager Print Options.

When using SPRSOPT (Set Option) to change (reset) a print option:

- If you specify 1 on the *Level* parameter when resetting a print option, you can only reset the default value. The existing validation rule and valid values (if any) remain unchanged.
- If you specify 2 on the *Level* parameter when resetting a print option that has no validation rule, valid values, or default:

- If you specify the valid values as null, the validation rule is reset to **PRD_NOVALIDATION** regardless of any new validation rule you specify. In this case, however, you can specify a new default value.
- If you specify a new valid value and validation rule, you can also specify a new default value. If you specify a null, blank, or zero length default value, the default is reset to null.

- If you specify 2 on the *Level* parameter when resetting a print option that has an existing validation rule and either existing or null valid values or default value:
  - If you specify a validation rule of **PRD_STRING**, **PRD_RANGE**, or **PRD_LIST** and a new valid value, if you specify a null for the default, the default value remains unchanged. If you specify a blank or zero length default, the default is reset to null.
  - If you specify a validation rule of **PRD_STRING**, **PRD_RANGE**, or **PRD_LIST** and a null valid value, the existing rule and valid values remain unchanged. If you specify null for the default, the default value remains unchanged. If you specify a blank or zero length default, the default is reset to null.
  - If you specify a rule of **PRD_NOVALIDATION**, the valid values remain unchanged. If you specify null for a default, the default value remains unchanged; If you specify a blank or zero length default, the default value is reset to null.

For the *hprd* parameter, ensure that you enter a valid edit-session identifier. Otherwise, in the 370 environment PrintManager will abend with an abend code of X'0245' and in OS/400 a value of 0 will be returned.

Return Values

success **(BOOL)**

A nonzero value means the print-option values were set successfully, whereas a value of 0 means an error has occurred.

Environment Restrictions

None.

PDSREF (Set Reference List)
PrdSetRefList

MVS	VM	OS/400	OS/2
X	X	X	

Function
Set, update, or delete print-descriptor references.

Syntax

```
PDSREF (hprd, Level, Length, Buffer, TotalItems, success)
```

Figure 7-17. Format of the PDSREF (Set Reference List) Verb

Parameters
hprd (HPRD)
(Input). Identifies a valid print-descriptor edit session.

Level (ULONG)
(Input). Specifies the format of the data in the buffer. Use a value of 1 to specify the format of the data in the buffer (an array of **PRDREF1** structures containing print-descriptor references).

Length (ULONG)
(Input). Specifies the buffer size.

Buffer (BUFFER)
(Input). An array of **PRDREF1** structures.

TotalItems (ULONG)
(Input). The total number of print-descriptor references contained in the buffer.

Usage
Use the PDSREF (Set Reference List) verb to change, set, or delete print-descriptor references for an edit session. You may first want to use the PDLREF (Query Reference List) verb to list print-descriptor references and to enter or change values for fields in the **PRDREF1** structures. Then use the PDSREF (Set Reference List) verb to set the print-descriptor references. To delete a reference, specify its reference name as null, blank, or 0 length.

The *SeqNum* value determines the print-descriptor reference order and the way a composite print descriptor is created with the PDBLDD (Build Descriptor) verb. If there is no sequence number, you must specify one in the *SeqNum* field. You should *not* use a sequence number of 0, because this refers to the current edit session.

The *RefName* field must uniquely identify the desired print descriptor. You may, therefore, want to use the exact-name or relative-name formats for these reference names.

Use the *ProcessFlag* field to specify if the PDBLDD (Build Descriptor) verb will use the print-descriptor reference to create a composite print descriptor. Set the *ProcessFlag* field as follows:

PRD_BUILD The PDBLDD (Build Descriptor) verb will use this print-descriptor reference to create a composite print descriptor

PRD_NOBUILD The PDBLDD (Build Descriptor) verb will not use this reference.

Refer to "PDBLDD (Build Descriptor) PrdBuildDescriptor" on page 7-6 for more information on how the PDBLDD (Build Descriptor) verb uses the *ProcessFlag* values during build processing.

For the *hprd* parameter, ensure that you enter a valid edit-session identifier. Otherwise, in the 370 environment PrintManager will abend with an abend code of X'0245' and in OS/400 a value of 0 will be returned.

Return Values

success **(BOOL)**

A nonzero value means the reference information was set, updated, or deleted successfully, whereas a value of 0 means an error has occurred.

PrintManager allows only one level of print-descriptor reference. That is, a referenced print descriptor cannot reference another print descriptor.

Environment Restrictions

None.

SPRFERI (Free Error Information)
PrtMgrFreeErrorInfo

MVS	VM	OS/400	OS/2
X	X	X	

Function
Frees storage for the error-information structure returned from the SPRGERI (Get Error Information) verb.

Syntax

```
SPRFERI (ErrorInfo, success)
```

Figure 7-18. Format of the SPRFERI (Free Error Information) Verb

Parameters
ErrorInfo (ERRINFO)
(Input). The **ERRINFO** structure.

Usage
Use SPRFERI (Free Error Information) to free storage for the error-information structure you specify in *ErrorInfo*. You should issue SPRFERI (Free Error Information) only once for every error-information structure returned with the SPRGERI (Get Error Information) verb.

Note: Storage is not actually freed until either:

- A successful call to the SPRTERM (Terminate PrintManager) verb is made
- Another error is received.

Return Values
success **(BOOL)**
A nonzero value means storage was freed successfully, whereas a value of 0 means an error has occurred.

Environment Restrictions
None.

SPRGEEM (Get Error Message)
PrtMgrGetExtErrorMsg

MVS	VM	OS/400	OS/2
X	X	X	

Function
Get additional error information associated with the error information structure.

Syntax

```
SPRGEEM (ErrorInfo, Index, ErrorMsg, TotalCount, success)
```

Figure 7-19. Format of the SPRGEEM (Get Error Message) Verb

Parameters
ErrorInfo (ERRINFO)
(Input). The error-information structure (**ERRINFO**) returned by SPRGERI (Get Error Information).

Index (ULONG)
(Input). Specifies which additional message to return. Use a value of 1 to select the first message, a value of 2 to select the second message, and so on.

ErrorMsg (ERRMSG)
(Output). The buffer containing the **ERRMSG** structure.

TotalCount (ULONG)
(Output). SPRGEEM (Get Error Message) updates this field to specify the number of additional messages associated with the error-information structure.

Usage
The SPRGEEM (Get Error Message) verb returns additional error information in the **ERRMSG** structure. The original **ERRINFO** structure returned from SPRGERI (Get Error Information) must be passed to SPRGEEM (Get Error Message). It must not be a copy.

Information in the **ERRMSG** structure is often useful in further resolving the cause of errors. For example, for error code PMERR_PRD_OPEN_ERROR, the *Value* field of the **ERRMSG** structure provides the name of the print-descriptor group where the error occurred. For operating system errors, PrintManager updates the *Value* field with error text provided by the operating system. Information in the *Value* field is truncated to 256 characters, if necessary.

There may be 1 to *n* additional error messages associated with the **ERRINFO** structure. Set *Index* to 0 to get the total number of messages that can be returned. If the value returned in *TotalCount* is 0, there are no additional messages. If there are additional messages, set *Index* to correspond to the message you want to return. If you want to return all additional messages, call SPRGEEM (Get Error Message) repeatedly, incrementing *Index* until *TotalCount* is reached.

Return Values

success **(BOOL)**

A nonzero value means that the requested function completed successfully, whereas a value of 0 means an error has occurred.

None.

SPRGERI (Get Error Information)
PrtMgrGetErrorInfo

MVS	VM	OS/400	OS/2
X	X	X	

Function
Get error-information structure from the last error.

Syntax

```
SPRGERI (hab, ERRINFO)
```

Figure 7-20. Format of the SPRGERI (Get Error Information) Verb

Parameters
hab (HAB)
(Input). The anchor block handle returned from a successful call to SPRINIT (Initialize PrintManager).

Usage
Use the SPRGERI (Get Error Information) verb to get the error-information structure associated with the last API error. The error information structure contains the error id. The error id is a four byte value in which the first two bytes are the severity and the last two bytes are the error code. You can then use the error-information structure (**ERRINFO**) as input to the SPRGEEM (Get Error Message) verb to get additional information (if any) about the error. To free storage for the error-information structure, use SPRFERI (Free Error Information).

Note: If PrintManager cannot be initialized (with SPRINIT (Initialize PrintManager)), SPRGERI (Get Error Information) will not return an error information structure.

For the *hab* parameter, ensure that you enter the anchor-block handle returned by the SPRINIT (Initialize PrintManager) verb. Otherwise, in the 370 environment PrintManager will abend with an abend code of X'0245' and in OS/400 a value of 0 will be returned.

Return Values
ERRINFO (ERRINFO)
The SPRGERI (Get Error Information) verb returns an **ERRINFO** structure for the last error. Refer to Appendix E, Print Descriptor Tool Messages and API Verb Error Codes for information on API verb error codes. A return value of 0 indicates that no previous error occurred.

Environment Restrictions
None.

SPRINIT (Initialize PrintManager)
PrtMgrInitialize

MVS	VM	OS/400	OS/2
X	X	X	

Function
Initialize PrintManager.

Syntax

```
SPRINIT (Options, HAB)
```

Figure 7-21. Format of the SPRINIT (Initialize PrintManager) Verb

Parameters
Options (USHORT)
(Input). This parameter is reserved and should have a value of 0.

Usage
Use the SPRINIT (Initialize PrintManager) verb to initialize PrintManager. You cannot issue any other API verbs (including error verbs) until you have successfully initialized PrintManager.

Return Values
HAB (HAB)
The SPRINIT (Initialize PrintManager) verb should return an anchor block handle (HAB). A return value of 0, however, indicates an error while initializing PrintManager.

Ensure that you save the HAB value that the SPRINIT (Initialize PrintManager) verb returns. You will need to specify this value on other API verbs. You cannot issue two successive calls to SPRINIT (Initialize PrintManager). You must terminate PrintManager with the SPRTERM (Terminate PrintManager) verb before reinitializing PrintManager.

Environment Restrictions
None.

SPRTERM (Terminate PrintManager)
PrtMgrTerminate

MVS	VM	OS/400	OS/2
X	X	X	

Function
Terminate PrintManager.

Syntax

```
SPRTERM (hab, success)
```

Figure 7-22. Format of the SPRTERM (Terminate PrintManager) Verb

Parameters
hab (HAB)
(Input). The anchor block handle returned from a successful call to SPRINIT (Initialize PrintManager).

Usage
Use the SPRTERM (Terminate PrintManager) verb to terminate PrintManager and release all associated resources. For the *hab* parameter, ensure that you enter the anchor block handle returned on the SPRINIT (Initialize PrintManager) verb when you initialized PrintManager. You must terminate a PrintManager session before you can reinitialize PrintManager.

For the *hab* parameter, ensure that you enter the anchor-block handle returned by the SPRINIT (Initialize PrintManager) verb. Otherwise, in the 370 environment PrintManager will abend with an abend code of X'0245' and in OS/400 a value of 0 will be returned.

Return Values
success **(BOOL)**

Environment Restrictions
None.

Chapter 8. Customizing the Print Request Facility

The following sections describe the parts of the Print Request Facility (PRF) component of IBM SAA PrintManager that you can customize to meet the needs of your organization. For more information on using the PRF, refer to *IBM SAA PrintManager User's Guide.*

Note: The PRF is available in MVS and VM only.

PRF Exec

PrintManager uses the PRF exec to invoke the PRF panel or command interfaces. This exec provides instructions (as comments) that you can use to modify PRF functions. For example, you can modify the PRF exec to select the default language other than ENU (American English) or to change the default for the MENU option so that the command interface is automatically invoked.

Main Menu Options and Function Keys

PrintManager also uses an exec to make assignments for Main Menu option names and PRF function keys, and this exec also provides instructions (as comments) that you can use to change these assignments. PrintManager provides the VM and MVS execs shown in Table 8-1.

Table 8-1. VM and MVS Names for Exec for PRF Main Menu Options and PF Keys

VM Exec Name	MVS Exec Name	Natural Language Supported
EKICMENU EXEC	EKI.V1R1M0.EKIEXEC(EKICMENU)	English (mixed case option names)
EKICMEUO EXEC	EKI.V1R1M0.EKIEXEC(EKICMEUO)	English (upper case option names)
EKICMDEU EXEC	EKI.V1R1M0.EKIEXEC(EKICMDEU)	German
EKICMFRA EXEC	EKI.V1R1M0.EKIEXEC(EKICMFRA)	French
EKICMITA EXEC	EKI.V1R1M0.EKIEXEC(EKICMITA)	Italian
EKICMESP EXEC	EKI.V1R1M0.EKIEXEC(EKICMESP)	Spanish
EKICMFRC EXEC	EKI.V1R1M0.EKIEXEC(EKICMFRC)	Canadian French
EKICMFRS EXEC	EKI.V1R1M0.EKIEXEC(EKICMFRS)	Swiss French
EKICMDES EXEC	EKI.V1R1M0.EKIEXEC(EKICMDES)	Swiss German
EKICMITS EXEC	EKI.V1R1M0.EKIEXEC(EKICMITS)	Swiss Italian
EKICMJPN EXEC	EKI.V1R1M0.EKIEXEC(EKICMJPN)	Japanese

Translation Tables

PrintManager uses a translation table to map the PRF options and their values to natural language names and descriptions on the PRF panels. PrintManager provides the VM and MVS translation tables shown in Table 8-2.

Table 8-2. VM and MVS Names for PRF Translation Tables

VM Table Name	MVS Table Name	Natural Language Supported
EKI$LANG OPTNSENU	EKI.V1R1M0.OPTNSENU	English (mixed case natural language option names and descriptions)
EKI$LANG OPTNSEUO	EKI.V1R1M0.OPTNSEUO	English (upper case natural language option names and descriptions)
EKI$LANG OPTNSDEU	EKI.V1R1M0.OPTNSDEU	German
EKI$LANG OPTNSFRA	EKI.V1R1M0.OPTNSFRA	French
EKI$LANG OPTNSITA	EKI.V1R1M0.OPTNSITA	Italian
EKI$LANG OPTNSESP	EKI.V1R1M0.OPTNSESP	Spanish
EKI$LANG OPTNSFRC	EKI.V1R1M0.OPTNSFRC	Canadian French
EKI$LANG OPTNSFRS	EKI.V1R1M0.OPTNSFRS	Swiss French
EKI$LANG OPTNSDES	EKI.V1R1M0.OPTNSDES	Swiss German
EKI$LANG OPTNSITS	EKI.V1R1M0.OPTNSITS	Swiss Italian
EKI$LANG OPTNSJPN	EKI.V1R1M0.OPTNSJPN	Japanese

Users select a translation table by entering the first three letters of the translation table type on the *language* PRF option. For example, to select English:

language (ENU)

The format of each translation table is based on the following tags:

Tag	Description
:tname.	PrintManager print option name (up to 31 bytes).
:trans.	PRF natural language name of the option (up to 32 bytes).
:desc.	PRF natural language description of the option. Either the first six lines (63 characters each) or up to the first blank line is displayed on the option Values panel. All lines are displayed on the help panel for the option.

For example, the PrintManager **CC** option is mapped to the PRF as follows:

```
:tname.CC
:trans.Carriage Control
:desc.Carriage control characters indicate that the first
character of each record is to be interpreted as a carriage control
character.  Carriage control characters in the print file are used to
control line spacing, line skipping, and page-eject operations as
the data is being formatted.

If the print file contains line data with American National Standards
Institute carriage control characters or Advanced Function Printing
data-stream (AFPDS) records preceded by a hex '5A' character,
specify YES.  Text-formatting programs, such as DCF, that
support AFP can generate AFPDS print files if you specify that the
output device is AFP.  If the print file does not contain any
carriage control characters, specify NO.  If the print file contains
machine carriage control characters, specify MACHINE.

If Data Type is set to MODCAP, AFPDS, or AFPDSLINE and
Carriage Control is not set or set to NO, Carriage Control
will be set to YES for VM or MACHINE for MVS when the print
job is submitted.
```

In this example, only the first part of the description (up to the blank line) appears on the Values panel, but all lines appear on the help panel for this option.

If an option does not have an entry in the translation table, the PrintManager print option name will be displayed in upper case on the PRF panel. You can use a text editor to modify the :trans. and :desc. entries in a translation table. The colon symbol (:), if used in the :desc. text, must be followed by a blank or it will be misinterpreted as a tag.

Using Print Descriptors to Customize the PRF

The PRF uses two types of print descriptors:

1. Profiles, which PRF users can specify on either the panel or command interface. These print descriptors are created by PRF users, are used to establish print options and default values for types of print jobs, and have a print-descriptor type of **PRD_PRINT_PROFILE**.
2. Printer definitions, which PRF users can also specify on either the panel or command interface. These print descriptors can represent the different types of printers in your organization, and should have a print-descriptor type of **PRD_PRESENTATION_DEVICE** or **PRD_PRESENTATION_SERVER**. These two print-descriptor types are architected values, although PrintManager provides no error checking to verify that these print descriptor types are used. Except for print descriptors of type **PRD_PRINT_PROFILE**, any print descriptor type can be used for printer definitions.

Note: For more information on print-descriptor types, refer to "Assigning Print-Descriptor Types" on page 3-11.

As described in "Using the PrintManager Samples" on page 3-15, PrintManager supplies a set of StdPrds that include sample profiles and printer definitions that can be used as a starter set for PRF users. For more information on modifying these samples to customize the PRF for your organization's users, refer to "Customizing the PrintManager Samples" on page 3-17.

Whether you modify the PrintManager-supplied print descriptors or create your own, you can use printer definitions to make available or restrict print system components to a PRF user as follows:

- An option will appear on the PRF panels only if the option and its validation information is defined in the printer definitions used for a print job. Therefore, if you do not want an option to appear, ensure that this option is not included in any printer definitions available to a user or group of users. For example, if you do not use the **CKPTSEC** option in your organization, you could delete this option from AFP CUTSHEET to ensure that this option is not included if you modify AFP CUTSHEET or use it to create other print descriptors.
- You can use print-descriptor groups to make printer definitions available on a system wide basis or to selected user groups. For more information, refer to "How to Set Up Print Descriptors for Your System" on page 3-14.
- You can use system security functions to restrict access to print-descriptor groups. For more information, refer to "Access Authority to Print Descriptor Groups" on page 3-10.

Note: PRF users may create profiles that, over time, are no longer used. You may want to provide users with information about the PrdT PRD DEL command, which can be used to delete unused profiles. For more information, see "PRD DEL" on page 5-8.

Chapter 9. Print Options

This chapter:

- Provides an overview of the PrintManager print options
- Discusses print option defaults, validation, and merging
- Describes AFP resources.

Overview of PrintManager Print Options

PrintManager defines a set of print options (documented in Appendix A, PrintManager Print Options) for the SAA environments. The PrintManager Interface supports a subset of these print options in each environment as shown (with an "X") in Table 9-1 and Table 9-2. Table 9-1 lists the print options that you can specify individually, and Table 9-2 lists those options whose values are used to build an inline form definition (when the **FORMDEF** option has a value of ***BUILD**).

The subset of options that the PrintManager Interface supports is determined by the printing capabilities of each SAA environment. All PrintManager options, however, are supported by the API because they can be set in a print descriptor and queried to determine the capabilities of a print system component.

Appendix A, PrintManager Print Options also documents valid values for the defined options, but you can modify these values or define additional values for your organization's needs. If you want, you can also define additional print options. Any additional values and options you define, however, are ignored by the PrintManager Interface. Installation defined options or values may be used to specify printing for or provide information about print system components that are unique to that installation.

You can specify print options in an application written to the PrintManager Interface or in a print descriptor. The PRF component of IBM SAA PrintManager also provides a set of options that are consistent with those provided with the PrintManager Interface. For more information, refer to *IBM SAA PrintManager User's Guide*.

Table 9-1 (Page 1 of 3). API Print Options by Environment

Option	MVS	VM	OS/400	OS/2
BACKGRNDMIX				
BARCODESET				
CAPSGRAPHIC				
CAPSRASTER				
CC	X	X	X	
CKPTLINE	X			
CKPTPAGE	X	X		
CKPTSEC	X			
CLASS	X	X		
COLORNUM				
COPIES	X	X	X	

Table 9-1 (Page 2 of 3). API Print Options by Environment				
Option	**MVS**	**VM**	**OS/400**	**OS/2**
DATACK	X	X	X	
DATATYPE	X	X	X	
DEVDRIVERTYPE				
DEVMODEL				
DNLDFONT				
DOCOWNER		X		
DUPLEX		X	X	
FCB	X	X		
FIDELITY			X	
FLASHCNT	X	X		
FLASHNAME	X	X		
FOCASET				
FONT	X	X	X	
FOREGRNDMIX				
FORM	X	X	X	
FORMDEF	X	X	X	
GDDMDEVTOKEN	X	X		
GRAPHICSET				
HORIZRES				
IMAGESET				
INBIN		X	X	
INDEX	X			
LINDEX	X			
LINECT	X			
LPI				
MEDIATTRIBUTES				
MEDIATYPE				
MEDIAXLEFTCLIP				
MEDIAXPELS				
MEDIAXRIGHTCLIP				
MEDIAXWIDTH				
MEDIAYBOTTOMCLIP				
MEDIAYHEIGHT				
MEDIAYPELS				
MEDIAYTOPCLIP				
MESSAGES	X	X		
MODCASET	X	X		
MODIFYNAME	X	X		

Table 9-1 (Page 3 of 3). API Print Options by Environment				
Option	**MVS**	**VM**	**OS/400**	**OS/2**
MODIFYTRC	X	X		
OUTDISP	X	X	X	
OUTMETHOD	X			
OUTPUTID	X	X	X	
OUTQ			X	
OVERLAY	X	X	X	
PAGEDEF	X	X	X	
PAGERANGE			X	
PAGESEG	X	X	X	
PRMODE	X	X	X	
PRTENVIRONMENT				
PRTY	X		X	
RSCSID		X		
SCHEDULE			X	
TECHNOLOGY				
TEXTSET				
TRC	X	X	X	
UCS	X			
VERTRES				
WRITER	X			

Table 9-2. API Print Options Used to Build Form Definitions				
Option	**MVS**	**VM**	**OS/400**	**OS/2**
DUPLEX	X	X	X	
FLASHNAME	X	X	X	
INBIN	X	X	X	
JOGOUT	X	X	X	
LEFTMAR	X	X	X	
OVERLAY	X	X	X	
PRTDIRECTION	X	X	X	
PRTQUAL	X	X	X	
TOPMAR	X	X	X	

Print Option Defaults, Validation, and Merging

By creating StdPrds, you can create default values and valid values for print options. Default values can provide job and printer defaults, and valid values in a print descriptor can be used to define the capabilities of a print option. The capabilities or valid values can be used by the PrintManager Interface to validate values specified for a print job before the job is submitted to the spool.

You can also use the PDBLDD (Build Descriptor) verb to merge print options by creating a composite print descriptor. The following sections discuss print option defaults, validation, and merging. For more information on StdPrds, refer to "Contents of Standard Print Descriptors (StdPrds)" on page 3-2.

Print Option Defaults

With StdPrds, you can set defaults for classes of print jobs or for printers. Any application written to the PrintManager Interface that specifies a print descriptor (on the SPROPEN (Open) or SPRSOPT (Set Option) verbs) uses the default values in that print descriptor as defaults.

For example, if a print descriptor specifies a default of 10 copies for a printer, any application that specifies the print descriptor to the PrintManager Interface when submitting a print job will have 10 copies as a default value. Similarly, if a print descriptor is used to specify one copy as a default for a class of jobs (such as a standard memo), any application that specifies this print descriptor will have one copy as the default.

A PrintManager Interface application can override the default values in a print descriptor, but only within the range of valid values specified in the validation rules in the print descriptor. For example, if a print descriptor specifies a default value of one copy and a range of valid values from one to ten copies, an application could specify five copies to override the default, but not fifteen copies. For more information, refer to "Print Option Validation."

PrintManager supplies defaults with some of the samples (refer to "Using the PrintManager Samples" on page 3-15). You will, however, probably want to set defaults for any print options you expect to use in your organization. Setting defaults provides application users with defaults and a limited set of values they can use to override defaults. Setting defaults also prevents print option values in an application from defaulting to hardware or system default values.

Note: When you set defaults for data of type **PRTMGR_STRL**, the API removes any leading or trailing blanks. For example, if you entered the literal value ' STANDARD ', the default will be saved as 'STANDARD'.

Print Option Validation

PrintManager uses print descriptors to validate print options in a PrintManager Interface application as follows:

1. If the print option exists in the print descriptor, but the print option does not specify a validation rule, valid values, or a default value, validation does not occur.
2. If the print option exists in the print descriptor and it specifies a validation rule, the values specified in the application must adhere to the validation rule specified in the print descriptor. For example, validation will fail if a value with a

validation rule of **PRD_RANGE** is specified in an application and the referenced print descriptor specifies a validation rule of **PRD_LIST**.

Values specified in an application must also match a valid value specified in the print descriptor. For example, if the valid value specified in the print descriptor is a range 1 – 10, an application can specify a value of 4 but not a value of 15. If the print descriptor specifies **PRD_NOVALIDATION**, validation does not occur.

3. If an application attempts to specify an invalid value, PrintManager returns an error, and the previous valid value (if any) remains set. The application can then respecify a valid value.

Appendix A, PrintManager Print Options lists the print options and validation rules defined by PrintManager. You can, of course, bypass validation (by either not specifying a validation rule or by specifying **PRD_NOVALIDATION** in a StdPrd). For most options, however, you will want to specify a validation rule in a StdPrd, just as you will generally want to assign a default value in the print descriptor.

The PrintManager validation rules are specified in a print descriptor as follows:

Rule	Description
PRD_NOVALIDATION	Validation does not occur and any value is valid.
PRD_STRING	Valid values must contain a positive integer which is the maximum length of the string. The value of the positive integer cannot exceed 32767. The strings being validated can contain any character except the null character, but cannot be zero length, all blanks, or exceed the maximum length defined by the valid value.
PRD_RANGE	Valid values must consist of three numbers: an integer which specifies the number of significant decimal places, a minimum value, and a maximum value. These numbers are separated by a delimiter character (blank or comma). When you define an integer range, the number of decimal places should be 0. For ranges in any other precision, the largest number of *significant* decimal places is 10. All values must be valid within the specified range; no null ranges can exist. Depending on how you specify the decimal precision for a range, PrintManager may reset the valid values and default values that you have specified either by extending or truncating these values. For example, if you specify a decimal precision of 0, a minimum value of 1.0, a maximum value of 9.0, and a default value of 9.9, PrintManager will truncate the default value to 9. because you specified a decimal precision of 0. The minimum value is truncated to 1. and the maximum is truncated to 9. in this case. PrintManager issues warning messages whenever this type of truncation occurs. Similarly, if you specify a decimal precision of 3, a minimum value of 1.1, a maximum value of 10.18999, and a default value of 8.8, PrintManager will reset the default value to 8.800 because you specified a decimal precision of 3. In this case, the minimum value is reset to 1.100 and the maximum is reset to 10.189. PrintManager issues warning messages whenever this type of truncation occurs. For **PRD_RANGE** values, therefore, it is recommended that you specify a decimal precision that corresponds to the greatest

number of significant decimal places that you intend to allow for print option values. For example, if you (or the operating system) will allow only integer values for a print option, set the decimal precision to 0. If, however, you will allow values with 5 significant decimal places (for example, for a value of 10.18999), set the decimal precision to 5 to avoid truncation of the significant decimal places.

PRD_LIST Valid values consist of a list of values separated by a delimiter character (a blank or a comma). For a specific value to be valid, it must appear on this list. The maximum length for a value is determined by the longest item on the list. At least one value must be specified for the list; no null lists can exist. Validation of list values is *not* case sensitive.

The format of numeric values supported by PrintManager is as follows:

- The value of the decimal character is a period (.).
- The negative indicator is a minus sign (-). There is no trailing negative indicator.
- There is no thousands-separator character.

Any string or part of a string enclosed within a pair of double quotes (" ") will be treated as a literal part of a value. This allows a comma, a blank, a semicolon, an equal sign, or a double quote (when enclosed by paired double quotes) to be used as part of a value. If no matching double quote is found, the end of the string is assumed to be the zero terminator.

For a validation rule of **PRD_LIST**, the values are validated using a character-for-character compare. For example, if the list of valid values is YES, NO, MAYBE, the value of YES will be valid, but the value of "YES" will not be valid. If "YES" is supposed to be valid, it must be included in the list of valid values (YES, "YES", NO, MAYBE).

For a validation rule of **PRD_LIST**, validation is not case sensitive. For example, if the valid value specified in the print descriptor is YES, application specified values of YES, yes, Yes, and yeS are among the possible valid values. Similarly, the same values would be valid if the print descriptor specified a valid value of yes. As a result, the print descriptor need only specify a single valid value (in any combination of case) to ensure that all combinations of case specified in an application are valid. Conversely, you cannot restrict a value to a particular combination of case. For example, valid values of YES, yes, Yes in a print descriptor will not exclude yES as a valid value (either as specified in an application or as a default value).

For the validation rules **PRD_NOVALIDATION** and **PRD_STRING**, double-byte character strings can be specified as part of a value. Values can contain single-byte character strings, double-byte characters strings, or both. However, double-byte character strings must be an even number of bytes and EBCDIC double-byte character strings must be delimited by the shift-out and shift-in characters. For validation rules of **PRD_STRING**, the maximum length defines the number of bytes.

How Print-Option Information Is Merged

As described in "Contents of Standard Print Descriptors (StdPrds)" on page 3-2, you can use the PDBLDD (Build Descriptor) verb or the PRDBUILD Print Descriptor Tool tag to build a composite print descriptor from a print descriptor that contains print-descriptor references. The composite print descriptor contains merged print-option information and the print-descriptor references (which are marked **PRD_USED** and will not be used again in the current API verb session or PRD TOOL session). If the composite print descriptor is stored, these references (marked **PRD_USED**) will also be stored.

During a merge, the *existing* print-option information (in the original print-descriptor containing the print-descriptor references) is compared to the *new* information in the referenced print descriptors.

Merging print-option information occurs in three steps, each of which must complete successfully for the information to merge. The steps are:

1. The new (referenced) rule is compared to the existing rule. If they do not match then the merge fails immediately, except when one of the rules is **PRD_NOVALIDATION**. In this case, the most restrictive rule becomes the new rule.[1]
2. Valid values are combined into a composite value according to the composite rule, as follows:

   PRD_NOVALIDATION
   Valid values is null. If the new rule is **PRD_NOVALIDATION**, the existing rule becomes the new rule.

   PRD_STRING
   Each valid value is a single number representing a maximum value length, and the smaller of the two numbers becomes the new value. A new value cannot be 0.

   PRD_RANGE
   Valid values consist of the number of significant decimal places, a minimum value, and a maximum value. The new value is the larger number of significant decimal places, the larger of the two minimum values, and the smaller of the two maximum values. The largest allowable number of significant decimal places is 10. If the minimum value is greater than the maximum value, the merge will fail.

   Note: The valid values for the combined print option can be altered to reflect the new minimum and maximum value with the new specified precision.

   PRD_LIST Valid values consist of a list of items separated by blanks. The new list consists of only those items that are identical in both lists. This comparison is not case sensitive, and the new list must contain at least one item.

---

1 Because **PRD_NOVALIDATION** is the least restrictive rule, the resulting new rule should never be **PRD_NOVALIDATION**.

3. If a default value is set in the original print descriptor, this value remains the default during a merge (any default values specified in a referenced print descriptor are not used). However, if no default is set in the original print descriptor, the default value in a referenced print descriptor is used. In this case, the default value from the referenced print descriptor is validated against the composite rule and valid values. If the default is not successfully validated, the merge fails.

   If multiple references exist and there is no default set in the original print descriptor, then the referenced default values will be validated (and successively set as the new default) in ascending sequence order of the referenced print descriptors.

Note: PDSOPT (Set Print Option) assigns a default validation of **PRD_NOVALIDATION** and a valid value of null to any print option that is specified without validation. Information in this type of option will merge successfully with information in any referenced option.

The following sequence shows an example of how print-option information is merged.

Table 9-3. Example of Merging Print-Option Information

Action	Comments
Initial Edit Session Values: Option: A, B Default Value: (none), 2 Valid. Rule: string, range Valid Values: 10 max., 0 decimal places, range 2-5	
Reference a Print Descriptor: Option: A, B Default Value: FRED, 2 Valid. Rule: string, range Valid Values: 8 max., 2 decimal places, range 1.00-3.00	A print descriptor is referenced, and the print descriptor contains new values for options A and B that will be merged with the initial edit session values.
Merged Edit Session Values: Option: A, B Default Value: FRED, 2 Valid. Rule: string, range Valid Values: 8 max., 2 decimal places, range 2.00-3.00	For option A, the value of FRED and maximum string length of 8 become the merged information for the current edit session. For option B, a new range of 2 – 3 is the merged range since it is the intersection of the initial range of 2 – 5 and the range of 1 – 3 from the referenced print descriptor. The number of significant decimal places for this range is reset to 2.

AFP Resources

On some of the options described in Appendix A, PrintManager Print Options, you can specify the AFP resources you need for a print job. AFP resources include form definitions, page definitions, fonts, overlays (electronic forms), and page segments (graphic images). AFP resources and the corresponding print options are as follows:

- **Form definitions**. Form definitions contain information about the printing medium (for example, a sheet of standard memo paper) and how it is handled. You can use a form definition to specify number of copies, overlays for the job, duplexing or simplexing, and the starting point for placement of data on the page.

  You specify the form definition on the **FORMDEF** print option. You can also build a simple form definition to be created when the job is written to the spool (refer to Table 9-2 on page 9-3 for a list of the print options whose values you can use to build a form definition).

  Note: If you build a form definition but do not specify print options, a default form definition with a page offset of (0, 0) and no other specifications will be built.

- **Page Definitions**. Page definitions contain information about the placement and formatting of line data on the print medium. For example, you can use page definitions to specify line length, page length, fonts for a print job, and rotated printing. You specify a page definition on the **PAGEDEF** print option.

- **Overlays** (electronic forms). Overlays are collections of coded information that describe where to put boxes, lines, shading, text, logos, and graphic images on forms. When printed with variable application data, overlays can replace the need for preprinted forms. You specify overlays on the **OVERLAY** print option.

- **Fonts**. Fonts are a family or assortment of characters in a given size and style. You specify fonts on the **FONT** print option.

- **Page segments**. Page segments can contain text, raster graphic data, or both. Examples of page segments include logos, signatures, bar charts, and drawings. Page segments can be printed anywhere on a page or at the same place in every page of a print job. The print data must specify page segments and their placement. You specify page segments on the **PAGESEG** print option.

When specifying AFP resources in print options:

- You can use PrintManager print options to specify AFP resources only if **DATATYPE** = **MODCAP**, **AFPDS**, **AFPDSLINE**, or **LINE**.

- You can specify multiple values on the **FONT**, **OVERLAY**, or **PAGESEG** options. For more information, refer to the descriptions of these options in Appendix A, PrintManager Print Options. For example, in VM you might specify:

  FONT *CHAR GT10 GT12
  OVERLAY *INLINE O1S16 OVLY3820 A ***SYSTEM** O11040

- You can specify only one value on the **FORMDEF** and **PAGEDEF** options.

- In addition, you must be aware of how your printer driver processes the AFP resources that you specify with PrintManager. For example, if you are using PSF/VM as your printer driver and you specify an inline overlay in your application, that overlay will not be used unless you also specify a form definition which names that overlay. For example, you may want to specify that a form definition is built as follows:

OVERLAY *INLINE O1596 OVLY3820 A ***FORMDEF** O1596
FORMDEF *BUILD

Refer to your printer-driver documentation for more information on how AFP resources are processed.

Inline Resources

Depending on the system, you can usually specify that AFP resources will be placed *inline* with a print job as the job is written to the spool. You do this by using the ***INLINE** keyword and specifying a fully qualified resource file name on the print option. Inline resources can be routed to another system because they are "packaged" with the print job and do not have to be sent separately or to reside on the remote system.

For example, suppose that a TSO user wants to print a job on a VM system using a particular overlay. A PrintManager application or print descriptor can specify that the overlay will be placed inline with the print job as the job is written to the TSO spool; therefore, the overlay does not have to reside on the VM system.

When specifying inline AFP resources, however, you must be aware of whether inline resources are supported on the system that will print the job. Refer to your printer-driver documentation for information on the support for inline AFP resources.

Note: For example, if you use PrintManager to submit a print job that contains an inline page definition or form definition where **CC** is set to NO, PSF/MVS cannot process the job.

Appendix A. PrintManager Print Options

The options in this appendix are consistently defined across the environments supported by PrintManager, although they may not be supported by PrintManager in all environments (refer to "Environment Notes"). All options, however, can be set and queried in all environments. This appendix describes the option values provided with PrintManager, but you can modify these values or define additional values for your organization's needs. If you want, you can also define additional print options. Any additional values and options you define, however, are ignored by the PrintManager Interface.

Print-option information is described in this format:

- Option name and description
- Validation rules and valid values
- Environment notes (environment support and use).

Using PrintManager Print Options for AFP

If you are using AFP, you must understand how the printer driver will use the PrintManager print option values that you specify. For more information, refer to the publications listed under "For Printing" on page 1-6. For a summary of how the PrintManager print options correspond to operating system commands and parameters, refer to Table A-1 on page A-24.

To successfully distribute print jobs between MVS and VM systems, the print options you specify must be supported by PrintManager in both environments. Refer to "Environment Notes" for information on the print options you want to specify and Table 9-1 on page 9-1 and Table 9-2 on page 9-3 for a summary of the options that are valid in both environments. Although PrintManager provides a full range of print options, typically you need only a small set of these options for a specific type of print job.

OPTION

BACKGRNDMIX Specifies the background-mix toning supported by the printer.

Validation Rule: **PRD_LIST**

Valid Values:

OR	Logical OR
OVERPAINT	Overpaint
XOR	Logical XOR
LEAVEALONE	Leave alone

Environment Notes: This option can only be set and queried. It is used to describe the capabilities of a print system component.

BARCODESET Specifies the MO:DCA-P bar code capability supported by the printer driver.

Validation Rule: **PRD_LIST**

Valid Values:

NONE	No bar code support
BCD1	BCD1 support

Environment Notes: This option can only be set and queried. It is used to describe the capabilities of a print system component.

CAPSGRAPHIC Specifies the graphics capability supported by the printer.

Validation Rule: **PRD_LIST**

Valid Values:

KERNING	Kerning is supported.
OUTLINEDEF	Printer has default outline font.
IMAGEDEF	Printer has default image font.
MARKERSDEF	Default markers will be scaled.

Environment Notes: This option can only be set and queried. It is used to describe the capabilities of a print system component.

CAPSRASTER Specifies the raster capabilities supported by the printer.

Validation Rule: **PRD_LIST**

Valid values:

BITBLT	GpiBitBlt is supported.
BANDING	Printer supports banding.
SCALING	Printer supports scaling.
SETPEL	GpiSetPel supported.
FONTS	Printer supports raster fonts.

Environment Notes: This option can only be set and queried. It is used to describe the capabilities of a print system component.

CC Specifies the type of carriage control characters contained in a data stream.

Validation Rule: **PRD_LIST**

Valid Values:

NO	Does not contain carriage control characters.
YES	Contains ANSI carriage control characters.
MACHINE	Contains machine carriage control characters.

Environment Notes: This option is functionally supported by a printer driver in the MVS, VM, and OS/400 operating systems. If **DATATYPE** is **MODCAP**, **AFPDS**, or **AFPDSLINE** and **CC** is not set or set to NO, **CC** is changed to YES for VM and OS/400 or MACHINE for MVS.

CKPTLINE Specifies the maximum number of lines in a logical page. JES also uses **CKPTLINE** with **CKPTPAGE** to determine when to take data-set checkpoints. Checkpoints are used to save all information necessary to resume printing if an error occurs.

Note: If you do not set a value for this option, a JES or PSF default will be used.

Validation Rule: **PRD_RANGE**

Valid Values: 0 (decimal places), 0 (minimum), 32767 (maximum)

Environment Notes: This option is functionally supported by a printer driver in the MVS operating system.

CKPTPAGE

Specifies the number of logical pages to be printed before taking a checkpoint. Checkpoints are used to save all information necessary to resume printing if an error occurs.

Note: If you do not set a value for this option, a JES or PSF default will be used.

Validation Rule: **PRD_RANGE**

Valid Values: 0 (decimal places), 0 (minimum), 32767 (maximum)

Environment Notes: This option is functionally supported by a printer driver in the MVS and VM operating systems. On MVS, if you do not specify a value, an installation-defined default is used.

CKPTSEC

Specifies the number of seconds to elapse before taking a checkpoint. Checkpoints are used to save all information necessary to resume printing if an error occurs.

Note: If you do not set a value for this option, a JES or PSF default will be used.

Validation Rule: **PRD_RANGE**

Valid Values: 0 (decimal places), 1 (minimum), 32767 (maximum)

Environment Notes: This option is functionally supported by a printer driver in the MVS operating system. If you specify both **CKPTPAGE** and **CKPTSEC**, the **CKPTSEC** value is used unless the installation has specified that checkpointing is based only on pages.

CLASS

Specifies the print output class (a printer or group of printers) for job scheduling.

Validation Rule: **PRD_LIST**

Valid Values: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F, G, H, I, J, K, L, M, N, O, P, Q, R, S, T, U, V, W, X, Y, Z, *

Note: These valid values make up the entire set of values PrintManager supports for **CLASS**, but your organization may define a subset of these values for actual use.

Environment Notes: This option is functionally supported by a printer driver in the MVS and VM operating systems.

COLORNUM

Specifies the number of distinct colors supported by the printer, including reset (grey scales count as distinct colors).

Validation Rule: **PRD_RANGE**

Valid Values: 0 (decimal places), 1 (minimum), 2^{31} (maximum)

Environment Notes: This option can only be set and queried. It is used to describe the capabilities of a print system component.

COPIES

Specifies how many copies of a print job will be printed.

Validation Rule: **PRD_RANGE**

Valid Values: 0 (decimal places), 1 (minimum), 255 (maximum)

Environment Notes: This option is functionally supported by a printer driver in the OS/400, MVS, and VM operating systems.

DATACK

Specifies which printer errors are reported. Two types of data check errors can occur. Print-positioning errors result from attempts to print outside the valid printable area. Invalid-character errors result from attempts to use a code point that is not assigned to a character.

If print-positioning errors are reported (UNBLOCK or BLKCHAR values), your printer may provide exception highlighting to mark the location of the error. For more information, refer to your printer documentation or *Advanced Function Printing: Printer Information*.

Note: If you do not set a value for this option, a PSF or OS/400 system default will be used.

Validation Rule: **PRD_LIST**

Valid Values:

BLOCK	Block (do not report) print-positioning errors and invalid-character errors. Printing continues but data may be lost.
UNBLOCK	Report all errors.
BLKCHAR	Block only invalid-character errors. Print-positioning errors are reported normally.
BLKPOS	Block only print-positioning errors. Invalid-character errors are reported normally.

Environment Notes: This option is functionally supported by a printer driver in the MVS, VM, and OS/400 operating systems. In OS/400, the value for this option:

- Is set to UNBLOCK if the **FIDELITY** option is set to ABSOLUTE.
- Is changed to BLOCK at print time if the **FIDELITY** option is set to CONTENT and the print file is destined for a twinax attached printer.

DATATYPE Identifies the data stream type of the print job.

Validation Rule: **PRD_LIST**

Valid Values:

AFPDS	An AFPDS data stream (with X'5A' carriage control characters)
AFPDSLINE	A mixed data stream (AFPDS and line data)
ASCII	An IBM ASCII data stream
DBCS_ASCII	An IBM double-byte character set ASCII data stream
LINE	Either a line data stream (for example, 1403) or unformatted data. If the data contains carriage control characters, **CC** must also be set to YES or MACHINE.
MODCAP	Any MO:DCA data stream
PCL	An HP data stream
PM_Q_RAW	A passthrough data stream

	PM_Q_STD	The standard Presentation Manager data stream
	POSTSCR	A PostScript data stream
	SCSSNA	An SNA SCS data stream (370 environment)
	SCS400	An OS/400 SCS data stream

Environment Notes: Supported in the VM, MVS, and OS/400 environments as follows:

- In VM, MVS, and OS/400, if **DATATYPE** is **MODCAP**, **AFPDS**, or **AFPDSLINE** and **CC** is not set or set to NO, **CC** is changed to YES for VM and OS/400 or MACHINE for MVS.
- In VM and MVS you can place any of the above data streams on the spool.
- In OS/400, you can place the **AFPDSLINE**, **LINE**, **MODCAP**, **AFPDS**, **ASCII**, and **SCS400** data streams on the spool. PrintManager assumes a default of **SCS400** in OS/400.

  For the **AFPDSLINE** and **LINE** data streams, a PrintManager Interface application must either:

  - If using the SPRWRIT (Write) verb, write the data a line at a time
  - If using the SPRADDF (Add File) verb, specify a file that contains either a single line or a single AFPDS structured field per record.

  In OS/400, If the data is line data containing double byte characters, the **PRMODE** option must be set to SOSI1 or SOSI2 and your system must be an IGC machine.

DEVDRIVERTYPE Identifies the printer driver.

Validation Rule: **PRD_LIST**

Valid Values:

PSF	Print Services Facility driving IPDS printers, for example, the IBM 3800 Printing Subsystem, IBM 3820 Page Printer, or IBM 3827 Page Printer.

Environment Notes: This option can only be set and queried. It is used to describe the capabilities of a print system component.

DEVMODEL Identifies the type and model number of the printer.

Validation Rule: **PRD_NOVALIDATION**

Valid Values: Any character string that represents the type and model number of the printer.

Environment Notes: This option can only be set and queried. It is used to describe the capabilities of a print system component.

DNLDFONT Specifies the number of fonts that can be downloaded to the printer at any one time.

Validation Rule: **PRD_RANGE**

Valid Values: 0 (decimal places), 0 (minimum), 2^{31} (maximum)

Environment Notes: This option can only be set and queried. It is used to describe the capabilities of a print system component.

DOCOWNER Specifies the print-job owner.

Validation Rule: **PRD_NOVALIDATION**

Valid Values: Any character string representing the print-job owner in the following format:

FORMAT

userid	The userid of the user.

Environment Notes: This option is functionally supported in the VM operating system. This value correlates to the FOR parameter on the CP SPOOL command. This value indicates the userid under which printed or punched output is produced.

In VM, **RSCSID** and **DOCOWNER** are mutually exclusive. If both are specified, the **DOCOWNER** value will be used. If the print job must be directed to a remote system using **RSCSID**, **DOCOWNER** must not be specified.

DUPLEX Specifies how the job will be printed on the media (for example, simplexing or duplexing). Values from this option are used to build an inline form definition when the ***BUILD** keyword is specified for the **FORMDEF** option.

Validation Rule: **PRD_LIST**

Valid Values:

NO	Simplexed output.
YES	Two-sided output.
TUMBLE	Tumble two-sided output.

Environment Notes: This option is functionally supported by a printer driver in the OS/400, MVS, and VM operating systems. In VM and OS/400, you can specify **DUPLEX** without specifying ***BUILD** for the **FORMDEF** option. In OS/400, the value specified for **DUPLEX** will override any value specified in the active form definition.

FCB Specifies the name of a forms control buffer (FCB) for formatting line data for a 3800-1 printer.

Validation Rule: **PRD_STRING**

Valid Values: 4 (maximum length)

Environment Notes: This option is functionally supported by a printer driver in the MVS and VM operating systems.

FIDELITY Specifies if the printer driver will try to print a file that contains errors.

Validation Rule: **PRD_LIST**

Valid Values:

ABSOLUTE	The job is printed only if the file can be printed exactly as specified by the data stream and external controls. If **DATACK** is set, the **DATACK** value is ignored and the system sets it to UNBLOCK.

CONTENT The file is printed if at all possible. All available exception handling is used.

Environment Notes: This option is functionally supported in OS/400 (only AFP).

FLASHCNT

Specifies the number of pages to print with the forms flash specified in the **FLASHNAME** option.

Validation Rule: **PRD_RANGE**

Valid Values: 0 (decimal places), 0 (minimum), 255 (maximum)

Environment Notes: This option is functionally supported by the MVS and VM operating systems. AFP printer drivers may provide different function support than the operating systems. For more information, refer to your printer driver documentation.

FLASHNAME

Specifies the name of a forms flash for printing photographic images on an IBM 3800. Values from this option are used only to build an inline form definition when the ***BUILD** keyword is specified for the **FORMDEF** option.

For information on forms flashing, refer to *Forms Design Reference for the 3800.*

Validation Rule: **PRD_STRING**

Valid Values: 4 (maximum length)

Environment Notes: This option is functionally supported by a printer driver in the MVS and VM operating systems.

FOCASET

Specifies the SAA font capability supported by the printer driver.

Validation Rule: **PRD_LIST**

Valid Values:

NONE No FOCA font support.

FOCA2 FOCA2 fonts supported.

Environment Notes: This option can only be set and queried. It is used to describe the capabilities of a print system component.

FONT

Specifies one of the following:

- The location of fonts that are included in a line, page, or mixed data stream
- The names of fonts to be associated with table reference characters (TRCs) in a document.

Validation Rule: **PRD_NOVALIDATION**

Valid Values: Any string specifying the fonts for the job in one or more of the following formats:

***INLINE name** This keyword is supported in MVS and VM. For only AFP print jobs, use ***INLINE** to select a font from a system library (on the system where the print job is submitted) and place it inline with the print job. Refer to your printer-driver documentation for information on how inline resources are handled in each environment. The *name* field is the fully qualified name of the coded font.

In VM, if the *name* field is not a fully qualified name, the default file type is based on the two-character prefix of the file name. If the prefix is "X0," the default file type is FONT3820. If the prefix is "X1," "X2," "X3," or "X4," the default file type is FONT38PP. In VM, a file mode of "A" is assumed. To search all accessed disks, "*" can be specified as a wild card.

***SYSTEM name**

This keyword is supported in MVS and VM. For only AFP print jobs, use ***SYSTEM** to select a font (specified in *name*) from a system library on the system where the print job is to be printed. The *name* field is the 1—8 character name of a font. If the font name is not specified, a system default is used.

Note: If the ***SYSTEM** keyword would be the first keyword of multiple keywords and values, the first ***SYSTEM** keyword can be omitted. For example:

```
X0GT10 *SYSTEM X0GT12 *INLINE X0GB1
```

For VM only, the *name* field can also include an optional 1—8 character file type. This file type is based on the two-character prefix of the *name* field. If the prefix is "X0," the default file type is FONT3820. If the prefix is "X1," "X2," "X3," or "X4," the default file type is FONT38PP.

***CHAR name**

This keyword is supported in MVS, VM, and OS/400. For line data print jobs with TRC controls, use the ***CHAR** *name* field to specify up to four font names or (for 3800 line printers) character-arrangement tables that are used for the print job. If the print file does not contain table reference characters (TRCs), only the first font is used. If the print file contains TRCs, the **TRC** option must be set to YES and the fonts must be specified in the order that corresponds to the TRC values. The *name* field must be 1—4 characters.

In OS/400, the object type is "*FNTRSC."

You can specify multiple values (up to 4) for the ***CHAR** keyword by either specifying the ***CHAR** keyword before each

value or by specifying the ***CHAR** keyword followed by multiple values. For example:

***CHAR** GT10 ***CHAR** GB10 or ***CHAR** GT10 GB10

In VM and MVS, to specify the name of a font for a line data print file and to specify that the font resides in a system library, use: ***CHAR** *name* ***SYSTEM** *name*, where the name field is the same for both keywords. ***INLINE** can also be paired with ***CHAR** to specify that the font to be used will be included inline with the print file.

Environment Notes: Refer to the environment information above.

FOREGRNDMIX Specifies the foreground-mix toning supported by the printer.

Validation Rule: **PRD_LIST**

Valid Values:

OR	Logical OR
OVERPAINT	Overpaint
XOR	Logical XOR
LEAVEALONE	Leave alone
AND	Logical AND
SUBTRACT	(inverse source) AND dest
MASKRCNOT	Source AND (inverse dest)
ZERO	All zeros
NOTMERGSRC	Inverse (source OR dest)
NOTXORSRC	Inverse (source XOR dest)
INVERT	Inverse (dest)
MERGESRCNOT	Source OR (inverse dest)
NOTCOPYSRC	Inverse (source)
MERGENOTSRC	Inverse (source) OR dest
NOTMASKSRC	Inverse (source AND dest)
ONE	All ones

Environment Notes: This option can only be set and queried. It is used to describe the capabilities of a print system component.

FORM Specifies an installation-defined forms code that specifies the media for a print job.

Validation Rule: **PRD_NOVALIDATION**

Valid Values: Any character string specifying the forms code.

Environment Notes: This option is functionally supported by a printer driver in the OS/400, MVS, and VM operating systems.

FORMDEF Specifies the form definition for an AFP print job.

Validation Rule: **PRD_NOVALIDATION**

Valid Values: Any string specifying the form definition in one of the following formats (all keywords are supported in VM, MVS, and OS/400):

***INLINE name**

Use ***INLINE** to select a form definition from a system library (on the system where the print job is submitted) and place it inline with the print job. Refer to your printer-driver documentation for information on how inline resources are handled in each environment. The *name* field is the fully qualified name of the file containing the form definition.

In VM, if the *name* field is not a fully qualified name, the default file type is FDEF38PP and a file mode of "A" is assumed. To search all accessed disks, "*" can be specified as a wild card.

In OS/400, the object type is "*FORMDF."

***SYSTEM name**

Use ***SYSTEM** to select a form definition (specified in *name*) from a system library on the system where the print job is to be printed. The *name* field is the 1—8 character name of a form definition. ***SYSTEM** is also used to name a form definition that is already a part of the print data. The ***SYSTEM** keyword is optional and can be omitted. For example:

```
F1A10110
```

For VM only, the *name* field can also include an optional 1—8 character file type. In VM, the default file type is FDEF38PP.

In OS/400, the object type is "*FORMDF."

***BUILD**

Use ***BUILD** to specify that a form definition will be built and placed inline using values from the following print options: **TOPMAR**, **LEFTMAR**, **INBIN**, **PRTDIRECTION**, **DUPLEX**, **PRTQUAL**, **OVERLAY**, **JOGOUT**, **FLASHNAME**.

Environment Notes: Refer to the environment information above.

GDDMDEVTOKEN

Specifies a GDDM device token. It is required by PrintManager for printing documents where the data stream must be transformed from MO:DCA-P format to AFPDS format. This option is valid only when **DATATYPE** = **MODCAP** and **MODCASET** = **AFPDS**. For more information on the MO:DCA-P to AFPDS data stream transform, refer to *Systems Application Architecture Common Programming Interface PrintManager Reference*.

Validation Rule: **PRD_LIST**

Valid Values: Any list of the valid GDDM device tokens for GDDM family 4 devices. For more information, refer to your GDDM documentation.

Environment Notes: This option is functionally supported by a printer driver in the MVS and VM operating systems.

GRAPHICSET Specifies the MO:DCA-P graphics capability supported by the printer driver.

Validation Rule: **PRD_LIST**

Valid Values:

NONE	No GOCA graphics support.
DR2V0	Identifies GOCA DR2V0 graphics support.

Environment Notes: This option can only be set and queried. It is used to describe the capabilities of a print system component.

HORIZRES Identifies the horizontal resolution of the specified device in pels per meter.

Validation Rule: **PRD_RANGE**

Valid Values: 0 (decimal places), 1 (minimum), 2^{31} (maximum)

Environment Notes: This option can only be set and queried. It is used to describe the capabilities of a print system component.

IMAGESET Specifies the MO:DCA-P image capability supported by the printer driver.

SYSTEM **VALID VALUES**

Validation Rule: **PRD_LIST**

Valid Values:

NONE	No image support
IMD1	IM1 images supported
FS10	IOCA FS10 images supported.

Environment Notes: This option can only be set and queried. It is used to describe the capabilities of a print system component.

INBIN Specifies the source for paper or other media for a print job. Values from this option are used to build an inline form definition when the ***BUILD** keyword is specified for the **FORMDEF** option.

Validation Rule: **PRD_RANGE**

Valid Values: 0 (decimal places), 1 (minimum), 255 (maximum)

Environment Notes: This option is functionally supported by a printer driver in the OS/400, MVS, and VM operating systems. In VM and OS/400, you can specify **INBIN** without specifying ***BUILD** for the **FORMDEF** option. In OS/400, the **INBIN** value overrides any value specified in the active form definition.

INDEX For 3211 printers with the indexing feature only, specifies the number of print positions for the left margin indentation.

Validation Rule: **PRD_RANGE**

Valid Values: 0 (decimal places), 1 (minimum), 31 (maximum)

Environment Notes: This option is functionally supported by a printer driver in the MVS operating system.

JOGOUT Specifies jogging (offsetting) of copy groups within a print job in the printer output bin. For continuous forms printers that do not burst output into cut sheets, specifies that edge marking is to change on copy group boundaries to facilitate output separation. Values from this option are used only to build an inline form definition when the ***BUILD** keyword is specified for the **FORMDEF** option.

Validation Rule: **PRD_LIST**

Valid Values:

YES	Jog output.
NO	Do not jog output.

Environment Notes: This option is functionally supported by a printer driver in the OS/400, MVS, and VM operating systems.

LEFTMAR Specifies the width of the left margin in pels. Values from this option are used only to build an inline form definition when the ***BUILD** keyword is specified for the **FORMDEF** option.

Validation Rule: **PRD_RANGE**

Valid Values: 0 (decimal places), 0 (minimum), 32767 (maximum)

Environment Notes: This option is functionally supported by a printer driver in the OS/400, MVS, and VM operating systems.

LINDEX For 3211 printers with the indexing feature only, specifies the number of print positions for the right margin indentation.

Validation Rule: **PRD_RANGE**

Valid Values: 0 (decimal places), 1 (minimum), 31 (maximum)

Environment Notes: This option is functionally supported by a printer driver in the MVS operating system.

LINECT For a line printer, specifies the maximum number of lines JES2 is to print on each output page.

Validation Rule: **PRD_RANGE**

Valid Values: 0 (decimal places), 0 (minimum), 255 (maximum)

Environment Notes: This option is functionally supported by a printer driver in the MVS operating system.

LPI Specifies the number of printed lines per inch.

Validation Rule: **PRD_RANGE**

Valid values: 0 (decimal places), 1 (minimum), 99 (maximum)

Environment Notes: This option is used by the PRF component of IBM SAA PrintManager to select a page definition for line printing.

MEDIATTRIBUTES Specifies media availability.

Validation Rule: **PRD_LIST**

Valid Values:

HCAPS_CURRENT	Current form or media is installed.

Environment Notes: This option can only be set and queried. It is used to describe the capabilities of a print system component.

MEDIATYPE Specifies the type of media (for example, continuous-forms paper or cut-sheet paper) being used by the printer.

Validation Rule: **PRD_LIST**

Valid Values:

CONTINUOUS	Continuous-forms paper.
SHEET	Cut-sheet paper.

Environment Notes: This option can only be set and queried. It is used to describe the capabilities of a print system component.

MEDIAXLEFTCLIP Specifies the left clip limit of the media in millimeters.

Validation Rule: **PRD_RANGE**

Valid Values: 0 (decimal places), 0 (minimum), 2^{31} (maximum)

Environment Notes: This option can only be set and queried. It is used to describe the capabilities of a print system component.

MEDIAXPELS Specifies the number of pels between the left and right clip limits of the media.

Validation Rule: **PRD_RANGE**

Valid Values: 0 (decimal places), 0 (minimum), 2^{31} (maximum)

Environment Notes: This option can only be set and queried. It is used to describe the capabilities of a print system component.

MEDIAXRIGHTCLIP Specifies the right clip limit of the media in millimeters.

Validation Rule: **PRD_RANGE**

Valid Values: 0 (decimal places), 0 (minimum), 2^{31} (maximum)

Environment Notes: This option can only be set and queried. It is used to describe the capabilities of a print system component.

MEDIAXWIDTH Specifies the width (left to right) of the media in millimeters.

Validation Rule: **PRD_RANGE**

Valid Values: 0 (decimal places), 0 (minimum), 2^{31} (maximum)

Environment Notes: This option can only be set and queried. It is used to describe the capabilities of a print system component.

MEDIAYBOTTOMCLIP

Specifies the bottom clip limit of the media in millimeters.

Validation Rule: **PRD_RANGE**

Valid Values: 0 (decimal places), 0 (minimum), 2^{31} (maximum)

Environment Notes: This option can only be set and queried. It is used to describe the capabilities of a print system component.

MEDIAYHEIGHT Specifies the height (top to bottom) of the media in millimeters.

Validation Rule: **PRD_RANGE**

Valid Values: 0 (decimal places), 0 (minimum), 2^{31} (maximum)

Environment Notes: This option can only be set and queried. It is used to describe the capabilities of a print system component.

MEDIAYPELS Specifies the number of pels between the top and bottom clip limits of the media.

Validation Rule: **PRD_RANGE**

Valid Values: 0 (decimal places), 0 (minimum), 2^{31} (maximum)

Environment Notes: This option can only be set and queried. It is used to describe the capabilities of a print system component.

MEDIAYTOPCLIP Specifies the top clip limit of the media in millimeters.

Validation Rule: **PRD_RANGE**

Valid Values: 0 (decimal places), 0 (minimum), 2^{31} (maximum)

Environment Notes: This option can only be set and queried. It is used to describe the capabilities of a print system component.

MESSAGES For a job sent to an AFP printer, specifies the number of errors to be received before terminating the job (the number of errors received may not equal the number of messages received). Setting a high value for this option helps to debug a document, whereas setting a low value ensures that print job processing will stop if an error is detected.

Validation Rule: **PRD_RANGE**

Valid Values: 0 (decimal places), 0 (minimum), 1000 (maximum). A value of 0 specifies no messages will be received. A value of 1000 specifies that all messages will be received.

Environment Notes: This option is functionally supported by a printer driver in the MVS and VM operating systems.

MODCASET Identifies the level of MO:DCA-P supported by the printer driver. This print option is used to determine if the data stream format is to be converted before placing the print job on the spool. For more information on the MO:DCA-P to AFPDS data stream transform, refer to *Systems Application Architecture Common Programming Interface PrintManager Reference.*

Validation Rule: **PRD_LIST**

Valid Values:

FS9	Specifies the Interchange Set 1 subset of the MO:DCA-P architecture that is defined as SAA print interchange format.
AFPDS	Identifies the subset of the MO:DCA-P architecture that is supported by PSF/MVS and PSF/VM.
AFPDSE1	Identifies the subset of the MO:DCA-P architecture that includes Interchange Set 1 and other extensions, such as, IM Image, bar codes, and page definitions.

Environment Notes: This option is meaningful only for AFPDS data streams, and is used by PrintManager to determine if data stream transforms are required in the MVS and VM environments.

MODIFYNAME Specifies the copy-modification module to be used to print a job on a 3800 printer.

Validation Rule: **PRD_STRING**

Valid Values: 4 (maximum length)

Environment Notes: This option is functionally supported by the MVS and VM operating systems.

MODIFYTRC Specifies the character-arrangement table to be used with a copy-modification module (specified in **MODIFYNAME**) on a 3800 printer.

Validation Rule: **PRD_RANGE**

Valid Values: 0 (decimal places), 0 (minimum), 3 (maximum)

Environment Notes: This option is functionally supported by the MVS and VM operating systems. In VM and MVS, if **MODIFYTRC** is specified, **MODIFYNAME** must be specified and must have a valid value.

OUTDISP Specifies the disposition of the print job.

Validation Rule: **PRD_LIST**

Valid Values:

HOLD	Specifies that the print job will be held until released.
NOHOLD	Specifies that the print job will not be held.
KEEP	Specifies that the print job will be saved after it is printed.
NOKEEP	Specifies that the print job will not be saved after it is printed.

Note: You can specify both HOLD and KEEP, and you can specify NOHOLD and NOKEEP.

Environment Notes: HOLD and NOHOLD are supported in the MVS, VM, and OS/400 environments. KEEP and NOKEEP are supported only in OS/400.

OUTMETHOD Specifies postprocessing operations done by the printer.

Validation Rule: **PRD_LIST**

Valid Values:

BURST	Burst continuous-forms output.

Environment Notes: This option is functionally supported only in the MVS environment.

OUTPUTID Specifies the printer name.

Validation Rule: **PRD_NOVALIDATION**

Valid values: Any character string specifying the printer name in one of the following formats:

FORMAT	
nodeid.outputid	The network nodeid and the destination (for VM) or the userid (for MVS) of the printer. Use this format when the user is on a network node different from the node where the job is to be processed. In VM, if this format is used, the **RSCSID** print option should be used to specify the userid of the local RSCS service machine. Otherwise, the user's virtual printer device (00E) must be spooled to the userid of the local RSCS service machine before running the print application, or the user must

transfer the print job to the userid of the local RSCS service machine after running the print application.

outputid The userid of the printer. Use this format when the user is on the same network node as the node where the job will be processed. In OS/400, this value is also used as the output queue name if it is not specified in the **OUTQ** option.

Environment Notes: This option is functionally supported by the spooling program in the MVS, VM, and OS/400 operating systems, and it correlates to the operating system values described in Table A-1 on page A-24.

OUTQ Specifies the output queue for the spooled output file.

Validation Rule: **PRD_STRING**

Valid Values: 10 (maximum length)

Environment Notes: This option is functionally supported by a printer driver in the OS/400 operating system.

OVERLAY Specifies the electronic overlays for a print job.

Validation Rule: **PRD_NOVALIDATION**

Valid Values: Any character string specifying up to 8 overlays for the print job in one or more of the following formats:

***INLINE name** This keyword is supported in VM, MVS, and OS/400. Use ***INLINE** to select an overlay from a system library (on the system where the print job is submitted) and place it inline with the print job. Refer to your printer-driver documentation for information on how inline resources are handled in each environment. In addition, the overlay must be named in the data stream in order to print. The *name* field is the fully qualified name of the file containing the overlay.

In VM, if the *name* field is not a fully qualified name, the default file type is OVLY38PP and a file mode of "A" is assumed. To search all accessed disks, "*" can be specified as a wild card.

In OS/400, the object type is "*OVL."

***SYSTEM name** This keyword is supported in MVS and VM. Use ***SYSTEM** to select an overlay (specified in *name*) from a system library on the system where the print job is to be printed. In addition, the overlay must be named in the data stream in order to print. The *name*

field is the 1–8 character name of an overlay. If the overlay name is not specified, a system default is used.

For VM only, the *name* field can also include an optional 1–8 character file type. The default file type is OVLY38PP.

Note: If the ***SYSTEM** keyword would be the first keyword of multiple keywords and values, the first ***SYSTEM** keyword can be omitted. For example:

```
01LOGO OVLYXX *SYSTEM 01ABC OVLYXX
```

***FORMDEF name** This keyword is supported in VM, MVS, and OS/400. Use **FORMDEF** to specify that the overlay in the *name* field will be specified in a form definition built with the ***BUILD** keyword. The location of the overlay used is determined by the printer driver. This overlay will be included on every page of the document. The *name* field is the 1–8 character name of the overlay. Values from this option are used only to build an inline form definition when the ***BUILD** keyword is specified for the **FORMDEF** option.

You can specify multiple overlays by entering multiple keywords and names separated by blanks. For example:

```
OVERLAY *SYSTEM 01IBM *SYSTEM 01SYS
```

To specify that an overlay be named in an inline form definition (built with the ***BUILD** keyword of the **FORMDEF** option) and that the overlay resides in a system library, use: **OVERLAY *FORMDEF** *name* ***SYSTEM** *name*, where the name field is the same in both keywords.

***INLINE** can also be paired with ***FORMDEF** to specify that for the overlay is to be named in an inline form definition and that the overlay will be included inline with the print file.

Environment Notes: Refer to the environment information above.

PAGEDEF Specifies the name and location of the page definition used for AFP printing of a line data print job.

Validation Rule: **PRD_NOVALIDATION**

Valid Values: Any character string that specifies the page definitions for the print job in one of the following formats (both keywords are supported in VM, MVS, and OS/400):

***INLINE name** Use ***INLINE** to select a page definition from a system library (on the system where the print job is submitted) and place it inline with the print job. Refer to your printer-driver documentation for

information on how inline resources are handled in each environment. The *name* field is the fully qualified name of the file containing the page definition.

In VM, if the *name* field is not a fully qualified name, the default file type is PDEF38PP and a file mode of "A" is assumed. To search all accessed disks, "*" can be specified as a wild card.

In OS/400, the object type is "*PAGDFN."

***SYSTEM name** Use ***SYSTEM** to select a page definition (specified in *name*) from a system library on the system where the print job is to be printed. The *name* field is the 1–8 character name of a page definition. ***SYSTEM** is also used to name a page definition that is already a part of the print data. The ***SYSTEM** keyword is optional and can be omitted. For example:

```
P1USERXX
```

For VM only, the *name* field can also include an optional 1–8 character file type. The default file type is PDEF38PP.

In OS/400, the object type is "*PAGDFN."

Environment Notes: This option is functionally supported by a printer driver in the MVS, VM, and OS/400 operating systems. Refer to the environment information above.

PAGERANGE Specifies the beginning page and ending page of a print job.

Validation Rule: **PRD_RANGE**

Valid Values: 0 (decimal places), 0 (minimum), 2^{31} (maximum)

Environment Notes: This option is functionally supported only in the OS/400 environment.

PAGESEG Specifies the location of the page segments for a print job. The page segments must be named in the data stream in order to print.

Validation Rule: **PRD_NOVALIDATION**

Valid Values: Any character string specifying the page segments for the job in one or more of the following formats:

***INLINE name** This keyword is supported in VM, MVS, and OS/400. Use ***INLINE** to select a page segment from a system library (on the system where the print job is submitted) and place it print job. Refer to your printer-driver documentation for information on how inline resources

are handled in each environment. The *name* field is the fully qualified name of the file containing the page segment.

In VM, if the *name* field is not a fully qualified name, the default file type is PSEG38PP and a file mode of "A" is assumed. To search all accessed disks, "*" can be specified as a wild card.

In OS/400, the object type is "*PAGSEG."

***SYSTEM name**

This keyword is supported in MVS and VM. Use ***SYSTEM** to select a page segment (specified in *name*) from a system library on the system where the print job is to be printed. The *name* field is the 1—8 character name of a page segment. If the page segment name is not specified, a system default is used.

For VM only, the *name* field can also include an optional 1—8 character file type. The default file type is PSEG38PP.

Note: If the ***SYSTEM** keyword would be the first keyword of multiple keywords and values, the first ***SYSTEM** keyword can be omitted. For example:

```
S1LOGO PSEGXX *SYSTEM S1X91 PSEGXX
```

To specify multiple page segments, enter multiple keywords and names separated by blanks. For example:

```
PAGESEG *SYSTEM S1IBM *INLINE S1LOGO
```

Environment Notes: Refer to the environment information above.

PRMODE

Specifies job processing. You can use **PRMODE** in one of two ways:

- In MVS, to schedule jobs by specifying installation-defined values
- To specify the type of shift-out and shift-in (SOSI) codes in the data stream. SOSI codes are used to change processing between single-byte and double-byte fonts. SOSI1 and SOSI2 are examples of values that specify the type of SOSI processing.

Validation Rule: **PRD_STRING**

Valid Values: 8 (maximum length), for example, SOSI1, SOSI2, and PAGE.

Environment Notes: This option is functionally supported by a printer driver in the MVS, VM, and OS/400 operating systems. In MVS, it is also used as a scheduling parameter. In OS/400, you must be on an IGC machine.

PRTDIRECTION Ensures page-presentation compatibility between the 3800 Printing Subsystem and the IBM 3835 Page Printer. Use this option when building a form definition (with the ***BUILD** keyword) for print jobs that meet one of the following conditions:

- Were formatted for landscape presentation on wide leading edge media for printing on the 3835
- Were formatted for the 3800 but are now printed on the 3835
- Are printed on both printers.

When printing on the 3835, the print direction specified in the page definition (or to a formatter that produces an AFPDS data stream) should correspond to the **PRTDIRECTION** value specified to PrintManager as follows:

- If you specify PORTRAIT or LANDSCAPE for **PRTDIRECTION**, specify the ACROSS print direction in the page definition or to the formatter.
- If you specify PORTRAIT90 or LANDSCAPE90 for **PRTDIRECTION**, specify the DOWN print direction in the page definition or to the formatter.

Note: Output printed in "portrait" page presentation has the shorter edges of the paper at the top and bottom of the page and the longer edges at the sides of the page. Output printed in "landscape" page presentation has the longer edges of the paper at the top and bottom of the page and the shorter edges at the sides of the page.

Validation Rule: **PRD_LIST**

Valid Values:

PORTRAIT	Portrait page presentation for jobs printed on narrow leading edge media.
PORTRAIT90	Portrait page presentation for jobs printed on wide leading edge media.
LANDSCAPE	Landscape page presentation for jobs printed on wide leading edge media.
LANDSCAPE90	Landscape page presentation for jobs printed on narrow leading edge media.

Environment Notes: This option is functionally supported by a printer driver in the OS/400, MVS, and VM operating systems.

PRTENVIRONMENT Defines the operating system environment of the printer.

Validation Rule: **PRD_LIST**

Valid values:

MVS	MVS environment
VM	VM environment
AS/400	AS/400 environment
OS/2	OS/2 environment

Environment Notes: This option can only be set and queried. It is used to describe the capabilities of a print system component.

PRTQUAL Specifies the print-quality level for the print job. Specifying 1 indicates the lowest print quality, and specifying 254 indicates the highest print quality. Actual print-quality levels are unique for each printer; refer to your printer documentation. Values from this option are used only to build an inline form definition when the ***BUILD** keyword is specified for the **FORMDEF** option.

Validation Rule: **PRD_RANGE**

Valid Values: 0 (decimal places), 1 (minimum), 254 (maximum)

Environment Notes: This option is functionally supported by a printer driver in the OS/400, MVS, and VM operating systems.

PRTY

Specifies the print-job priority. The effect of this option is installation dependent.

Validation Rule: **PRD_RANGE**

Valid Values: 0 to 255, where 0 specifies the lowest priority and 255 the highest priority.

Environment Notes: This option is functionally supported in MVS and OS/400.

RSCSID

Specifies the userid of the local RSCS service machine in VM. **RSCSID** is used with the **OUTPUTID** option, and **RSCSID** is required when the user is on a different network node from the VM or MVS node where the job will be processed.

Validation Rule: **PRD_STRING**

Valid Values: 8 (maximum length)

Environment Notes: This option is functionally supported in the VM environment, and it corresponds to the userid specified on the *TO* parameter of the CP SPOOL command. If the *nodeid.outputid* format is specified on the **OUTPUTID** option in VM, **RSCSID** should also be specified. Otherwise, the user's virtual printer device (00E) must be spooled to the userid of the local RSCS service machine before running the print application, or the user must transfer the print job to the userid of the local RSCS service machine after running the print application. This value correlates to the *TO* parameter on the CP SPOOL command. This value indicates the userid under which printed or punched output is produced.

In VM, **RSCSID** and **DOCOWNER** are mutually exclusive. If both are specified, the **DOCOWNER** value will be used. If the print job must be directed to a remote system using **RSCSID**, **DOCOWNER** must not be specified.

SCHEDULE

Specifies when the spooled output file is available to a printer driver.

Validation Rule: **PRD_LIST**

Valid Values:

JOBEND	File is available when processing is completed for the job containing the file.
FILEEND	File is available when processing is completed for the file.
IMMED	File is available immediately.

Environment Notes: This option is functionally supported by a printer driver in the OS/400 operating system.

TECHNOLOGY

Identifies the type of technology of the specified device.

Validation Rule: **PRD_LIST**

Valid Values:

UNKNOWN	Undefined device

	VECTORPLOTTER	Vector plotter
	RASTERDISPLAY	Raster display
	RASTERPRINTER	Raster printer
	IMPACTPRINTER	Impact printer
	POSTSCRIPT	PostScript device

Environment Notes: This option can only be set and queried. It is used to describe the capabilities of a print system component.

TEXTSET

Specifies the MO:DCA-P text capability supported by the printer driver.

Validation Rule: **PRD_LIST**

Valid Values:

NONE	No text support
PT1	PTOCA PT1 text supported
PT2	PTOCA PT2 text supported.

Environment Notes: This option can only be set and queried. It is used to describe the capabilities of a print system component.

TOPMAR

Specifies the width of the top margin in pels. Values from this option are used only to build an inline form definition when the ***BUILD** keyword is specified for the **FORMDEF** option.

Validation Rule: **PRD_RANGE**

Valid Values: 0 (decimal places), 0 (minimum), 32767 (maximum)

Environment Notes: This option is functionally supported by a printer driver in the OS/400, MVS, and VM operating systems.

TRC

Specifies if the print data contains font table reference characters (TRCs). Fonts referenced by TRCs can be named in the **FONT** option. This option is valid if the data stream is line or mixed, the data stream contains TRCs, and the job will be printed on a 3800-1 or on an AFP printer.

Validation Rule: **PRD_LIST**

Valid Values:

YES	Print data contains TRCs.
NO	Print data does not contain TRCs.

Environment Notes: This option is functionally supported by a printer driver in the MVS, VM, and OS/400 operating systems.

UCS

Specifies the following:

- The universal character set image to use in printing the file
- A print train to use in printing the file on an impact printer
- A character-arrangement table for a file printed on an IBM 3800 line printer on a JES2 system.

Validation Rule: **PRD_STRING**

Valid Values: 4 (maximum length)

Environment Notes: This option is functionally supported by a printer driver in the MVS operating system.

VERTRES

Specifies the vertical resolution of the printer in pels per meter.

Validation Rule: **PRD_RANGE**

Valid Values: 0 (decimal places), 1 (minimum), 2^{31} (maximum)

Environment Notes: This option can only be set and queried. It is used to describe the capabilities of a print system component.

WRITER Specifies a spool program other than JES to process the file.

Validation Rule: **PRD_STRING**

Valid Values: 8 (maximum length)

Environment Notes: This option is functionally supported by a printer driver in the MVS operating system.

PrintManager Print Options and Existing Operating System Methods

Table A-1 lists the PrintManager print options and commonly used existing methods of specifying print job attributes in the VM, MVS, and OS/400 operating systems. For more information refer to "Related Documentation" on page 1-4.

Table A-1 (Page 1 of 3). PrintManager Print Options and Existing Operating System Methods

PrintManager Option	MVS	VM	OS/400
CC	RECFM = *xx*A or *xx*M where *xx* can equal FB, VB,F, V, FT, and so forth, in the DD JCL statement.	CC or NOCC options in the PRINT or PSF commands.	CTLCHAR in OVRPRTF command; CC NJE input parameter.
CKPTLINE	CKPTLINE in OUTPUT JCL statement.		
CKPTPAGE	CKPTPAGE in OUTPUT JCL statement.	CKPTPAGE in PSF command.	
CKPTSEC	CKPTSEC in OUTPUT JCL statement.		
CLASS	CLASS in OUTPUT JCL statement or SYSOUT in DD JCL statement.	CLASS in SPOOL PRT command.	
COPIES	COPIES in OUTPUT and DD JCL statements.	COPY in SPOOL PRT command.	COPIES in CHGSPLFA command.
DATACK	DATACK in OUTPUT JCL statement.	DATACK in PSF command.	DATACK NJE input parameter.
DATATYPE			DEVTYPE in OVRPRTF command; PRMODE NJE input parameter.
DOCOWNER		SPOOL 00E FOR *userid*	
DUPLEX	Specified in form definition.	DUPLEX in PSF command	DUPLEX in OVRPRTF command; DUPLEX NJE input parameter.
FCB	FCB in DD JCL statement or PAGEDEF in OUTPUT JCL statement.	FCB in SPOOL PRT command.	

Table A-1 (Page 2 of 3). PrintManager Print Options and Existing Operating System Methods

PrintManager Option	MVS	VM	OS/400
FIDELITY			FIDELITY in CHGSPLFA command.
FLASHCNT	FLASH in OUTPUT or DD JCL statements.	FLASH in SPOOL PRT command.	
FLASHNAME	FLASH in OUTPUT or DD JCL statements.	FLASH in SPOOL PRT command.	
FORM	FORMS in OUTPUT JCL statement or SYSOUT in DD JCL statement.	FORM in SPOOL PRT command.	FORMTYPE in CHGSPLFA command.
FORMDEF	FORMDEF in OUTPUT JCL statement.	FORMDEF in PSF command.	
INBIN	Specified in form definition.	BIN in the PSF command	DRAWER in OVRPRTF command; BIN NJE input parameter.
INDEX	Supported in OUTPUT JCL statement (3211 printer only).		
LINDEX	Supported in OUTPUT JCL statement (3211 printer only).		
LINECT	LINECT in OUTPUT JCL statement (line printer only).		
MESSAGES	PIMSG in OUTPUT JCL statement.	MESSAGE RETURN in PSF command.	
MODIFYNAME	MODIFY under JES only.	MODIFY in SPOOL PRT command.	
MODIFYTRC	MODIFY under JES only.	MODIFY in SPOOL PRT command.	
OUTDISP	HOLD option in DD JCL statement.	HOLD or NOHOLD options in SPOOL PRT command.	HOLD and SAVE in OVRPRTF command.
OUTMETHOD = BURST	BURST = Y,N in OUTPUT or DD JCL statements.		

Table A-1 (Page 3 of 3). PrintManager Print Options and Existing Operating System Methods

PrintManager Option	MVS	VM	OS/400
OUTPUTID = *nodeid.outputid*	DEST = *nodeid.outputid* in OUTPUT JCL statement.	TAG DEV 00E *nodeid* and SPOOL 00E DEST *outputid*.	
OUTPUTID = *outputid*	DEST = *outputid* in OUTPUT JCL statement.	SPOOL 00E DEST *outputid*.	DEV in CHGSPLFA command.
OUTQ			OUTQ in CHGSPLFA command.
PAGEDEF	PAGEDEF in OUTPUT JCL statement.	PAGEDEF in PSF command.	
PAGERANGE			PAGERANGE in CHGSPLFA command.
PRMODE	PRMODE in OUTPUT JCL statement.	PRMODE in PSF command.	IGCDTA and IGCSOSI in OVRPRTF command.
PAGESEG		PAGESEG in PSF command.	
PRTY	PRTY in OUTPUT JCL statement under JES only.		OUTPTY in CHGSPLFA command.
RSCSID		SPOOL 00E TO *rscs userid*.	
SCHEDULE			SCHEDULE in CHGSPLFA command.
TRC	TRC in OUTPUT JCL statement or DCB in DD JCL statement.	TRC or NOTRC options in the PRINT or PSF commands.	TRC NJE input parameter.
UCS	UCS in OUTPUT or DD JCL statements.		
WRITER	Supported in OUTPUT JCL statement under JES only.		

Appendix B. Example of a PRD TOOL Input File

```
/*-------------------------------------------------------------------*/
/*              Example PRD TOOL Input File                          */
/*                                                                   */
/* This example tag file does the following:                         */
/*     - Creates a print descriptor that contains two options        */
/*       and two references.                                         */
/*     - Deletes one option and one reference.                       */
/*     - Saves the print descriptor.                                 */
/*     - Deletes the print descriptor.                               */
/*                                                                   */
/*-------------------------------------------------------------------*/

:PRD.                              /* start PRD session          */

:SETOPT.Option1                    /* option # 1                 */
   :RULE.PRD_STRING
   :VVALUES.30
   :DEFAULT.Default for PRD_STRING rule
:ESETOPT.

:SETOPT.Option2                    /* option # 2                 */
   :RULE.PRD_LIST
   :VVALUES.FRED BILL LARRY
   :DEFAULT.FRED
:ESETOPT.

:DELOPTION.Option2                 /* Delete option #            */


:REFLIST.                          /* start reference list.      */

   :REFERENCE.1                    /* set reference 1            */
      :PRDNAME.GRPALIAS=USER PRDNAME=PRD SAMPLE 2
      :PROCESSFLAG.PRD_BUILD
   :EREFERENCE.

   :REFERENCE.2                    /* set reference 2            */
      :PRDNAME.GRPALIAS=USER PRDNAME=PRD SAMPLE 3
      :PROCESSFLAG.PRD_BUILD
   :EREFERENCE.

:EREFLIST.

                                   /* start reference list.      */
:REFLIST.

   :DELREF.2                       /* delete reference 2         */

:EREFLIST.
```

Figure B-1 (Part 1 of 2). Example of a PRD TOOL Input File

```
                                    /* save the print descriptor.     */
:SAVE.GRPEXACT=GROUP1 SAMPLE PRDNAME=PRD SAMPLE 1
   :DESCRIPTION.This is a print descriptor with valid syntax
   :PRDID.PRD_PRESENTATION_DEVICE
   :CONTROLPRD.PRD_CREATE
   :CONTROLGRP.PRD_AUTO_CREATE
:ESAVE.

:EPRD.                               /* end PRD session             */


:DESTROY.GRPEXACT=GROUP1 SAMPLE PRDNAME=PRD SAMPLE 1
:CONTROLGRP.PRD_AUTO_DELETE
:EDESTROY.
```

Figure B-1 (Part 2 of 2). Example of a PRD TOOL Input File

Appendix C. PrintManager API Verb C Language Applications

This appendix provides the following information about PrintManager C applications:

- C language coding conventions
- Creating and running PrintManager C applications in System/370
- Compiling and running API C applications in OS/400
- API data types for the C language
- API C language verb calls.

C Language Coding Conventions

Verb Calls

In C language, you can use the PrintManager verb long names or short names when coding an application. For example, when calling the Get Descriptor verb, you can use PDGETD or PrdGetDescriptor. The coding example shown in Appendix D, "Example of a C Language Print-Descriptor Edit Session" on page D-1 uses the long names for the verbs.

Header Files

The PrintManager header files are shipped with the IBM C/400 compiler (for OS/400) or with PrintManager (for VM and MVS). The application source program must have two define statements (for the API verbs and the API error codes) followed by an include statement for the system header file. These statements must be coded before any calls to API verbs. These statements are as follows:

```
#define  INCL_PRD
#define  INCL_ERRORS
#include  <ekipmgr.h>
```

Note: If your print descriptor application makes calls to the PrintManager Interface verbs, you must also code:

```
#define  INCL_PRTMGR
```

APIENTRY

All functions are implicitly of type APIENTRY. The entry is defined in the system header file EKIPMGR.H as:

```
#define APIENTRY
```

Buffers

In C language, when a buffer contains a list of names (for example, with PDLOPT (List Print Options)), this is implemented as an array of pointers that refer to the list in the buffer. Otherwise, the buffer contains an array of structures or data.

When allocating storage for a structure (except for the SDF structure), the *sizeof* C function should be used to determine the size of the structure because the C compiler may add additional bytes for structure alignment or padding between fields of a structure.

Creating and Running IBM SAA PrintManager C Applications in the System/370 Environment

In the System/370 environment, IBM SAA PrintManager provides dynamic linking of IBM SAA PrintManager C language applications to the IBM SAA PrintManager program code. Dynamic linking allows changes to be made to the IBM SAA PrintManager licensed program without requiring you to modify or relink your IBM SAA PrintManager API and PrintManager Interface applications. Furthermore, dynamic linking provides for smaller application executable modules because IBM SAA PrintManager applications will only link to verb "stub" libraries at link-edit time. Using stub libraries, however, requires that the IBM SAA PrintManager execution time libraries are available when running a IBM SAA PrintManager application.

Note: IBM SAA PrintManager in MVS/ESA, VM/XA, or VM/ESA systems only supports IBM SAA PrintManager applications operating in 31 bit addressing mode. Applications in these environments must specify the AMODE 31 option at program link-edit or load time. On VM/SP Release 6 systems, specifying AMODE 31 is recommended but not required.

For general information on how to create and run C programs in the System/370 environment, refer to *IBM C/370 User's Guide*, SC09-1264. The following sections describe IBM SAA PrintManager-specific information for compiling, link-editing, and running C IBM SAA PrintManager applications in a System/370 environment.

Creating and Running MVS IBM SAA PrintManager C Applications

The following sections tell how to compile, link-edit, and run C applications in the MVS environment.

Note: The library names shown in the examples in the following sections are examples only. For the actual names of these libraries, refer to your program directory.

Compiling MVS Applications

As described in "Header Files" on page C-1, any application source file making calls to IBM SAA PrintManager verbs must include the IBM SAA PrintManager header files. These header files are installed in a IBM SAA PrintManager header file library 'EKI.V*v*R*r*M*m*.EKIHSRC'. This library must be concatenated to the SYSLIB DD statement of your compile job, along with the C/370 product system header file library 'EDC.V*v*R*r*M*m*.SEDCHDRS'.

Note: For both 'EKI.V*v*R*r*M*m*.EKIHSRC' and 'EDC.V*v*R*r*M*m*.SEDCHDRS', replace *v*, *r*, and *m* respectively with the version, release, and modification of IBM SAA PrintManager and C/370 that you have installed.

Link-Editing MVS Applications

After you successfully compile your application modules, they must be link-edited, along with the IBM SAA PrintManager and C/PLI stub libraries. Figure C-1 on page C-3 shows an example of an MVS batch JCL job for link-editing IBM SAA PrintManager applications.

```
//JOBname     JOB    acctno,name,...
//stepname    EXEC   PGM=IEWL,PARM='AMODE=31,RMODE=ANY,..'
//SYSPRINT    DD     SYSOUT=A
//SYSLMOD     DD     DSNAME=datasetname(member),UNIT=SYSDA,
                     DISP=(NEW,PASS),SPACE=(subparams)
//SYSLIB      DD     DSNAME=EDC.V1R2M0.SEDCBASE,DISP=SHR
                     DSNAME=PLI.V2R2M1.SIBMBASE,DISP=SHR
                     DSNAME=EKI.V1R1M0.EKIPMLIB,DISP=SHR
//SYSLIN      DD     DSNAME=datasetname(member),DISP=SHR
//SYSUT1      DD     UNIT=SYSDA,SPACE=(subparams)
/*
```

Figure C-1. Example MVS Batch Job for Link-Editing IBM SAA PrintManager Applications

Running MVS Applications

Before running an MVS IBM SAA PrintManager application, you must make the IBM SAA PrintManager, C, and PLI execution time libraries accessible to the application. You can do this by either:

- Installing the IBM SAA PrintManager execution time library in the Link Pack Area (LPA). In this case, do not include the execution time library in the STEPLIB DD statement.
- Explicitly concatenating the libraries to the STEPLIB DD statement at the program execution step. In this case, the libraries are *not* installed in the LPA.

Figure C-2 shows an example of running an MVS application with the library concatenation at the program execution step because the IBM SAA PrintManager, C, and PLI libraries are not installed in the LPA.

```
//JOBname     JOB    acctno,name,...
//STEP1       EXEC   PGM=progname,PARMS='parms'
//STEPLIB     DD     DSN=EDC.V1R2M0.SEDCLINK,DISP=SHR
//            DD     DSN=PLI.V2R2M1.SIBMLINK,DISP=SHR
//            DD     DSN=EKI.V1R1M0.EKIRUNLB,DISP=SHR
//EKIGLNME    DD     DSN=EKI.$GLDS$.EKIPM,DISP=SHR
/*
```

Figure C-2. Example of Running an MVS IBM SAA PrintManager Application

If the IBM SAA PrintManager execution time library cannot be found, the SPRINIT (Initialize PrintManager) verb will return a null *hab* parameter, and dynamic linking will not occur.

Creating and Running VM IBM SAA PrintManager C Applications

The following sections tell how to compile, link-edit, and run C applications in the VM environment.

Compiling VM Applications

As described in "Header Files" on page C-1, ensure that the IBM SAA PrintManager header files are accessible to the C/370 compiler on your VM system.

Link-Editing VM Applications

When a IBM SAA PrintManager application is link-edited or loaded, the IBM SAA PrintManager stub library is a txtlib that must be included in the txtlib concatenation at link-edit or load time, along with the C/PLI stub library. Figure C-3 shows an example of the CMS command sequence you can use to create an executable module **pmappl** from two C program files **appmain** and **appsub**.

```
GLOBAL TXTLIB EDCBASE IBMLIB CMSLIB EKIPMLIB
LOAD appmain appsub (RESET CEESTART
GENMOD pmappl (FROM CEESTART
```

Figure C-3. Example CMS Command Sequence for Link-Editing IBM SAA PrintManager Applications

The CMS LKED command may also be used to create executable IBM SAA PrintManager applications in a CMS LOADLIB. Refer to the publications listed in "For VM" on page 1-5 for more information on CMS commands.

Running VM Applications

Before running a VM IBM SAA PrintManager application, you must make the IBM SAA PrintManager and C/PLI common execution time libraries accessible to the application. If the IBM SAA PrintManager execution time library is not installed as a VM Discontiguous Shared Segment (DCSS), do one of the following:

- Load the IBM SAA PrintManager execution time library into your own nucleus extension space, using the following CMS commands:

  ```
  FILEDEF ekild DISK EKIRUNLB LOADLIB *
  NUCXLOAD EKIDISPT EKIDISPT ekild (SYSTEM
  ```

- Add the IBM SAA PrintManager execution time load library EKIRUNLB into your global load library list using the CMS GLOBAL LOADLIB command.

If the IBM SAA PrintManager execution time library cannot be found or loaded, the SPRINIT (Initialize PrintManager) verb will return a null *hab* parameter, and dynamic linking will not occur.

Note: On VM/SP Release 5, if the execution time library cannot be found or loaded, a CMS abend will occur, which indicates that entry point EKIDISPT is undefined or unresolved.

Invalid Handles

In the System/370 environment, when you pass an invalid handle (in the *hab*, *hprm*, or *hprd* parameters) to a IBM SAA PrintManager verb, your application program will abend with abend code X'0245'. A program dump is initiated unless the following C pragma preprocessor directive is coded in your main C routine:

```
#pragma runopts(nospie, nostae)
```

This pragma shown above disables the C/PLI abend handler. Refer to *IBM C/370 User's Guide* for more information on the C pragma directive.

Compiling and Running PrintManager C Applications in the OS/400 Environment

Compiling and running PrintManager C applications in the OS/400 Environment requires the following:

1. Use the PrintManager header files as described in "Header Files" on page C-1.
2. After compiling your application, the EXTPGMINF command may be required to create a library information file for your application.
3. Before executing your application, the SETPGMINF command must be executed to set program environment information. The library information file which must be included in this command is QACJINFO located in the QSYS library. The following is an example of how to use the SETPGMINF command for a user's program named MYPGM in the library named MYLIB:

   ```
   SETPGMINF ROOTPGM(MYLIB/MYPGM) LIBFILE((QSYS/QACJINFO) (*CLIB))
   ```

   For more information about the EXTPGMINF and SETPGMINF commands, refer to *Application System/400 Languages: C/400 User's Guide*, SC09-1303.

Data Types for C Language

This section describes the data types for C language, which are similar to the general data types described in "API Verb Data Types" on page 7-2. Each description of a C data type, therefore, provides data characteristics or mappings that are unique to the C language.

Data Type	Description
BOOL	In C, TRUE is defined as a nonzero value, and FALSE is defined as 0. Do not use the constant TRUE in an equality comparison with another BOOL data type, because this may produce incorrect results. Takes the format: `typedef unsigned short BOOL;`
BYTE	A byte of data in the following format: `typedef unsigned char BYTE;`
CHAR	A single-byte character in the following format: `typedef unsigned char CHAR;`

ERRINFO	The data structure used with SPRGERI (Get Error Information) to return error information in the following format:
	`typedef struct _ERRINFO`
	`{`
	`USHORT cbFixedErrInfo;`
	`ERRORID idError;`
	`USHORT cDetailLevel;`
	`USHORT offaoffszMsg;`
	`USHORT offBinaryData;`
	`} ERRINFO;`

ERRMSG	The data structure used with SPRGEEM (Get Error Message) to return additional error information in the following format:

```
typedef struct _ERRMSG
   {
   ULONG  idIdentifier;
   CHAR   szValue[256];
   } ERRMSG;
```

ERRORID A four byte value in which the first two bytes are the severity and the last two bytes are the error code in the following format:

```
typedef ULONG ERRORID;
```

HAB Handle to a PrintManager anchor block in the following format:

```
typedef LHANDLE HAB;
```

HPRD A print-descriptor edit session identifier in the following format:

```
typedef LHANDLE HPRD;
```

LHANDLE Pointer for the handle for PrintManager verbs in the following format:

```
typedef void       *LHANDLE;
```

OPTDEFN1 A print-option definition structure in the following format:

```
struct optdefn1
   {
   PSZ    pszName;                /* name of the option          */
   ULONG  flDefType;              /* type of option default      */
   ULONG  cbDefLength;            /* len in bytes of option def */
   PBYTE  pbDef;                  /* ptr to option default       */
   } OPTDEFN1;
```

OPTDEFN2 A print-option definition structure in the following format:

```
struct optdefn2
   {
   PSZ    pszName;                /* name of the option          */
   ULONG  flDefType;              /* type of option default      */
   ULONG  cbDefLength;            /* len in bytes of option def */
   PBYTE  pbDef;                  /* ptr to option default       */
   ULONG  flRule;                 /* print opt validation rule   */
   PSZ    pszVv;                  /* print option valid values   */
   } OPTDEFN2;
```

PBYTE Pointer to a byte of data in the following format:

```
typedef BYTE   * PBYTE;
```

PERRINFO Pointer to an ERRINFO structure in the following format:

```
typedef ERRINFO * PERRINFO;
```

PERRMSG Pointer to an ERRMSG structure in the following format:

```
typedef ERRMSG * PERRMSG;
```

PRDDEFN2 A print-descriptor definition structure in the following format:

```
struct PRDDEFN2
   {
   PSZ             pszPrdName;    /* Prd name                    */
   PRDDESCRIPTION  szDescription; /* description                 */
   ULONG           idPrdId;       /* id (type) of this Prd       */
   PRDLEVELSTAMP   chLevelStamp;  /* date/time/level stamp       */
   } PRDDEFN2;
```

PRDDESCRIPTION Description of the print descriptor in the following format:

```
typedef  char    PRDDESCRIPTION
[PRD_DESC_LENGTH+1];
```

PRDGRP1 A print-descriptor group structure in the following format:

```
struct prdgrp1
   {
   ULONG           idSeqNum;       /* identifies this group      */
   ULONG           flProcessFlag;  /* controls group processing  */
   PSZ             pszGrpAlias;    /* alias name for the group   */
   PSZ             pszGrpName;     /* system-specific group name */
   PRDDESCRIPTION  szDescription;  /* text desc of Prd Group     */
   } PRDGRP1;
```

PRDLEVELSTAMP The date and time a print descriptor was created or last updated in the following format:

```
typedef  char     PRDLEVELSTAMP
[PRD_STAMP_LENGTH+1];
```

PRDREF1 A print-descriptor reference structure in the following format:

```
struct prdref1
   {
   ULONG   idSeqNum;
   ULONG   flProcessFlag;
   PSZ     pszRefName;
   } PRDREF1;
```

PSDF Pointer to an SDF structure in the following format:

```
typedef SDF  *PSDF;
```

PSZ Pointer to a zero-terminated string in the following format:

```
typedef char  *PSZ;
```

PSZARRAY Pointer to an array of zero terminated strings in the following format:

```
typedef PSZ  *PSZARRAY
```

PULONG Pointer to a ULONG data type in the following format:

```
typedef ULONG  *PULONG;
```

SDF A self-defining field (structure) in the following format:

```
typedef struct _SDF {
   ULONG  cbLength;
   ULONG  flType;
   BYTE    bData[1];
   } SDF;
```

ULONG An unsigned 4-byte integer value (32 bits) in the following format:

```
typedef unsigned long ULONG;
```

USHORT An unsigned 2-byte integer value (16 bits) in the following format:

```
typedef unsigned short USHORT;
```

PDBLDD (Build Descriptor)

```
extern BOOL APIENTRY PDBLDD (HPRD hprd);
```

PDCLS (Close Session)

```
extern BOOL APIENTRY PDCLS (HPRD hprd);
```

PDEXPD (Export Descriptor)

```
extern BOOL APIENTRY PDEXPD (HAB   hab,
                             PSZ   pszPrdName,
                             PSZ   pszExportName,
                             ULONG flControlPrd);
```

PDGETD (Get Descriptor)

```
extern BOOL APIENTRY PDGETD (HPRD   hprd,
                             PSZ    pszPrdName,
                             ULONG  ulLevel,
                             ULONG  cbLength,
                             PBYTE  pbBuffer,
                             PULONG pulLengthNeeded,
                             ULONG  flControl);
```

PDIMPD (Import Descriptor)

```
extern BOOL APIENTRY PDIMPD (HAB   hab,
                             PSZ   pszImportName,
                             PSZ   pszPrdName,
                             ULONG flControlPrd,
                             ULONG flControlGrp);
```

PDLDEF (List Descriptors)

```
extern BOOL APIENTRY PDLDEF (HAB    hab,
                             PSZ    pszPrdGroupName,
                             ULONG  ulLevel,
                             ULONG  cbLength,
                             PBYTE  pbBuffer,
                             PULONG pulLengthNeeded,
                             PULONG pulItemsReturned,
                             PULONG pulItemsRemaining,
                             PSDF   psdfContinue);
```

PDLGRP (Query Group List)

```
extern BOOL APIENTRY PDLGRP (HPRD   hprd,
                             ULONG  ulLevel,
                             ULONG  cbLength,
                             PBYTE  pbBuffer,
                             PULONG pulLengthNeeded,
                             PULONG pulItemsReturned);
```

PDLOPT (List Print Options)

```
extern BOOL APIENTRY PDLOPT (HPRD   hprd,
                             ULONG  ulLevel,
                             ULONG  cbLength,
                             PBYTE  pbBuffer,
                             PULONG pulLengthNeeded,
                             PULONG pulItemsReturned,
                             PULONG pulItemsRemaining,
                             PSDF   psdfContinue);
```

PDLREF (Query Reference List)

```
extern BOOL APIENTRY PDLREF (HPRD   hprd,
                             ULONG  ulLevel,
                             ULONG  cbLength,
                             PBYTE  pbBuffer,
                             PULONG pulLengthNeeded,
                             PULONG pulItemsReturned);
```

PDOPEN (Open Session)

```
extern HPRD APIENTRY PDOPEN (HAB hab);
```

PDQOPT (Query Print Option)

```
extern BOOL APIENTRY PDQOPT (HPRD   hprd,
                             ULONG  ulLevel,
                             ULONG  cbLength,
                             PBYTE  pbBuffer,
                             PULONG pulLengthNeeded);
```

PDRDEF (Delete Descriptor)

```
extern BOOL APIENTRY PDRDEF (HAB   hab,
                                   PSZ   pszPrdName,
                                   ULONG flControlGrp);
```

PDREMO (Delete Print Option)

```
extern BOOL APIENTRY PDREMO (HPRD hprd,
                                PSZ  pszOptionName);
```

PDSAVD (Save Descriptor)

```
extern BOOL APIENTRY PDSAVD (HPRD  hprd,
                                ULONG ulLevel,
                                ULONG cbLength,
                                PBYTE pbBuffer,
                                ULONG flControlPrd,
                                ULONG flControlGrp);
```

PDSGRP (Set Group List)

```
extern BOOL APIENTRY PDSGRP (HPRD  hprd,
                             ULONG ulLevel,
                             ULONG cbLength,
                             PBYTE pbBuffer,
                             ULONG ulTotalItems);
```

PDSOPT (Set Print Option)

```
extern BOOL APIENTRY PDSOPT (HPRD  hprd,
                             ULONG ulLevel,
                             ULONG cbLength,
                             PBYTE pbBuffer);
```

PDSREF (Set Reference List)

```
extern BOOL APIENTRY PDSREF (HPRD  hprd,
                              ULONG ulLevel,
                              ULONG cbLength,
                              PBYTE pbBuffer,
                              ULONG ulTotalItems);
```

SPRFERI (Free Error Information)

```
extern BOOL APIENTRY SPRFERI (PERRINFO pErrorInfo);
```

SPRGEEM (Get Error Message)

```
extern BOOL APIENTRY SPRGEEM (PERRINFO pErrorInfo,
                                    ULONG    ulIndex,
                                    PERRMSG  pErrorMsg,
                                    PULONG   pulTotalCount);
```

SPRGERI (Get Error Information)

```
extern PERRINFO APIENTRY SPRGERI (HAB hab);
```

SPRINIT (Initialize PrintManager)

```
extern HAB APIENTRY SPRINIT (USHORT fsOptions);
```

SPRTERM (Terminate PrintManager)

```
extern BOOL APIENTRY SPRTERM (HAB hab);
```

Appendix D. Example of a C Language Print-Descriptor Edit Session

```
/*********************************************************************/
/*               PrintManager Example C Language Edit Session        */
/*                                                                   */
/*  SYNOPSIS: Example of basic API operations                        */
/*                                                                   */
/*  Sequence of Operations:                                          */
/*                                                                   */
/*  - Start a Print Manager process              (PrtMgrInitialize) */
/*  - Open a Print-Descriptor session                    (PrdOpen) */
/*  - Set options                                   (PrdSetOption) */
/*  - List Options                                 (PrdListOptions) */
/*  - Query Options                                (PrdQueryOption) */
/*  - Set references                                (PrdSetRefList) */
/*  - Save print descriptor                      (PrdSaveDescriptor) */
/*  - Close Print-Descriptor session                     (PrdClose) */
/*  - Free error storage                       (PrtMgrFreeErrorInfo) */
/*  - Terminate Print Manager process             (PrtMgrTerminate) */
/*                                                                   */
/*                                                                   */
/*  Display error information if error occurs:                       */
/*                                                                   */
/*  - Get error information                     (PrtMgrGetErrorInfo) */
/*  - Get extended error messages              (PrtMgrGetExtErrorMsg) */
/*                                                                   */
/*                                                                   */
/* NOTE: This example must be able to access a Group List Print      */
/*       Descriptor to run without errors.                           */
/*********************************************************************/
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

#define  INCL_PRD                    /* print descriptor declares   */
#define  INCL_ERRORS                 /* error code definitions      */
#include <ekipmgr.h>

static ERRINFO      errInfo;         /* ERRINFO buffer.             */
static PERRINFO     pErrInfo = NULL; /* pointer to ERRINFO buffer.  */

static void extErrMsgs(HAB hab);
```

Figure D-1 (Part 1 of 12). Example of a C Language Print-Descriptor Edit Session

```
/*======================== Main Program Starts ======================*/
int main(void)

{                                       /* GENERAL DECLARATIONS ...    */
   BOOL           rc;                   /* return code                 */
   HPRD           hprd;                 /* handle for Prd session      */
   HAB            hab;                  /* anchor block handle         */
   USHORT         i;                    /* index for loop              */
   ULONG          level;                /* level of data in buffer     */
   ULONG          length;               /* length of data in buffer    */
   PBYTE          buffer;               /* ptr to buffer of data       */
   PBYTE          listbuffer = NULL;    /* ptr to list options buffer  */
   PBYTE          qrybuffer  = NULL;    /* ptr to query options buffer */
   ULONG          lengthNeeded;         /* length needed for buffer    */
   ULONG          itemsReturned;        /* # items returned in buffer  */
   ULONG          itemsRemaining;       /* # items remaining in buffer */
   ULONG          totalItems;           /* total items in array        */
   ULONG          controlPrd;           /* Prd control parameter       */
   ULONG          controlGrp;           /* Prd group control parameter */

                                        /* OPTION DECLARATIONS ...     */
   OPTDEFN2       optDefn2;             /* OPTDEFN2 type structure     */
   OPTDEFN2       * opt_data;           /* ptr. to optDefn2 structure  */
   CHAR           option[32];           /* option name                 */
   CHAR           def[80];              /* default option value        */
   ULONG          rule;                 /* rule type for valid values  */
   CHAR           vv[255];              /* option valid values         */
   PSZ            *opt_names;           /* address of option names buff*/

                                        /* REFERENCE DECLARATIONS ...  */
   PRDREF1        references[3];        /* array of PRDREF1 types      */
   CHAR           refName[3][80];       /* array of Prd reference names*/

                                        /* SAVE Prd DECLARATIONS ...   */
   PRDDEFN2       prdDefn2;             /* PRDDEFN2 type structure     */
   CHAR           prdName[80];          /* Prd name                    */

   /*====================================================================*/
   /* Start a Print Manager Process                (PrtMgrInitialize) */
   /* Open a Print-Descriptor session                        (PrdOpen) */
   /*====================================================================*/

   hab = PrtMgrInitialize(0);
   if (hab == NULL)
      {
      printf("\nERROR issued by PrtMgrInitialize");
      return(0);
      }

   hprd = PrdOpen(hab);
   if (hprd == NULL)
      {
      printf("\nERROR issued by PrdOpen");
      extErrMsgs(hab);
      goto Endit;
      }
```

Figure D-1 (Part 2 of 12). Example of a C Language Print-Descriptor Edit Session

```
/*================================================================*/
/* Set options                                      (PrdSetOption) */
/*================================================================*/

/*----------------------------------------------------------------*/
/* Define buffer level using an OPTDEFN2 structure type & define  */
/* constant values used in optDefn2 structure for setting options. */
/*----------------------------------------------------------------*/

level = 2;                          /* use OPTDEFN2 structure type */
length = (LONG)sizeof(OPTDEFN2);    /* length of optdata.          */
buffer = (PBYTE) &optDefn2;         /* pointer to optDefn2 buffer. */

optDefn2.flDefType = PRTMGR_STRL;   /* default is string format.   */
optDefn2.cbDefLength = 0;           /* not used for string format. */

/*----------------------------------------------------------------*/
/* Set COPIES option ...                                          */
/*----------------------------------------------------------------*/

strcpy(option,"COPIES");            /* option name = COPIES.       */
strcpy(def, "1");                   /* default = 1 copy.           */
rule = PRD_RANGE;                   /* rule is of RANGE type.      */
strcpy(vv, "0 1 99");               /* valid values for COPIES,    */
                                    /*   precision = 0,            */
                                    /*   minimum = 1,              */
                                    /*   maximum = 99.             */

optDefn2.pszName = option;
optDefn2.pbDef = def;
optDefn2.flRule = rule;
optDefn2.pszVv = vv;

rc = PrdSetOption(hprd,
                  level,
                  length,
                  buffer);

if (rc==0)
   {
   printf("ERROR issued by PrdSetOption.");
   extErrMsgs(hab);
   goto Endit;
   }
```

Figure D-1 (Part 3 of 12). Example of a C Language Print-Descriptor Edit Session

```
/*-------------------------------------------------------------------*/
/* Set OUTPUTID option ...                                           */
/*-------------------------------------------------------------------*/

strcpy(option, "OUTPUTID");          /* option name = OUTPUTID.      */
strcpy(def, "SYSTEMS");              /* default = SYSTEMS printer.   */
rule = PRD_LIST;                     /* rule is of LIST type.        */
strcpy(vv, "SYSTEMS DEPT");          /* valid values for OUTPUTID.   */
                                     /*   SYSTEMS = systems printer.*/
                                     /*   DEPT = department printer.*/

optDefn2.pszName = option;
optDefn2.pbDef = def;
optDefn2.flRule = rule;
optDefn2.pszVv = vv;

rc = PrdSetOption(hprd,
                  level,
                  length,
                  buffer);

if (rc==0)
   {
   printf("ERROR issued by PrdSetOption.");
   extErrMsgs(hab);
   goto Endit;
   }
/*-------------------------------------------------------------------*/
/* Set DATATYPE option ...                                           */
/*-------------------------------------------------------------------*/

strcpy(option, "DATATYPE");          /* option name = DATATYPE.      */
strcpy(def,"AFPDS");                 /* default = AFPDS.             */
rule = PRD_LIST;                     /* rule is of LIST type.        */
strcpy(vv, "AFPDS LINE");            /* valid values for DATATYPE.   */

optDefn2.pszName = option;
optDefn2.pbDef = def;
optDefn2.flRule = rule;
optDefn2.pszVv = vv;

rc = PrdSetOption(hprd,
                  level,
                  length,
                  buffer);

if (rc==0)
   {
   printf("ERROR issued by PrdSetOption.");
   extErrMsgs(hab);
   goto Endit;
   }
```

Figure D-1 (Part 4 of 12). Example of a C Language Print-Descriptor Edit Session

```
/*================================================================*/
/* List Options                                  (PrdListOptions) */
/*================================================================*/

/*----------------------------------------------------------------*/
/* Invoke first time to get the size of the listbuffer needed.    */
/*----------------------------------------------------------------*/

level = 1;                          /* returns array of STRL type.  */
length = 0;                         /* set to zero and NULL to get  */
listbuffer = NULL;                  /* the actual buffer length.    */

rc = PrdListOptions(hprd,
                    level,
                    length,
                    listbuffer,
                    &lengthNeeded,
                    &itemsReturned,
                    &itemsRemaining,
                    NULL);

if (rc==0)
   {
   printf("ERROR issued by PrdListOptions");
   extErrMsgs(hab);
   goto Endit;
   }
```

Figure D-1 (Part 5 of 12). Example of a C Language Print-Descriptor Edit Session

```
/*-------------------------------------------------------------*/
/* Allocate space needed and invoke again to get option names. */
/*-------------------------------------------------------------*/
if ( (listbuffer =(PBYTE) malloc( (USHORT) lengthNeeded ) ) == NULL)
   {
   printf("\nERROR memory allocation failure");
   goto Endit;
   }

length = lengthNeeded;            /* length of the listbuffer      */

rc = PrdListOptions(hprd,
                    level,
                    length,
                    listbuffer,
                    &lengthNeeded,
                    &itemsReturned,
                    &itemsRemaining,
                    NULL);

if (rc==0)
   {
   printf("ERROR issued by PrdListOptions");
   extErrMsgs(hab);
   goto Endit;
   }

opt_names = (PSZ *)listbuffer; /* set up for array of option names */
```

Figure D-1 (Part 6 of 12). Example of a C Language Print-Descriptor Edit Session

```
/*====================================================================*/
/* Query Options                                    (PrdQueryOption) */
/*====================================================================*/

/*--------------------------------------------------------------------*/
/* Invoke first time to get the maximum size of buffer needed         */
/* for the largest option in the print descriptor.                    */
/*--------------------------------------------------------------------*/

level = 2;                          /* using optDefn2 structure.   */
length = 0;                         /* set to get max. length.     */
qrybuffer = NULL;

rc = PrdQueryOption(hprd,
                    level,
                    length,
                    qrybuffer,
                    &lengthNeeded);

if (rc==0)
   {
   printf("ERROR issued by first PrdQueryOption");
   extErrMsgs(hab);
   goto Endit;
   }
```

Figure D-1 (Part 7 of 12). Example of a C Language Print-Descriptor Edit Session

```
/*-------------------------------------------------------------------*/
/* Allocate storage for the largest print option and                  */
/* query each option for it's value.                                 */
/*-------------------------------------------------------------------*/
if ( (qrybuffer = (PBYTE) malloc( (USHORT) lengthNeeded ) ) == NULL)
   {
   printf("\nERROR memory allocation failure");
   goto Endit;
   }

length = lengthNeeded;              /* length of the qrybuffer     */
opt_data = (OPTDEFN2 *) qrybuffer;  /* points to optDefn2 type.    */

printf("\n List of Options in Print Descriptor\n");

                                    /* query all options from list */
for (i = 0; i < (USHORT) itemsReturned; i++)
   {
   opt_data->pszName = opt_names[i];/* set option name in optDefn2 */

   rc = PrdQueryOption(hprd,
                       level,
                       length,
                       qrybuffer,
                       &lengthNeeded);
   if (rc==0)
      {
      printf("ERROR issued by PrdQueryOption");
      extErrMsgs(hab);
      goto Endit;
      }

/*-------------------------------------------------------------------*/
/* Print the options to the screen                                   */
/*-------------------------------------------------------------------*/
   if (opt_data->pszName != NULL)
      printf("\n\n %s", opt_data->pszName);

   if (opt_data->pbDef != NULL)
      printf("\n      Default     = %s", opt_data->pbDef);

   if (opt_data->flRule != 0)
      printf("\n      Rule        = %ld", opt_data->flRule);

   if (opt_data->pszVv != NULL)
      printf("\n      ValidValues = %s", opt_data->pszVv);
   }
```

Figure D-1 (Part 8 of 12). Example of a C Language Print-Descriptor Edit Session

```
/*==================================================================*/
/* Set references                                  (PrdSetRefList) */
/*==================================================================*/

/*------------------------------------------------------------------*/
/* Define and set the first reference. This is a reference to       */
/* a print descriptor that describes a printer.                     */
/*------------------------------------------------------------------*/

references[0].idSeqNum = 10;        /* sequence # = 10.             */
                                    /* reference used by build.     */
references[0].flProcessFlag = PRD_BUILD;
strcpy(refName[0],"PRDNAME=My Printer");
references[0].pszRefName = refName[0];

/*------------------------------------------------------------------*/
/* Define and set the second reference. This is a reference to      */
/* a print descriptor that describes the type of media used by the  */
/* printer.                                                         */
/*------------------------------------------------------------------*/

references[1].idSeqNum = 20;        /* sequence # = 20.             */
                                    /* reference not used by build.*/
references[1].flProcessFlag = PRD_NOBUILD;
                                    /* name of Print Descriptor     */
strcpy(refName[1], "PRDNAME=Standard Computer Paper");
references[1].pszRefName = refName[1];

level = 1;                          /* using array of PRDREF1 type.*/
length = sizeof(references);        /* length of buffers.           */
buffer = (PBYTE) references;        /* buffer = references.         */
totalItems = 2;                     /* number of references.        */

rc = PrdSetRefList(hprd,
                   level,
                   length,
                   buffer,
                   totalItems);

if (rc==0)
   {
   printf("\nERROR issued by first PrdSetRefList");
   extErrMsg(hab);
   goto Endit;
   }
```

Figure D-1 (Part 9 of 12). Example of a C Language Print-Descriptor Edit Session

```
   /*=================================================================*/
   /* Save print descriptor                          (PrdSaveDescriptor) */
   /*=================================================================*/

   /*-----------------------------------------------------------------*/
   /* Define the prdDefn2 structure, naming the Prd, giving it a      */
   /* a description and a type.                                       */
   /*-----------------------------------------------------------------*/
   strcpy(prdName, "PRDNAME=FRED");            /* Prd name            */
   prdDefn2.pszPrdName = prdName;

                                    /* Prd description                */
   strcpy(prdDefn2.szDescription, "Fred's normal printer");

                                    /* id (type) of this Prd          */
   prdDefn2.idPrdId = PRD_PRESENTATION_DEVICE;

   /*-----------------------------------------------------------------*/
   /* Save the Prd with the control parameters set to always update   */
   /* the Prd regardless if it previously existed, and  automatically */
   /* create a new Prd group for the new Prd.                         */
   /*                                                                 */
   /* NOTE: If the Prd group already exists, then the "controlGrp"    */
   /* parameter must be changed to PRD_NOAUTO_CREATE.                 */
   /*-----------------------------------------------------------------*/

   level = 2;                          /* using PRDDEFN2 type.        */
   length = sizeof(PRDDEFN2);          /* length of buffers.          */
   buffer = (PBYTE) &prdDefn2;         /* buffer = prdDefn2 buffer.   */
   controlPrd = PRD_CREATE_OR_UPDATE;  /* Always update Prd.          */
   controlGrp = PRD_AUTO_CREATE;       /* Create new group.           */
   rc = PrdSaveDescriptor(hprd,
                          level,
                          length,
                          buffer,
                          controlPrd,
                          controlGrp);

   if (rc==0)
      {
      printf("\n\nERROR issued by PrdSaveDescriptor");
      extErrMsgs(hab);
      }


Endit:

   if (listbuffer != NULL)
      free (listbuffer);

   if (qrybuffer != NULL)
      free (qrybuffer);
```

Figure D-1 (Part 10 of 12). Example of a C Language Print-Descriptor Edit Session

```
   /*================================================================*/
   /* Close Print-Descriptor session                      (PrdClose) */
   /* Free error storage.                  (PrtMgrFreeErrorInfo) */
   /* Terminate Print Manager process         (PrtMgrTerminate) */
   /*================================================================*/

   rc = PrdClose(hprd);
   if (rc==0)
      {
      printf("\nERROR issued by PrdClose");
      extErrMsgs(hab);
      }

   if (pErrInfo != NULL)
      {
      rc = PrtMgrFreeErrorInfo(pErrInfo);
      if (rc==0)
         {
         printf("\nERROR issued by PrtMgrFreeErrorInfo");
         }
      }

   rc = PrtMgrTerminate(hab);

   if (rc==0)
      {
      printf("\nERROR issued by PrtMgrTerminate");
      }

   return(rc);
}

  /*********************************************************************/
  /*                    Display Error Function                         */
  /*********************************************************************/
  static void extErrMsgs(HAB hab)
  {
     BOOL         rc;
     ERRMSG       errMsg;
     LONG         index;
     ULONG        totalCount;

     /*================================================================*/
     /* Get error information                (PrtMgrGetErrorInfo) */
     /* Get extended error messages         (PrtMgrGetExtErrorMsg) */
     /*================================================================*/
```

Figure D-1 (Part 11 of 12). Example of a C Language Print-Descriptor Edit Session

```
   /*-------------------------------------------------------------*/
   /* The idError field in the ERRINFO structure returned by      */
   /* PrtMgrGetErrorInfo contains the error code in the 2 high order */
   /* bytes and the severity in the 2 low order bytes.            */
   /*-------------------------------------------------------------*/
   pErrInfo = PrtMgrGetErrorInfo(hab);
   if (pErrInfo)
      {
      printf("\n  Error Code = %X:    Severity = %X\n",
         ERRORIDERROR(pErrInfo->idError),ERRORIDSEV(pErrInfo->idError));

      if (ERRORIDERROR(pErrInfo->idError)== PMERR_PRD_CONTROLGRP_ERROR)
         {
         printf("\n");
         printf("\n    While attempting to write a print descriptor");
         printf("\n    from memory to disk with the PrdSaveDescriptor");
         printf("\n    verb, it was found that the user specified");
         printf("\n    PRD_AUTO_CREATE when the specified group");
         printf("\n    already existed on the disk, or the user");
         printf("\n    specified PRD_NOAUTO_CREATE when the specified");
         printf("\n    group did not exist.");
         printf("\n");
         }

      index = totalCount = 0;          /* init. message index        */

      rc = TRUE;
      while (index <= totalCount)
         {
         index = index +  1;             /* increment message index  */

         rc = PrtMgrGetExtErrorMsg(pErrInfo,
                                   index,            /* message number */
                                   &errMsg,
                                  &totalCount); /* returns number of msg*/

         if (rc == 1 && totalCount > 0)
            {
            printf("\n   Message identifier: %X", errMsg.idIdentifier);
            printf("\n   Message value     :  %s", errMsg.szValue);
            }
         } /* endwhile */
      }

   printf("\n");
   return;
}
```

Figure D-1 (Part 12 of 12). Example of a C Language Print-Descriptor Edit Session

Appendix E. Print Descriptor Tool Messages and API Verb Error Codes

This appendix provides:

- Information about Print Descriptor Tool messages in "Print Descriptor Tool Messages"
- Information about the API verb error codes in "API Verb Error Codes" on page E-16.

Note: For C applications in the System/370 environment, when you pass an invalid handle (in the *hab* or *hprd* verb parameters) to a PrintManager verb, your application program will abend with abend code X'0245'. A program dump is initiated unless the following C pragma preprocessor directive is coded in your main C routine:

```
#pragma runopts(nospie, nostae)
```

This pragma shown above disables the C/PLI abend handler. Refer to *IBM C/370 User's Guide* for more information on the C pragma directive.

Print Descriptor Tool Messages

PrdT messages are described in the following format:

- The message and message text
- Explanation
- Response
- Additional messages issued (if any). Additional message information includes:
  - Message and message text
  - *Value* string containing additional error information. For more information on the tag values or command parameters that contain these strings, refer to Chapter 5, Print Descriptor Tool Reference.

PRDT0000

Batch Print Descriptor Tool ended successfully

Explanation: No error occurred.

Response: No response is necessary.

PRDT4102

Valid values conflict with the type of rule for the option

Explanation: A valid value could not be set either:

- Because incorrect syntax was used in setting a valid value
- Because a valid value's range (for options with a validation rule of **PRD_RANGE**) or length (for options with a validation rule of **PRD_STRING**) did not comply with the validation rule when setting or merging print-option information.

Response: Either:

- Correct the syntax of the incorrect valid value, or

- Specify valid values that comply with the rules for validation and merging.

For general information on print options, refer to Chapter 9, Print Options. For information on the syntax, validation rule, and allowable values for specific print options refer to Appendix A, PrintManager Print Options.

PRDT4103

Default conflicts with the rule or valid values for the option

Explanation: An error occurred either:

- Because a default value for a print option was incorrectly specified. The specified default value either did not comply with the validation rule for the print option or was not a valid value for the print option.
- Because print-option information was not merged.

Response: When setting a default value, specify a default value that complies with the validation rule for the option. When merging print-option information, default values from referenced print descriptors are not used unless no default is specified in the print descriptor that contains the references.

For general information on print options, refer to Chapter 9, Print Options. For information on the syntax, validation rule, and allowable values for specific print options refer to Appendix A, PrintManager Print Options.

PRDT4104

Invalid type of rule

Explanation: An error occurred because the specified validation rule is not valid.

Response: Specify a valid validation rule for the print option.

For general information on print options, refer to Chapter 9, Print Options. For information on the syntax, validation rule, and allowable values for specific print options refer to Appendix A, PrintManager Print Options.

PRDT4106

Error opening a print descriptor group or interchange format print descriptor

Explanation: A StdPrd or a print descriptor in interchange format could not be retrieved, stored, or listed because:

- The specified print-descriptor group was not found
- The user did not have access to the print-descriptor group (MVS and OS/400)
- The record format of the print-descriptor group is invalid (MVS and VM).

Response: Ensure that the specified print-descriptor group or interchange format print descriptor exists, it has a valid record format, or that the user has access to the print-descriptor group. For general information on print descriptors, print-descriptor groups, and print-descriptor name formats, refer to Chapter 6, Using the API Verbs.

Additional Message Information:

Message	Value
PRDT4106	*GroupName* or *file name*

PRDT4107

Error reading a print descriptor group

Explanation: A print-descriptor read error occurred.

Response: If no hardware error occurred, re-create the print descriptor.

Additional Message Information:

Message	Value
PRDT4107	*GroupName*

PRDT4108

Error seeking a print descriptor group

Explanation: A print-descriptor access error occurred.

Response: If no hardware error occurred, re-create the print descriptor.

Additional Message Information:

Message	Value
PRDT4108	*GroupName, PrdName*

PRDT4109

Error writing to a print descriptor group

Explanation: A print-descriptor write error occurred.

Response: If no hardware error occurred, re-create the print descriptor.

Additional Message Information:

Message	Value
PRDT4109	*GroupName, PrdName*

PRDT410B

The specified print descriptor has an invalid format

Explanation: The format of a print descriptor (either in system-specific or interchange format) is not correct.

Response: Ensure that the print descriptor is in the correct format. For more information, refer to "Exchanging Print Descriptors between Systems" on page 3-10.

Additional Message Information:

Message	Value
PRDT410B	*GroupName, PrdName*

PRDT410C

Error building a composite print descriptor

Explanation: An error occurred while PrintManager was creating a composite print descriptor.

Response: In the additional error messages shown in "Additional Message Information." *SeqNum* provides the sequence number of the referenced print descriptor that caused the error, and *Name* provides the option name associate with the error.

Additional Message Information:

Message	Value
PRDT4102	*SeqNum, Name*
PRDT4103	*SeqNum, Name*
PRDT4106	*SeqNum, Name*
PRDT4107	*SeqNum, Name*
PRDT4108	*SeqNum, Name*
PRDT410B	*SeqNum, Name*
PRDT410C	*SeqNum, Name*
PRDT410D	*SeqNum, Name*
PRDT410E	*SeqNum, Name*
PRDT410F	*SeqNum, Name*
PRDT4115	*SeqNum, Name*
PRDT4117	*SeqNum, Name*
PRDT411B	*SeqNum, Name*
PRDT411C	*SeqNum, Name*
PRDT411D	*SeqNum, Name*
PRDT411E	*SeqNum, Name*

PRDT410D

Specified alias not found in group-list print descriptor

Explanation: A print-descriptor group could not be found because the GLPrd did not contain the specified group alias name.

Response: Either specify the correct alias name, use the exact-name format to specify the print-descriptor group, or define an alias name for the print-descriptor group.

Additional Message Information:

Message	Value
PRDT410D	*GroupAlias*

PRDT410E

UserGlPrdName is not found or syntax is invalid

Explanation: The specified UserGLPrdName could not be found either:

- Because the syntax was not valid
- Because the UserGLPrdName could not be retrieved.

Response: If no hardware error occurred, use correct syntax to specify the UserGLPrdName. Refer to Chapter 3, Overview of Print Descriptors for more information.

Additional Message Information:

Message	Value
PRDT4E00 UserGLPrdName has a syntax error	UserGLPrdName
PRDT4E01 UserGLPrdName cannot be accessed	None

PRDT410F

Option can not be merged because of a rule mismatch with the existing rule

Explanation: Print-option information cannot be merged because the new print-option validation rule does not match the existing rule.

Response: Ensure that the print-option validation rules match when merging print-option information. For general information on print option-validation rules, refer to "Print Option Validation" on page 9-4. For information on the validation rules for specific print options refer to Appendix A, PrintManager Print Options.

Additional Message Information:

Message	Value
PRDT410F	*Name*

PRDT4110

Option with a rule of PRD_STRING has an invalid value and can not be merged

Explanation: Print-option information (with a validation rule of **PRD_STRING**) cannot be merged.

Response: If you get this error, contact your IBM support Center representative.

PRDT4111

Option with a rule of PRD_RANGE has an invalid value and can not be merged

Explanation: Print-option information (with a validation rule of **PRD_RANGE**) cannot be merged.

Response: If you get this error, contact your IBM support Center representative.

PRDT4112

Option with a rule of PRD_LIST has an invalid value and can not be merged

Explanation: Print-option information (with a validation rule of **PRD_LIST**) cannot be merged.

Response: If you get this error, contact your IBM support Center representative.

PRDT4113

CONTROLPRD value is invalid for existence or non-existence of print descriptor

Explanation: The specified value for **CONTROLPRD** keyword or CONTROLPRD tag is not valid.

Response: Specify a valid value for the **CONTROLPRD** keyword or CONTROLPRD tag.

PRDT4114

CONTROLGRP value is invalid for existence or non-existence of print descriptor group

Explanation: The specified value for **CONTROLGRP** keyword or CONTROLGRP tag is not valid.

Response: Specify a valid value for the **CONTROLGRP** keyword or CONTROLGRP tag.

PRDT4115

Invalid name format specified for the print descriptor

Explanation: The format of the specified print-descriptor name is not valid.

Response: Use a valid name format to specify the print descriptor. For more information, refer to "Print-Descriptor Name Formats" on page 3-4.

PRDT4116

Unrecognized value for CONTROLPRD or CONTROLGRP

Explanation: The value specified for either the **CONTROLPRD** keyword or CONTROLPRD tag or for the **CONTROLGRP** keyword or CONTROLGRP tag is not valid.

Response: Specify a valid value for the keyword or tag. For more information, refer to Chapter 5, Print Descriptor Tool Reference.

PRDT4117

No valid groups entries specified to search in the group list print descriptor

Explanation: Print-descriptor groups in the specified GLPrd could not be searched either:

- Because the GLPrd contains no entries
- Because the PROCESSFLAG tag value is specified as **PRD_NOSEARCH** for all entries.

Response: Either specify a GLPrd with entries or, for entries you want to include in the search order, change the PROCESSFLAG tag value to **PRD_SEARCH**.

Additional Message Information:

Message	Value
PRDT4117	*PrdName*

PRDT411A

Value specified for the type conflicts with a previous setting

Explanation: The specified print descriptor cannot be stored (with Store a Print Descriptor) because the print-descriptor type specified in the *PrdId* field does not match the actual print-descriptor contents.

Response: Specify the correct print-descriptor type on the PRDID tag. For more information, refer to "Assigning Print-Descriptor Types" on page 3-11.

PRDT411B

Print descriptor group not accessed because it's type is not PRD_GROUP_LIST

Explanation: A GLPrd cannot be accessed because the print-descriptor type is not **PRD_GROUP_LIST**.

Response: Either change the print-descriptor type to **PRD_GROUP_LIST** or specify a valid GLPrd.

Additional Message Information:

Message	Value
PRDT411B	*PrdName*

PRDT411C

Print descriptor group not accessed because format of the group is invalid

Explanation: A print-descriptor group cannot be accessed because the format of the print-descriptor group is not valid.

Response: Recreate the specified group. For more information, refer to "Print-Descriptor Concepts" on page 3-3.

Additional Message Information:

Message	Value
PRDT411C	*GroupName*

PRDT411D

Specified print descriptor could not be found

Explanation: The specified print descriptor cannot be found.

Response: Either specify a valid print descriptor or re-create the print descriptor.

PRDT411E

Specified group list print descriptor could not be found

Explanation: The specified GLPrd cannot be found.

Response: Either specify a valid GLPrd or re-create the GLPrd. For more information, refer to "Print-Descriptor Concepts" on page 3-3.

Additional Message Information:

Message	Value
PRDT411E	*PrdName*

PRDT4120

Option not found in print descriptor

Explanation: A print option could not be queried or deleted because the print option specified in the *Name* field is not defined for the current PRD TOOL session.

Response: Either specify an option that is in the current PRD TOOL session or use the Get a Print Descriptor function to get a print descriptor with the desired option.

PRDT4121

Option has invalid name or is missing

Explanation: A print option could not be set, queried, or deleted because the print-option name is either null, blank, or not specified.

Response: Either specify an option that is in the current PRD TOOL session or use Get a Print Descriptor to get a print descriptor with the desired option.

PRDT4123

An error occurred while PrintManager was deleting a print-descriptor group

Explanation: An error occurred while PrintManager was deleting a print-descriptor group.

Response: If no hardware error or system error occurred, ensure that you have authorization to delete the print-descriptor group.

PRDT4124

An error occurred while PrintManager was compressing a print-descriptor group

Explanation: An error occurred while PrintManager was compressing storage in the print-descriptor group named in *GroupName* in the additional error message.

Response: In the additional error message, the group named in *GroupName* no longer has valid data. Copy the group named in *TemporaryGroupName* (which is a backup version of the group) to the group named in *GroupName*.

Additional Message Information:

Message ID	Value
PRDT4124	*TemporaryGroupName*, *GroupName*

PRDT4125

Total size of option information for the print descriptor exceeds 64K bytes

Explanation: The total size of all print options in the current PRD TOOL session exceeds the limit of 64K bytes.

Response: Use the Delete a Print Option function to delete print options from the current PRD TOOL session.

Additional Message Information:

Message	Value
PRDT4125	Total size of all print options in the current PRD TOOL session.

PRDT4126

Total size of reference information for the print descriptor exceeds 64K bytes

Explanation: The total size of all print-descriptor references in the current PRD TOOL session exceeds the limit of 64K bytes.

Response: Use the Delete a Print-Descriptor Reference function to delete print-descriptor references from the current PRD TOOL session.

Additional Message Information:

Message	Value
PRDT4126	Total size of all print-descriptor references in the current PRD TOOL session.

PRDT4127

Total size of group list information for the print descriptor exceeds 64K bytes

Explanation: The total size of all group-list entries in the current PRD TOOL session exceeds the limit of 64K bytes.

Response: Use the Delete a Group-List Entry function to delete group-list entries from the current PRD TOOL session.

Additional Message Information:

Message	Value
PRDT4127	Total size of all group-list entries in the current PRD TOOL session.

PRDT4129

Default value of option was truncated

Explanation: For an option of type **PRD_RANGE**, the specified default value has a decimal precision that is greater than the decimal precision of the valid value. Therefore, the default was truncated and set to the truncated value.

Response: Either:

- Use the truncated default value
- Respecify the default value with a different precision
- Modify the valid value and respecify the default.

For more information, refer to "Print Option Validation" on page 9-4.

Additional Message Information:

Message ID	Value
PRDT4129	Option name, truncated default

PRDT412A

Minimum and/or maximum value of option was truncated.

Explanation: For an option of type **PRD_RANGE**, either the minimum or maximum value being set has a decimal precision that is greater than the precision specified for the valid value. Therefore, the minimum or maximum value (or both) was truncated.

Response: Either:

- Use the truncated valid value
- Respecify the valid value with a different precision
- Respecify the precision and the valid value.

For more information, refer to "Print Option Validation" on page 9-4.

Additional Message Information:

Message ID	Value
PRDT412A	Option name, truncated valid value(s)

PRDT4401

File name for input file was not specified

Explanation: No file name was specified for the input tag file.

Response: Specify a file name for the input tag file.

PRDT4402

Error opening the input file

Explanation: An error occurred while opening the input tag file.

Response: Ensure that the input tag file exists.

PRDT4403

Error reading the input file

Explanation: An error occurred while reading the input tag file.

Response: If no hardware error occurred, re-create the input tag file.

PRDT4404

No tags found in the input file

Explanation: No tags were found in the input file.

Response: Create an input file with tags in correct format.

PRDT4405

Memory allocation failure

Explanation: Processor storage was not sufficient.

Response: Allocate more processor storage.

PRDT4406

Input string exceeded the maximum length

Explanation: A PrdT command input string exceeded the maximum length.

Response: Reduce the size of the input string to less than 1000 characters (for VM) or less than 100 characters (for MVS).

PRDT4407

Invalid tag was specified

Explanation: An invalid PrdT tag was specified.

Response: Specify a valid PrdT tag. For more information, refer to "Print Descriptor Tool Tag Reference" on page 5-14.

PRDT4408

Tag specified in invalid position

Explanation: A valid PrdT tag was specified in an incorrect position in the PRD TOOL input file.

Response: Specify the tag in a correct position in the tag file. For more information, refer to "Print Descriptor Tool Tags" on page 4-2.

PRDT4410

An internal error occurred

Explanation: A PrintManager error occurred during processing of a PrdT command.

Response: If you get this error, contact your IBM support Center representative.

PRDT4411

Print descriptor name not specified

Explanation: No print-descriptor name was specified.

Response: Specify valid print-descriptor name. For more information, refer to Chapter 5, Print Descriptor Tool Reference.

PRDT4412

Option name not specified

Explanation: No print option name was specified on a PrdT tag.

Response: Specify a valid print option name. For general information on print options, refer to Chapter 9, Print Options. For information on the syntax, validation rule, and allowable values for specific print options refer to Appendix A, PrintManager Print Options.

PRDT4414

File id not specified

Explanation: No file id was specified.

Response: Specify a valid file id.

PRDT4415

Invalid sequence number

Explanation: An invalid sequence number was specified on a PrdT tag.

Response: Specify a valid sequence number. For more information on sequence numbers, refer to "Print Descriptor Tool Tag Reference" on page 5-14 and "Contents of Group-List Print Descriptors (GLPrds)" on page 3-3.

PRDT4416

Invalid process flag

Explanation: An invalid process value was specified.

Response: Specify a valid process value. For more information, refer to Chapter 5, Print Descriptor Tool Reference.

PRDT4417

Invalid print descriptor ID specified

Explanation: The specified print descriptor cannot be saved (with the Store a Print Descriptor tag function) because the print-descriptor type specified on the PRDID tag does not match the actual print-descriptor contents.

Response: Specify the correct print-descriptor type for the PRDID tag. For more information, refer to "Store a Print Descriptor" on page 5-29 and "Assigning Print-Descriptor Types" on page 3-11.

PRDT4418

Unrecognized value for CONTROLPRD

Explanation: The specified value for the CONTROLPRD tag or the **CONTROLPRD** keyword is not valid.

Response: Specify a valid value. For more information, refer to Chapter 5, Print Descriptor Tool Reference.

PRDT4419

Unrecognized value for CONTROLGRP

Explanation: The specified value for the CONTROLGRP tag or the **CONTROLGRP** keyword is not valid.

Response: Specify a valid value. For more information, refer to Chapter 5, Print Descriptor Tool Reference.

PRDT4420

Comment started but no end of comment found

Explanation: A comment was begun in a tag file but no end of comment was found.

Response: Add the end of comment characters (*/).

PRDT4421

No arguments specified on command line

Explanation: No keywords or variables were specified for a PrdT command.

Response: Specify valid keywords or parameters. For more information, refer to "Print Descriptor Tool Command Reference" on page 5-1.

PRDT4422

No print descriptors found to list

Explanation: No print descriptors were found for the requested list.

Response: No response is necessary.

PRDT4423

Print descriptor group name not specified

Explanation: No print-descriptor group name was specified on the command line.

Response: Specify a valid group name. For more general information on print-descriptor names, refer to "Print-Descriptor Name Formats" on page 3-4.

PRDT4424

Invalid or missing keyword

Explanation: When specifying a parameter for a PrdT command either:

- An invalid keyword was specified
- A required keyword was missing.

Response: Specify the correct keyword or add the missing keyword. For more information on PrdT command parameters, refer to "Invoking the Print Descriptor Tool Commands" on page 5-1, "Format of the Command Descriptions" on page 5-2, and "Command Reference" on page 5-4.

PRDT4425

No value specified for keyword

Explanation: No value was specified on a PrdT command parameter.

Response: Specify a value for the variable for the keyword for the specified parameter. For more information on PrdT command parameters, refer to "Invoking the Print Descriptor Tool Commands" on page 5-1, "Format of the Command Descriptions" on page 5-2, and "Command Reference" on page 5-4.

PRDT4426

Target print descriptor group or alias not specified, or specified incorrectly

Explanation: No target print-descriptor group was specified on the PRD COPY command or no keyword was specified or a value was not specified in the correct position.

Response: Specify a target print-descriptor group. For more information on the PRD COPY command, refer to "PRD COPY" on page 5-5.

PRDT4427

Invalid name for input tag file

Explanation: A PRD TOOL output file was used as an PRD TOOL input file without renaming the file.

Response: Rename the output file, then use the renamed file as a PRD TOOL input file.

PRDT4428

Target print descriptor name not specified, or specified incorrectly

Explanation: No target print-descriptor name was specified for the PRD COPY command or no keyword was specified or a value was not specified in the correct position.

Response: Specify a target print-descriptor name. For more information on the PRD COPY command, refer to "PRD COPY" on page 5-5.

PRDT4429

Source print descriptor name not specified, or specified incorrectly

Explanation: No source print-descriptor name was specified for the PRD COPY command or no keyword was specified or a value was not specified in the correct position.

Response: Specify a source print-descriptor name. For more information on the PRD COPY command, refer to "PRD COPY" on page 5-5.

PRDT4430

Command line arguments incorrect

Explanation: PrdT command parameters were incorrectly specified.

Response: Specify correct parameters for the desired PrdT command. For more information on PrdT commands, refer to "Print Descriptor Tool Command Reference" on page 5-1.

PRDT509G

There is no virtual disk address available.

Explanation: There is no virtual disk address available to access the disk that contains the licensed programs required for the PrdT.

Response: Contact your system administrator.

PRDT509H

There is no disk mode available for access.

Explanation: There is no disk mode available to access the disk that contains the licensed programs required for the PrdT.

Response: Make a disk mode available for access. If necessary, contact your system administrator.

PRDT509I

You are unable to link to the required licensed programs.

Explanation: You were unable to access the disk that contains the licensed programs required for the PrdT.

Response: Contact your system administrator.

PRDT509J

No command or invalid command specified on command line

Explanation: A command operand was not specified for a PrdT command or an invalid command was specified.

Response: Specify a PrdT command operand. For more information, refer to "Print Descriptor Tool Commands" on page 4-1 or "Print Descriptor Tool Command Reference" on page 5-1.

PRDT509Z

The option you specified is not supported or is misspelled.

Explanation: A specified PrdT command option is either not supported or was misspelled.

Response: Specify a valid PrdT command keyword. For more information, refer to "Print Descriptor Tool Command Reference" on page 5-1.

PRDT599W

Message not found

Explanation: The file containing the PrdT messages exists, but a message is missing from the file.

Response: Contact your system administrator.

PRDT599X

Error reading message file

Explanation: The file containing the PrdT messages could not be accessed.

Response: Contact your system administrator.

PRDT599Y

Language specified is not supported

Explanation: The language specified on the **LANG** keyword is not supported.

Response: Either

- Do not specify the **LANG** keyword to use the default language
- Specify a valid value for the **LANG** keyword. See your system administrator for the values that correspond to the languages that are supported at your location.

PRDT599Z

Message file not found

Explanation: The file containing the PrdT messages could not be found.

Response: Contact your system administrator.

API Verb Error Codes

API verb error codes are described in the following format:

- The error code, which consists of the hexadecimal error value and corresponding constant. These constants will be included in your program if you define INCL_ERRORS and include the ekipmgr.h file as described in "Header Files" on page C-1.
- Explanation
- Severity level of the message. For more information on severity levels, refer to "Using PrintManager Error Verbs" on page 6-4.
- List of API verbs that may have issued the error
- Response
- Additional messages issued (if any). Additional message information includes:
  - Message ID, which consists of the hexadecimal error value and corresponding constant.
  - *Value* string containing additional error information.

    For more information on the parameters that contain these strings, either refer to Chapter 7, API Verb Reference for descriptions of the verbs issuing the error codes or refer to "API Verb Data Types" on page 7-2 for descriptions of the data structures that contain these parameters.

For more information on using the API verbs for problem diagnosis, refer to "Using PrintManager Error Verbs" on page 6-4.

X'0000'

PMERR_OK

Explanation: No error occurred.

Severity: None

API Verbs: All

Response: No response is necessary.

X'4007'

PMERR_PRD_NO_MEMORY

Explanation: Processor storage was not sufficient.

Severity: Error

API Verbs: PDBLDD (Build Descriptor), PDEXPD (Export Descriptor), PDGETD (Get Descriptor), PDIMPD (Import Descriptor), PDLDEF (List Descriptors), PDOPEN (Open Session), PDSAVD (Save Descriptor), PDSGRP (Set Group List), PDSOPT (Set Print Option), PDSREF (Set Reference List)

Response: Either allocate more processor storage or reduce the storage required by the API application.

X'4100'

PMERR_PRD_INV_HPRD

Explanation: The *hprd* parameter does not identify a valid edit session.

Severity: Error

API Verbs: PDBLDD (Build Descriptor), PDCLS (Close Session), PDREMO (Delete Print Option), PDGETD (Get Descriptor), PDLOPT (List Print Options), PDLGRP (Query Group List), PDQOPT (Query Print Option), PDLREF (Query Reference List), PDSAVD (Save Descriptor), PDSGRP (Set Group List), PDSOPT (Set Print Option), PDSREF (Set Reference List)

Response: Ensure that:

- You use the PDOPEN (Open Session) verb to open an edit session and return a session identifier.
- You use the session identifier on subsequent API verbs that require this identifier.

X'4102'

PMERR_PRD_INV_VALID_VALUES

Explanation: A valid value *could not* be set either:

- Because incorrect syntax was used in setting a valid value
- Because a valid value's range (for options with a validation rule of **PRD_RANGE**) or length (for options with a validation rule of **PRD_STRING**) did not comply with the validation rule when setting or merging print-option information.

Severity: Error

API Verbs: PDBLDD (Build Descriptor), PDSOPT (Set Print Option)

Response: Either:

- Correct the syntax of the incorrect valid value, or
- Specify valid values that comply with the rules for validation and merging.

For general information on print options, refer to Chapter 9, Print Options. For information on the syntax, validation rule, and allowable values for specific print options refer to Appendix A, PrintManager Print Options.

X'4103'

PMERR_PRD_INV_DEFAULT

Explanation: An error occurred either:

- Because a default value for a print option was incorrectly specified. The specified default value either did not comply with the validation rule for the print option or was not a valid value for the print option.
- Because print-option information was not merged.

Severity: Error

API Verbs: PDBLDD (Build Descriptor), PDSOPT (Set Print Option)

Response: When setting a default value, specify a default value that complies with the validation rule for the option. When merging print-option information, default values from referenced print descriptors are not used unless no default is specified in the print descriptor that contains the references.

For general information on print options, refer to Chapter 9, Print Options. For information on the syntax, validation rule, and allowable values for specific print options refer to Appendix A, PrintManager Print Options.

X'4104'

PMERR_PRD_INV_RULE

Explanation: An error occurred because the specified validation rule is not valid.

Severity: Error

API Verbs: PDSOPT (Set Print Option)

Response: Specify a valid validation rule for the print option.

For general information on print options, refer to Chapter 9, Print Options. For information on the syntax, validation rule, and allowable values for specific print options refer to Appendix A, PrintManager Print Options.

X'4105'

PMERR_PRD_NO_PRD_NAME

Explanation: No print-descriptor name was specified.

Severity: Error

API Verbs: PDEXPD (Export Descriptor), PDGETD (Get Descriptor), PDIMPD (Import Descriptor), PDSAVD (Save Descriptor)

Response: Specify a valid print-descriptor name on the PDGETD (Get Descriptor), PDSAVD (Save Descriptor), PDIMPD (Import Descriptor), or PDEXPD (Export Descriptor) verb. If you do not specify a print-descriptor name with PDSAVD (Save Descriptor), a name must exist in the current edit session via the PDGETD (Get Descriptor) verb or the save will fail. For more information about these verbs, refer to Chapter 7, API Verb Reference.

X'4106'

PMERR_PRD_OPEN_ERROR

Explanation: A StdPrd or a print descriptor in interchange format could not be retrieved, stored, or listed because:

- The specified print-descriptor group was not found
- The user did not have access to the print-descriptor group (MVS and OS/400)
- The record format of the print-descriptor group is invalid (MVS and VM).

Severity: Error

API Verbs: PDBLDD (Build Descriptor), PDRDEF (Delete Descriptor), PDEXPD (Export Descriptor), PDGETD (Get Descriptor), PDIMPD (Import Descriptor), PDLDEF (List Descriptors)

Response: Ensure that the specified print-descriptor group or interchange format print descriptor exists, it has a valid record format, or that the user has access to the print-descriptor group. For general information on print descriptors, print-descriptor groups, and print-descriptor name formats, refer to Chapter 6, Using the API Verbs.

Additional Message Information:

Message ID	Value
X'4106' PMERR_PRD_OPEN_ERROR	*GroupName, file name*

X'4107'

PMERR_PRD_READ_ERROR

Explanation: A print-descriptor read error occurred.

Severity: Error

API Verbs: PDBLDD (Build Descriptor), PDRDEF (Delete Descriptor), PDEXPD (Export Descriptor), PDGETD (Get Descriptor), PDIMPD (Import Descriptor), PDLDEF (List Descriptors)

Response: If no hardware error occurred, re-create the print descriptor.

Additional Message Information:

Message ID	Value
X'4107' PMERR_PRD_READ_ERROR	*GroupName*

X'4108'

PMERR_PRD_SEEK_ERROR

Explanation: A print-descriptor access error occurred.

Severity: Error

API Verbs: PDBLDD (Build Descriptor), PDRDEF (Delete Descriptor), PDEXPD (Export Descriptor), PDGETD (Get Descriptor), PDIMPD (Import Descriptor), PDSAVD (Save Descriptor)

Response: If no hardware error occurred, re-create the print descriptor.

Additional Message Information:

Message ID	Value
X'4108' PMERR_PRD_SEEK_ERROR	*GroupName*, *PrdName*

X'4109'

PMERR_PRD_WRITE_ERROR

Explanation: A print-descriptor write error occurred.

Severity: Error

API Verbs: PDRDEF (Delete Descriptor), PDEXPD (Export Descriptor), PDIMPD (Import Descriptor), PDSAVD (Save Descriptor)

Response: If no hardware error occurred, re-create the print descriptor.

Additional Message Information:

Message ID	Value
X'4109' PMERR_PRD_WRITE_ERROR	*GroupName*, *PrdName*

X'410B'

PMERR_PRD_FORMAT_ERROR

Explanation: The format of a print descriptor (either in system-specific or interchange format) is not correct.

Severity: Error

API Verbs: PDBLDD (Build Descriptor), PDEXPD (Export Descriptor), PDGETD (Get Descriptor), PDIMPD (Import Descriptor)

Response: Ensure that the print descriptor is in the correct format. For more information, refer to "Exchanging Print Descriptors between Systems" on page 3-10.

Additional Message Information:

Message ID	Value
X'410B' PMERR_PRD_FORMAT_ERROR	*GroupName, PrdName*

X'410C'

PMERR_PRD_BUILD_ERROR

Explanation: An error occurred while PrintManager was creating a composite print descriptor.

Severity: Error

API Verbs: PDBLDD (Build Descriptor)

Response: Use the API error verbs to retrieve the additional error messages shown in "Additional Message Information." *SeqNum* provides the sequence number of the referenced print descriptor that caused the error, and *Name* provides the option name associate with the error. Refer to "Using PrintManager Error Verbs" on page 6-4 for more information on retrieving additional error messages.

Additional Message Information:

Message ID	Value
X'4102' PMERR_PRD_INV_VALID_VALUES	*SeqNum, Name*
X'4103' PMERR_PRD_INV_DEFAULT	*SeqNum, Name*
X'4106' PMERR_PRD_OPEN_ERROR	*SeqNum, Name*
X'4107' PMERR_PRD_READ_ERROR	*SeqNum, Name*
X'4108' PMERR_PRD_SEEK_ERROR	*SeqNum, Name*
X'410B' PMERR_PRD_FORMAT_ERROR	*SeqNum, Name*
X'410C' PMERR_PRD_BUILD_ERROR	*SeqNum, Name*
X'410D' PMERR_PRD_GRPALIAS_UNDEFINED	*SeqNum, Name*
X'410E' PMERR_PRD_GRPLISTPTR_ERROR	*SeqNum, Name*
X'410F' PMERR_PRD_RULE_MISMATCH	*SeqNum, Name*
X'4115' PMERR_PRD_INV_PRD_NAME	*SeqNum, Name*
X'4117' PMERR_PRD_GRPLIST_EMPTY	*SeqNum, Name*
X'411B' PMERR_PRD_INV_GRPLIST	*SeqNum, Name*
X'411C' PMERR_PRD_INV_GROUP	*SeqNum, Name*
X'411D' PMERR_PRD_NOT_FOUND	*SeqNum, Name*
X'411E' PMERR_PRD_GRPLIST_NOT_FOUND	*SeqNum, Name*

X'410D'

PMERR_PRD_GRPALIAS_UNDEFINED

Explanation: A print-descriptor group could not be found because the GLPrd did not contain the specified group alias name.

Severity: Error

API Verbs: PDBLDD (Build Descriptor), PDRDEF (Delete Descriptor), PDEXPD (Export Descriptor), PDGETD (Get Descriptor), PDIMPD (Import Descriptor), PDSAVD (Save Descriptor)

Response: Either specify the correct alias name, use the exact-name format to specify the print-descriptor group, or define an alias name for the print-descriptor group.

Additional Message Information:

Message ID	Value
X'410D' PMERR_PRD_GRPALIAS_UNDEFINED	*GroupAlias*

X'410E'

PMERR_PRD_GRPLISTPTR_ERROR

Explanation: The specified UserGLPrdName could not be found either:

- Because the syntax was not valid
- Because the UserGLPrdName could not be retrieved.

Severity: Error

API Verbs: PDBLDD (Build Descriptor), PDRDEF (Delete Descriptor), PDEXPD (Export Descriptor), PDGETD (Get Descriptor), PDIMPD (Import Descriptor), PDLDEF (List Descriptors), PDSAVD (Save Descriptor)

Response: If no hardware error occurred, use correct syntax to specify the UserGLPrdName. Refer to Chapter 6, Using the API Verbs for more information.

Additional Message Information:

Message ID	Value
X'4E00' GRPLISTPTR_SYNTAX_ERR	UserGLPrdName
X'4E01' GRPLISTPTR_NOT_ACCESSIBLE	None

X'410F'

PMERR_PRD_RULE_MISMATCH

Explanation: Print-option information cannot be merged because the new print-option validation rule does not match the existing rule.

Severity: Error

API Verbs: PDBLDD (Build Descriptor)

Response: Ensure that the print-option validation rules match when merging print-option information. For general information on print option-validation rules, refer to "Print Option Validation" on page 9-4. For information on the validation rules for specific print options refer to Appendix A, PrintManager Print Options.

Additional Message Information:

Message ID	Value
X'410F' PMERR_PRD_RULE_MISMATCH	*Name*

X'4110'

PMERR_PRD_BAD_STRING_MERGE

Explanation: Print-option information (with a validation rule of **PRD_STRING**) cannot be merged.

Severity: Error

API Verbs: PDBLDD (Build Descriptor)

Response: If you get this error, contact your IBM support Center representative.

X'4111'

PMERR_PRD_BAD_RANGE_MERGE

Explanation: Print-option information (with a validation rule of **PRD_RANGE**) cannot be merged.

Severity: Error

API Verbs: PDBLDD (Build Descriptor)

Response: If you get this error, contact your IBM support Center representative.

X'4112'

PMERR_PRD_BAD_LIST_MERGE

Explanation: Print-option information (with a validation rule of **PRD_LIST**) cannot be merged.

Severity: Error

API Verbs: PDBLDD (Build Descriptor)

Response: If you get this error, contact your IBM support Center representative.

X'4113'

PMERR_PRD_CONTROLPRD_ERROR

Explanation: The specified *ControlPrd* parameter is not valid.

Severity: Error

API Verbs: PDIMPD (Import Descriptor), PDSAVD (Save Descriptor)

Response: Specify a valid *ControlPrd* parameter. For more information, refer to "PDSAVD (Save Descriptor) PrdSaveDescriptor" on page 7-31.

X'4114'

PMERR_PRD_CONTROLGRP_ERROR

Explanation: The specified *ControlGrp* parameter is not valid.

Severity: Error

API Verbs: PDIMPD (Import Descriptor), PDSAVD (Save Descriptor)

Response: Specify a valid *ControlGrp* parameter. For more information, refer to "PDSAVD (Save Descriptor) PrdSaveDescriptor" on page 7-31.

X'4115'

PMERR_PRD_INV_PRD_NAME

Explanation: The format of the specified print-descriptor name is not valid.

Severity: Error

API Verbs: PDBLDD (Build Descriptor), PDRDEF (Delete Descriptor), PDGETD (Get Descriptor), PDIMPD (Import Descriptor), PDSAVD (Save Descriptor)

Response: Use a valid name format to specify the print descriptor. For more information, refer to "Print-Descriptor Name Formats" on page 3-4.

X'4116'

PMERR_PRD_INV_CONTROL

Explanation: The value specified for either the *ControlPrd* or the *ControlGrp* parameter is not valid.

Severity: Error

API Verbs: PDRDEF (Delete Descriptor), PDEXPD (Export Descriptor), PDIMPD (Import Descriptor), PDSAVD (Save Descriptor)

Response: For more information about these verbs, refer to Chapter 7, API Verb Reference.

X'4117'

PMERR_PRD_GRPLIST_EMPTY

Explanation: Print-descriptor groups in the specified GLPrd could not be searched either:

- Because the GLPrd contains no entries
- Because the *ProcessFlag* field is specified as **PRD_NOSEARCH** for all entries.

Severity: Error

API Verbs: PDBLDD (Build Descriptor), PDEXPD (Export Descriptor), PDGETD (Get Descriptor), PDIMPD (Import Descriptor), PDSAVD (Save Descriptor)

Response: Either specify a GLPrd with entries or, for entries you want to include in the search order, change the *ProcessFlag* field to **PRD_SEARCH**.

Additional Message Information:

Message ID	Value
X'4117' PMERR_PRD_GRPLIST_EMPTY	*PrdName*

X'4118'

PMERR_PRD_INV_LEVEL

Explanation: The specified value for the *Level* parameter is not valid.

Severity: Error

API Verbs: PDGETD (Get Descriptor), PDLGRP (Query Group List), PDQOPT (Query Print Option), PDLREF (Query Reference List), PDLDEF (List Descriptors), PDLOPT (List Print Options), PDSAVD (Save Descriptor), PDSGRP (Set Group List), PDSOPT (Set Print Option), PDSREF (Set Reference List)

Response: Specify a valid value for the *Level* parameter. For more information about these verbs, refer to Chapter 7, API Verb Reference.

X'4119'

PMERR_PRD_INV_BUFFER

Explanation: No storage was allocated for the buffer.

Severity: Error

API Verbs: PDGETD (Get Descriptor), PDLGRP (Query Group List), PDQOPT (Query Print Option), PDLREF (Query Reference List), PDLDEF (List Descriptors), PDLOPT (List Print Options), PDSAVD (Save Descriptor), PDSGRP (Set Group List), PDSOPT (Set Print Option), PDSREF (Set Reference List)

Response: Allocate sufficient buffer storage. For more information, refer to Chapter 7, API Verb Reference.

X'411A'

PMERR_PRD_ID_ERROR

Explanation: The specified print descriptor cannot be stored (with PDSAVD (Save Descriptor)) because the print-descriptor type specified in the *PrdId* field does not match the actual print-descriptor contents.

Severity: Error

API Verbs: PDSAVD (Save Descriptor)

Response: Specify the correct print-descriptor type in the *PrdId* field. For more information, refer to "Assigning Print-Descriptor Types" on page 3-11.

X'411B'

PMERR_PRD_INV_GRPLIST

Explanation: A GLPrd cannot be accessed because the type specified in the *PrdId* field is not **PRD_GROUP_LIST**.

Severity: Error

API Verbs: PDBLDD (Build Descriptor), PDRDEF (Delete Descriptor), PDEXPD (Export Descriptor), PDGETD (Get Descriptor), PDIMPD (Import Descriptor), PDSAVD (Save Descriptor)

Response: Either change the *PrdId* field to **PRD_GROUP_LIST** or specify a valid GLPrd.

Additional Message Information:

Message ID	Value
X'411B' PMERR_PRD_INV_GRPLIST	*PrdName*

X'411C'

PMERR_PRD_INV_GROUP

Explanation: A print-descriptor group cannot be accessed because the format of the print-descriptor group is not valid.

Severity: Error

API Verbs: PDBLDD (Build Descriptor), PDRDEF (Delete Descriptor), PDEXPD (Export Descriptor), PDGETD (Get Descriptor), PDIMPD (Import Descriptor), PDLDEF (List Descriptors), PDSAVD (Save Descriptor)

Response: Recreate the specified group. For more information, refer to "Print-Descriptor Concepts" on page 3-3.

Additional Message Information:

Message ID	Value
X'411C' PMERR_PRD_INV_GROUP	*GroupName*

X'411D'

PMERR_PRD_NOT_FOUND

Explanation: The specified print descriptor cannot be found.

Severity: Error

API Verbs: PDRDEF (Delete Descriptor), PDEXPD (Export Descriptor), PDGETD (Get Descriptor), PDIMPD (Import Descriptor)

Response: Either specify a valid print descriptor or re-create the print descriptor.

X'411E'

PMERR_PRD_GRPLIST_NOT_FOUND

Explanation: The specified GLPrd cannot be found.

Severity: Error

API Verbs: PDBLDD (Build Descriptor), PDRDEF (Delete Descriptor), PDEXPD (Export Descriptor), PDGETD (Get Descriptor), PDIMPD (Import Descriptor), PDSAVD (Save Descriptor)

Response: Either specify a valid GLPrd or re-create the GLPrd. For more information, refer to "Print-Descriptor Concepts" on page 3-3.

Additional Message Information:

Message ID	Value
X'411E' PMERR_PRD_GRPLIST_NOT_FOUND	*PrdName*

X'411F'

PMERR_PRD_BUFFER_TOO_SMALL

Explanation: The buffer size specified on the *Length* parameter is not sufficient.

Severity: Error

API Verbs: PDGETD (Get Descriptor), PDLDEF (List Descriptors), PDLOPT (List Print Options), PDLGRP (Query Group List), PDQOPT (Query Print Option), PDLREF (Query Reference List), PDSGRP (Set Group List), PDSOPT (Set Print Option), PDSREF (Set Reference List)

Response: Specify a sufficient buffer size. For more information, refer to Chapter 7, API Verb Reference.

X'4120'

PMERR_PRD_OPT_NOT_DEFINED

Explanation: A print option could not be queried or deleted because the print option specified in the *Name* field is not defined for the current edit session.

Severity: Error

API Verbs: PDREMO (Delete Print Option), PDQOPT (Query Print Option)

Response: Either specify an option that is in the current edit session or use PDGETD (Get Descriptor) to get a print descriptor with the desired option.

X'4121'

PMERR_PRD_INV_OPTNAME

Explanation: A print option could not be set, queried, or deleted because the print-option name is either null, blank, or not specified.

Severity: Error

API Verbs: PDREMO (Delete Print Option), PDQOPT (Query Print Option), PDSOPT (Set Print Option)

Response: Either specify an option that is in the current edit session or use PDGETD (Get Descriptor) to get a print descriptor with the desired option.

X'4122'

PMERR_PRD_TYPE_MISMATCH

Explanation: Print-option information cannot be merged because the print-option default types do not match.

Severity: Error

API Verbs: PDBLDD (Build Descriptor)

Response: Specify print options with matching default types.

X'4123'

PMERR_PRD_DESTROY_ERROR

Explanation: An error occurred while PrintManager was deleting a print-descriptor group.

Severity: Error

API Verbs: PDRDEF (Delete Descriptor)

Response: If no hardware error or system error occurred, ensure that you have authorization to delete the print-descriptor group.

X'4124'

PMERR_PRD_COMPRESS_ERROR

Explanation: An error occurred while PrintManager was compressing storage in the print-descriptor group named in *GroupName* in the additional error message.

Severity: Error

API Verbs: PDRDEF (Delete Descriptor)

Response: In the additional error message, the group named in *GroupName* no longer has valid data. Copy the group named in *TemporaryGroupName* (which is a backup version of the group) to the group named in *GroupName*.

Additional Message Information:

Message ID	Value
X'4124' PMERR_PRD_COMPRESS_ERROR	*TemporaryGroupName*, *GroupName*

X'4125'

PMERR_PRD_MAX_OPTSIZE_EXCEEDED

Explanation: The total size of all print options in the current edit session exceeds the limit of 64K bytes.

Severity: Error

API Verbs: PDSAVD (Save Descriptor)

Response: Use PDREMO (Delete Print Option) to delete print options from the current edit session. For more information, refer to "PDREMO (Delete Print Option) PrdDeleteOption" on page 7-30.

Additional Message Information:

Message ID	Value
X'4125' PMERR_PRD_MAX_OPTSIZE_EXCEEDED	Total size of all print options in the current edit session.

X'4126'

PMERR_PRD_MAX_REFSIZE_EXCEEDED

Explanation: The total size of all print-descriptor references in the current edit session exceeds the limit of 64K bytes.

Severity: Error

API Verbs: PDSAVD (Save Descriptor)

Response: Use PDSREF (Set Reference List) to delete print-descriptor references from the current edit session. For more information, refer to "PDSREF (Set Reference List) PrdSetRefList" on page 7-38.

Additional Message Information:

Message ID	Value
X'4126' PMERR_PRD_MAX_REFSIZE_EXCEEDED	Total size of all print-descriptor references in the current edit session.

X'4127'

PMERR_PRD_MAX_GRPSIZE_EXCEEDED

Explanation: The total size of all group-list entries in the current edit session exceeds the limit of 64K bytes.

Severity: Error

API Verbs: PDSAVD (Save Descriptor)

Response: Use PDSGRP (Set Group List) to delete group-list entries from the current edit session. For more information, refer to "PDSGRP (Set Group List) PrdSetGrpList" on page 7-34.

Additional Message Information:

Message ID	Value
X'4127' PMERR_PRD_MAX_GRPSIZE_EXCEEDED	Total size of all group-list entries in the current edit session.

X'4128'

PMERR_PRD_INV_TYPE

Explanation: The specified print-option default type is not valid.

Severity: Error

API Verbs: PDSOPT (Set Print Option)

Response: Specify a valid print-option default type. For more information, refer to "Print Option Defaults" on page 9-4.

X'4129'

PMERR_PRD_DEFAULT_TRUNCATED

Explanation: For an option of type **PRD_RANGE**, the specified default value has a decimal precision that is greater than the decimal precision of the valid value. Therefore, the default was truncated and set to the truncated value.

Severity: Warning

API Verbs: PDSOPT (Set Print Option)

Response: Either:

- Use the truncated default value
- Respecify the default value with a different precision
- Modify the valid value and respecify the default.

For more information, refer to "Print Option Validation" on page 9-4.

Additional Message Information:

Message ID	Value
X'4129' PMERR_PRD_DEFAULT_TRUNCATED	Option name, truncated default

X'412A'

PMERR_PRD_VALID_VALUE_TRUNCATED

Explanation: For an option of type **PRD_RANGE**, either the minimum or maximum value being set has a decimal precision that is greater than the precision specified for the valid value. Therefore, the minimum or maximum value (or both) was truncated.

Severity: Warning

API Verbs: PDSOPT (Set Print Option)

Response: Either:

- Use the truncated valid value
- Respecify the valid value with a different precision
- Respecify the precision and the valid value.

For more information, refer to "Print Option Validation" on page 9-4.

Additional Message Information:

Message ID	Value
X'412A' PMERR_PRD_VALID_VALUE_TRUNCATED	Option name, truncated valid value(s)

X'4200'

PRTMGR_INV_HAB

Explanation: An invalid anchor block handle was specified on the *hab* parameter or access to the associated storage was denied. This error code will occur only in OS/400.

Severity: Error

API Verbs: PDRDEF (Delete Descriptor), PDEXPD (Export Descriptor), PDIMPD (Import Descriptor), PDLDEF (List Descriptors), PDOPEN (Open Session)

Response: Specify a valid anchor block handle or obtain access to the associated storage. For more information, refer to Chapter 7, API Verb Reference.

Appendix F. IBM SAA PrintManager Operating Environment

PrintManager/400 is available in OS/400 Release 3 and Version 2 Release 1. PrintManager Interface C language bindings are available with Release 3, and PrintManager Interface C, COBOL, and RPG bindings are available with Version 2 Release 1.

Machine Requirements

MVS Machine Requirements

IBM SAA PrintManager is supported on all processors that run under:

- MVS/SP Version 3 Release 1.0/e (MVS/ESA) with JES2
- MVS/ESA SP-JES2 Version 4 Release 1 or higher
- MVS/ESA SP-JES3 Version 4 Release 2 or higher.

VM Machine Requirements

IBM SAA PrintManager is supported on all processors that run under:

- Virtual Machine Facility/System Product (VM/SP) Release 5 or higher
- Virtual Machine Facility/System Product High Performance Option (VM/SP HPO) Release 5 or higher
- Virtual Machine/Extended Architecture (VM/XA) Release 2 or higher
- VM/ESA Release 1.

Programming Requirements

General Programming Requirements

- IBM SAA PrintManager requires the IBM C/370 Library Version 1 Release 2 (5688-039).

  In addition, the PRF menu interactive interface requires the following:

  - ISPF/PDF Version 3 Release 2 (5685-054) for MVS
  - ISPF Version 3 Release 2 (5684-043) for VM.

- Customer applications that use the PrintManager Interface and API must be written in the C programming language, and they will require the IBM C/370 Compiler Version 1 Release 2 (5688-040).
- Distributing print jobs from OS/2 to a host system requires IBM SAA Application Connection Services (5685-117).
- Distributing print jobs between host systems (MVS and VM) requires the following:

  - JES2 Version 3 Release 1 Modification 1 (5685-001) or JES3 Version 4 Release 2 or higher (5695-048).
  - RSCS Version 2 Release 2 (with PTF VM30575), or Version 2 Release 3 (5664-188).

- If Print Services Facility (PSF) is used as the printer driver, one of the following is required:
  - PSF/VM Version 1 Release 3 (5664-198) or higher
  - PSF/MVS Version 1 Release 3 (5665-275) or higher.
- One of the following Graphical Data Display Manager products is required when submitting a print job that contains a MO:DCA-P data stream of PT1 text and DR2 graphics that will be printed on a printer that only supports an AFPDS data stream with PT1 text and IM1 image:
  - GDDM/MVS Version 2 Release 3 (5665-356)
  - GDDM/VM Version 2 Release 2 (5664-200) for VM/SP Release 5
  - GDDM/VMXA Version 2 Release 3 (5684-007) for VM/SP Release 6, VM/XA, or VM/ESA.

Note: In the VM/XA, VM/ESA, and MVS/ESA environments, IBM SAA PrintManager takes full advantage of 31-bit addressing, and can be installed above 16Mb. However, in these environments, IBM SAA PrintManager does not support application programs that operate in 24-bit addressing mode.

MVS Programming Requirements

- IBM System Modification Program Extended (SMP/E) Version 1 Release 5 (5668-949) is required to install IBM SAA PrintManager.
- Before you can install IBM SAA PrintManager, the operating system must be at one of the following levels:
  - MVS/SP Version 3 Release 1.0/e (MVS/ESA) with JES2 Version 3 Release 1 Modification 1 (5685-001)
  - MVS/ESA SP-JES2 Version 4 Release 1 or higher (5695-047)
  - MVS/ESA SP-JES3 Version 4 Release 2 or higher (5695-048).
- The PRF component of IBM SAA PrintManager requires Time Sharing Option/Extensions (TSO/E) Version 2 Release 1 (5685-025).

Notes:

1. IBM SAA PrintManager supports TSO/E and MVS batch job users *only* in the MVS environment.
2. IBM SAA PrintManager in an MVS/ESA system only supports PrintManager applications operating in 31 bit addressing mode.

VM Programming Requirements

Before you can install IBM SAA PrintManager, the operating system must be at one of the following levels:

- VM/SP Release 5 (5664-167), with the Macro Compatibility Enhancement PTF (VM34760) installed.
- VM/SP Release 6 (5664-167).
- VM/SP HPO Release 5 (5664-173), with the Macro Compatibility Enhancement PTF (VM34760) installed.
- VM/SP HPO Release 6 (5664-173).
- VM/XA SP Release 2 (5664-308), with the APSS PTF (VM34943) installed.
- VM/XA SP Release 2 Modification 1 (5664-308)
- VM/ESA Release 1 (5684-112).

Notes:

1. The PRF interactive menus are not supported on VM/SP Release 5 or VM/SP HPO Release 5.
2. IBM SAA PrintManager in VM/XA or VM/ESA systems only supports PrintManager applications operating in 31 bit addressing mode.
3. IBM SAA PrintManager does not support the CMS Shared File System (SFS).

Glossary

Source Identifiers

This publication includes terms and definitions from:

- The *American National Dictionary for Information Processing Systems*, copyright 1982 by the Computer and Business Equipment Manufacturers Association (CBEMA). Copies may be purchased from the American National Standards Institute, 1430 Broadway, New York, New York 10018. Definitions are identified by the symbol (A)[12] after the definition.
- The *Information Technology Vocabulary*, developed by Subcommittee 1, Joint Technical Committee 1, of the International Organization for Standardization and the International Electrotechnical Committee (ISO/IEC JTC1/SC1). Definitions of published segments of the vocabularies are identified by the symbol (I)[12] after the definition; definitions from draft international standards, draft proposals, and working papers in development by the ISO/IEC JTC1/SC1 vocabulary subcommittee are identified by the symbol (T)[12] after the definition, indicating final agreement has not yet been reached among participating members.

References

The following cross-references are used in this glossary:

Deprecated term for Indicates that the term should not be used (because it is obsolete, misleading, ambiguous, or jargonistic) and refers to the preferred term. For a deprecated term, the commentary contains only this reference; the deprecated term is not defined.

Synonymous with Appears in the commentary of a preferred term and identifies less desirable or less specific terms that have the same meaning. The commentaries of the less desirable or less specific terms refer back to the preferred term with the *Synonym for* reference words.

Synonym for Appears in the commentary of a less desirable or less specific term and identifies the preferred term that has the same meaning.

Contrast with Refers to a term that has an opposite or substantively different meaning.

See Refers to a multiple-word term in which this term appears.

See also Refers to related terms that have similar (but not synonymous) meanings.

A

Advanced Function Printing (AFP). A group of IBM licensed programs (or, for OS/400, operating system functions) and printers that uses the all-points-addressable concept to print text and graphics on a page printer.

Advanced Function Printing data stream (AFPDS). A subset of the Mixed Object Content Document Architecture that consists of the structured fields that Print Services Facility accepts.

AFP. Advanced Function Printing.

AFPDS. Advanced Function Printing data stream.

AFP resources. AFP resources consist of form definitions, page definitions, fonts, overlays (electronic forms), and page segments (graphic images). With PrintManager, resources can either exist in a system library, or be placed inline with a print job as the job is written to the spool.

API. Application Programming Interface.

API verbs. A set of programming verbs used to invoke the PrintManager API functions.

Application Programming Interface. A PrintManager component that can be used to create common print descriptors for an entire organization.

B

burst. To separate continuous-forms paper into separate sheets.

C

Common Programming Interface (CPI). The SAA Common Programming Interface.

CPI. Common Programming Interface.

D

dynamic print management. Using PrintManager to make changes to a print operation without interrupting system functions.

E

electronic overlay. An AFP resource object that is a collection of predefined data such as lines, shading, text, boxes, or logos, that can be merged with variable data on a page while printing.

exact-name format. A print-descriptor naming convention that uses system-specific (actual) group names instead of group alias names.

F

font. *1.* A family or assortment of characters of a given size and style; for example, 9 point Bodoni modern.(A)

2. An AFP resource object that defines the fonts for a print job.

form definition. An AFP resource object that defines the characteristics of the printed media, for example: overlays to be used, text suppression, position of page data on the form, and number and modifications of a page.

G

GLPrd. Group-list print descriptor.

group-list print descriptor. A special type of print descriptor used to define print-descriptor groups and the search order used when a print descriptor is referenced.

I

IBM SAA PrintManager. The PrintManager licensed program for the MVS and VM operating systems.

interchange format. A print-descriptor format required to exchange print descriptors between systems.

J

JES. Job Entry Subsystem

Job Entry Subsystem. An MVS subsystem that receives jobs into the system, converts them to internal format, selects them for execution, processes their output, and purges them from the system.

M

Mixed Object Document Content Architecture for Presentation Interchange Set. The presentation subset of an architected, device-independent data stream for interchanging documents.

MO:DCA-P. Mixed Object Document Content Architecture for Presentation Interchange Set.

O

Operating System/2. An IBM licensed program that can be used as the operating system for the PS/2 processor series.

Operating System/400. An IBM licensed program that can be used as the operating system for the AS/400 processor series.

OS/2. Operating System/2.

OS/400. Operating System/400.

overlay. See electronic overlay.

P

page definition. An AFP object resource that contains the formatting controls for line data. It can include controls for number of lines per logical page, font selection, print direction, and mapping individual fields to positions on the logical page.

page segment. An AFP resource object that can contain text and images and can be positioned on any addressable point on a page or an electronic overlay.

PDG profile. A profile used to define your default group-list print descriptor name.

PrdT. Print Descriptor Tool.

PRF. Print Request Facility.

print descriptor. A PrintManager object that is created and maintained with the API. Print descriptors describe where a print job will be printed, how it will be processed, and how the output will appear. These print descriptors contain capabilities and defaults of options used for printing. Multiple print descriptors can exist for different devices and types of print jobs, which allows print jobs to be tailored for different types of applications and routed to the correct destination. Group-list print descriptors are used to define print-descriptor groups and the search order used when a print descriptor is referenced.

print-descriptor group. An object used to store print descriptors so that they can be managed effectively on a system.

Print Descriptor Tool. A PrintManager component, consisting of a set of commands and tags, that is used to invoke the PrintManager API functions.

PrintManager Interface. A PrintManager component that is an element of the SAA PrintManager Interface. The PrintManager Interface can be used to write common applications to send print files to a system spool for printing.

PrintManager. The collective name for a set of IBM programs or operating system functions that provide cross-system print management for an entire organization.

Print Request Facility (PRF). A PrintManager component that provides the casual user with a consistent way to submit print jobs.

Print Services Facility. The collective name for a group of IBM licensed programs (or operating system functions, for OS/400) that produces printer commands for page printers.

PSF. Print Services Facility.

R

relative-name format. A print-descriptor naming convention that uses group alias names instead of system-specific (actual) group names.

Remote Spooling Communication Subsystem. The licensed program that transfers spool files, commands, and messages between VM users, remote stations, and remote and local batch systems through HASP-compatible telecommunication facilities.

RSCS. Remote Spooling Communication Subsystem

S

SAA. Systems Application Architecture.

StdPrd. Standard print descriptor.

Systems Application Architecture. A set of software interfaces, conventions, and protocols that provide a framework for designing and developing applications with cross-system consistency.

system-specific format. A print-descriptor naming convention required to store a print descriptor in a print-descriptor group.

U

UserGLPrdName. System-specific method of referring to a user's GLPrd.

Index

R

S

T

V

Readers' Comments

IBM SAA PrintManager
Application System/400
PrintManager
Application Programming Interface Reference

Publication No. S544-3699-01

Use this form to provide comments about this publication, its organization, or subject matter. Understand that IBM may use the information any way it believes appropriate, without incurring any obligation to you. Your comments will be sent to the author's department for the appropriate action. Comments may be written in your language.

Note: IBM publications are not stocked at the location to which this form is addressed. Direct requests for publications or for assistance in using your IBM system, to your IBM representative or local IBM branch office.

	Yes	No
• Does the publication meet your needs?	☐	☐
• Did you find the information:		
Accurate?	☐	☐
Easy to read and understand?	☐	☐
Easy to retrieve?	☐	☐
Organized for convenient use?	☐	☐
Legible?	☐	☐
Complete?	☐	☐
Well illustrated?	☐	☐
Written for your technical level?	☐	☐
• Do you use this publication:		
As an introduction to the subject?	☐	☐
As a reference manual?	☐	☐
As an instructor in class?	☐	☐
As a student in class?	☐	☐
• What is your occupation?		

Thank you for your input and cooperation.

Note: If mailed in the U.S.A., no postage stamp is necessary. For residents outside the U.S.A., your local IBM office or representative will forward your comments.

Comments:

Name _____ Address _____

Company or Organization _____ _____

Phone No. _____ _____

Readers' Comments
S544-3699-01

IBM®

Cu
Alc

Fold and Tape | **Please do not staple** | Fold and Tape

NO POSTAGE
NECESSARY
IF MAILED IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS MAIL PERMIT NO. 40 ARMONK, NEW YORK

POSTAGE WILL BE PAID BY ADDRESSEE

International Business Machines Corporation
Information Development
Department 588
Facsimile: 303-924-6873
P.O. Box 1900
Boulder, CO 80301-9191

Fold and Tape | **Please do not staple** | Fold and Tape

S544-3699-01

Cut
Alo

File Number: S370-40
Program Number: 5688-179
5738-SS1 8011

Printed in Denmark by
Scanprint a-s, Viby J.

S544-3699-01