Infoprint Server for z/OS
Infoprint Transforms

Infoprint Server Transforms
Customizing transforms and deamons
Using transforms

Paul Rogers
Juha Vainikainen

ibm.com/redbooks
Note: Before using this information and the product it supports, read the information in “Notices” on page ix.

First Edition (August 2008)

This edition applies to Version 1 Release 8 of z/OS (5694-A01), Version 1 Release 8 of z/OS.e (5655-G52), and to all subsequent releases and modifications until otherwise indicated in new editions.
## Contents

1. **Preface** ................................................................. xi
   - The team that wrote this book ................................... xi
   - Become a published author .................................... xi
   - Comments welcome ............................................. xii

2. **Chapter 1. Introduction to Infoprint Transforms** .......................... 1
   - 1.1 Introducing Infoprint Server .................................. 2
   - 1.1.1 Printer Inventory and Printer Inventory Manager .......... 2
   - 1.1.2 IP PrintWay .................................................. 3
   - 1.1.3 Internet Printing Protocol (IPP) ........................... 5
   - 1.1.4 NetSpool ................................................... 5
   - 1.1.5 Print Interface ............................................ 6
   - 1.1.6 Infoprint Central .......................................... 6
   - 1.1.7 Transform interface and Infoprint transforms ............ 7
   - 1.1.8 Infoprint Port Monitor .................................... 7
   - 1.2 Infoprint transforms ........................................... 8
   - 1.2.1 Print data formats ....................................... 8
   - 1.2.2 Transform interface ...................................... 12
   - 1.2.3 Using transforms with Infoprint Server .................. 14
   - 1.2.4 Print data transforms using the Print Interface subsystem . 15
   - 1.2.5 IP Printway data transform processing .................. 17
   - 1.3 Submitting jobs ............................................... 19
   - 1.3.1 Printing with the AOPPRINT JCL procedure ............... 21
   - 1.4 z/OS transforms using commands ............................ 24

3. **Chapter 2. Infoprint Server transforms** .................................. 25
   - 2.1 The Transform Interface ..................................... 26
   - 2.1.1 Transform products ....................................... 26
   - 2.1.2 Software requirements for transforms on z/OS ............ 27
   - 2.2 Infoprint Server environment variables ........................ 28
   - 2.3 Infoprint Server configuration files ........................ 29
   - 2.4 Infoprint Server configuration file (aopd.conf) ................ 31
   - 2.5 Transform configuration file (aopxfd.conf) ................. 32
   - 2.5.1 Create an aopxfd.conf file ................................ 33
   - 2.6 Creating a transform configuration file using ISPF shell .... 35
   - 2.6.1 Create an aopxfd.conf file ................................ 36
   - 2.7 Edit an aopxfd.conf file .................................... 41
   - 2.7.1 Activate the aopxfd.conf file ............................. 43
   - 2.7.2 Infoprint Server transform error messages ............... 44
   - 2.8 Create a remote transform configuration file (aoprx.conf) .... 47
   - 2.8.1 File conversion to AFP format ............................ 47
   - 2.9 Setting up security for transform resources .................. 50
   - 2.9.1 Security for AFP resource libraries ........................ 51
   - 2.9.2 Security for the file ps2afpd ............................. 52
   - 2.10 Customizing transforms ..................................... 56
   - 2.10.1 Using transform classes .................................. 57
4.6 Transforms for XML data streams .................................................. 155
  4.6.1 xml2afp - Transform XML data to AFP data ............................. 155
  4.6.2 xml2pdf - Transform XML data to PDF data .............................. 157
4.7 The AOPBATCH program and the transforms ...................................... 157
  4.7.1 EXEC JCL statement PARM parameters .................................... 157
  4.7.2 AOPBATCH DD statements ..................................................... 159
  4.7.3 AOPBATCH examples ........................................................... 160
  4.7.4 Using AOPBATCH in REXX programs in TSO/E environment ............ 164
4.8 Using the Print Interface subsystem .................................................. 166
  4.8.1 OUTPUT JCL statements and transforms from AFP ......................... 167
4.9 Printing to e-mail addresses .......................................................... 174
  4.9.1 Using IP PrintWay for e-mail destinations ................................ 175
  4.9.2 Infoprint Server sendmail support ......................................... 176
  4.9.3 IP PrintWay and basic mode .................................................. 177
4.10 National languages and document-codepage attribute .......................... 181

Appendix A. Sample REXX programs ....................................................... 185
A.1 XFDERRE REXX exec ................................................................. 186
A.2 ITSOCF30 ISPF edit macro .......................................................... 189
A.3 AFPFIX REXX exec ................................................................. 190
A.4 Instructions for downloading the sample programs ............................... 196

Appendix B. Transform fonts .................................................................. 199
B.1 PDF to AFP and PostScript to AFP fonts .......................................... 200

Appendix C. Transform environment variables .......................................... 203
C.1 Environment variables for PCL to AFP .............................................. 204
C.2 Environment variables for PS and PDF to AFP ................................. 206
C.3 Environment variables for AFP to PCL ............................................. 208
C.4 Environment variables for AFP to PDF ........................................... 214
C.5 Environment variables for AFP to PS .............................................. 223
C.6 Search hierarchies for forms and page definitions ................................ 228
  C.6.1 Search hierarchy for the AFP resource libraries ............................. 230
C.7 AFP architecture limitations for transforms ....................................... 230
  C.7.1 PCL to AFP limitations ........................................................... 230
  C.7.2 PDF to AFP limitations ......................................................... 231
  C.7.3 PS to AFP limitations ............................................................. 231
  C.7.4 Transform limitations ............................................................. 232

Appendix D. IBM presentation architecture .............................................. 233
D.1 Supported objects and controls for AFP data streams ......................... 234
  D.1.1 Objects ................................................................................. 234
  D.1.2 Object content architecture (OCA) ........................................... 235
  D.1.3 Supported objects for AFP transforms ....................................... 235
  D.1.4 AFP resources for transforms .................................................. 236

Appendix E. Transform filters ................................................................. 239
E.1 Filter options for transform filters ................................................... 240
E.2 Filter options for the pcl2afp.dll filter ............................................. 241
E.3 Filter options for the pdf2afp.dll filter ............................................. 242
E.4 Filter options for the ps2afp.dll filter ............................................. 245
E.5 Filter options for the sap2afp.dll filter ............................................. 246
E.6 Filter options for the afp2pcl.dll filter ............................................ 247
E.7 Filter options for the afp2ps.dll filter ............................................. 247
Contents

E.8 Filter options for the afp2pdf.dll filter ................................................................. 247
E.9 Filter options for the xml2afp.dll and xt2afp.dll filters ................................. 250
E.10 Filter options for the aoprform.dll filter ......................................................... 251
E.11 Filter options for the aoprf.so filter ................................................................. 253

Appendix F. Transform UNIX command parameters ................................................ 257
  F.1 Transform commands for z/OS UNIX ............................................................... 258
    F.1.1 z/OS UNIX transform commands to AFP format ...................................... 258
    F.1.2 z/OS UNIX transform commands from AFP format .................................. 260
  F.2 XML transform commands .............................................................................. 263

Related publications ................................................................................................. 265
  IBM Redbooks ........................................................................................................ 265
Other publications .................................................................................................... 265
  Help from IBM ....................................................................................................... 266

Index .......................................................................................................................... 267
Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:
IBM Director of Licensing, IBM Corporation, North Castle Drive, Armonk, NY 10504-1785 U.S.A.

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.
Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. These and other IBM trademarked terms are marked on their first occurrence in this information with the appropriate symbol (® or ™), indicating US registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. A current list of IBM trademarks is available on the Web at http://www.ibm.com/legal/copytrade.shtml

The following terms are trademarks of the International Business Machines Corporation in the United States, other countries, or both:

- AIX®
- BookMaster®
- CICS®
- IBM®
- IMS™
- IP PrintWay™
- Language Environment®
- MVS™
- NetSpool™
- OS/390®
- Print Services Facility™
- PrintWay™
- RACF®
- Redbooks®
- REXX™
- S/390®
- System p™
- Systems Application Architecture®
- VTAM®
- z/OS®
- Redbooks (logo) ®

The following terms are trademarks of other companies:

- Acrobat, Adobe Type Manager, Adobe, PostScript, and Portable Document Format (PDF) are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, other countries, or both.

- ABAP, SAP R/3, SAP, and SAP logos are trademarks or registered trademarks of SAP AG in Germany and in several other countries.

- Internet Explorer, Microsoft, OpenType, Windows Server, Windows, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

- UNIX is a registered trademark of The Open Group in the United States and other countries.

- Linux is a trademark of Linus Torvalds in the United States, other countries, or both.

Other company, product, or service names may be trademarks or service marks of others.
Preface

One component of Infoprint Server is the Transform Interface. This component communicates with transforms that IBM® transform products provide. Transforms convert data from one format to another, for example, from PDF to AFP format, or from AFP to PCL format. Administrators can set up the transforms to automatically transform documents before they are printed. Users can also use the z/OS® UNIX® command line to transform documents, which can then be saved in the converted format and later printed or sent to other users.

This IBM Redbooks® publication presents a comprehensive discussion of the Infoprint Server transforms and provides detailed steps for customizing and using them. It includes the following topics:

- An introduction to Infoprint Transforms
- How to customize the transforms
- How to customize the transform daemons
- How end users use the transforms

The team that wrote this book

This book was produced by a team of specialists from around the world working at the International Technical Support Organization, Poughkeepsie Center.

**Paul Rogers** is a Consulting IT Specialist at the International Technical Support Organization, Poughkeepsie Center. He writes extensively and teaches IBM classes worldwide on various aspects of z/OS, z/OS UNIX, JES3, and Infoprint Server. Before joining the ITSO 20 years ago, Paul worked in the IBM Installation Support Center (ISC) in Greenford, England for eight years, providing OS/390® and JES support for IBM EMEA, and also in the Washington Systems Center for three years. He has worked for IBM for over 40 years.

**Juha Vainikainen** was a Senior I/T Specialist in IBM Finland until his recent retirement. He has extensive experience with all aspects of OS/390 and z/OS and all related subsystems and he has worked with Infoprint Server since OS/390 Release 5. Juha continues to share his vast knowledge about these topics as a consultant for various ITSO documentation projects.

Become a published author

Join us for a two- to six-week residency program! Help write a book dealing with specific products or solutions, while getting hands-on experience with leading-edge technologies. You will have the opportunity to team with IBM technical professionals, Business Partners, and Clients.

Your efforts will help increase product acceptance and customer satisfaction. As a bonus, you will develop a network of contacts in IBM development labs, and increase your productivity and marketability.
Find out more about the residency program, browse the residency index, and apply online at:
ibm.com/redbooks/residencies.html

Comments welcome

Your comments are important to us!

We want our books to be as helpful as possible. Send us your comments about this book or other IBM Redbooks in one of the following ways:

▶ Use the online Contact us review Redbooks form found at:
  ibm.com/redbooks

▶ Send your comments in an e-mail to:
  redbooks@us.ibm.com

▶ Mail your comments to:
  IBM Corporation, International Technical Support Organization
  Dept. HYTD Mail Station P099
  2455 South Road
  Poughkeepsie, NY 12601-5400
This chapter introduces the z/OS Infoprint Server, including the IBM Infoprint transform products. IBM provides several separately priced Infoprint transform products that convert data from one format to another. Some of these products run on the z/OS system, while others run on other systems. The documentation for each transform product describes the supported functions and limitations of the transforms.

Infoprint Server lets you also transform data from one format to another remotely using Infoprint Transform Manager for Linux® (5639-P51) transforms. The Infoprint Transform Manager for Linux is the IBM strategic solution for installations that need to transform large and complex documents.

In addition, Infoprint Server works with several separate IBM transform products. It can automatically transform data streams from one format to another before the data is printed or sent to an e-mail destination. For example, you can use the PDF to AFP transform to convert documents in PDF format to AFP format.

Administrators can set up the transforms to automatically transform documents before they are printed. Users can also use the z/OS UNIX command line to transform documents, which can then be saved in the converted format and later printed or sent to other users.

This chapter covers the following topics:
- An overall description of Infoprint Server
- The Transform Interface components
1.1 Introducing Infoprint Server

Infoprint Server is an optional feature of z/OS that supports printing on local printers and remote printers in a TCP/IP or SNA network. With Infoprint Server, users can submit print requests from remote workstations in a TCP/IP network, from z/OS UNIX System Services applications, from batch applications, from VTAM® applications (such as CICS® and IMS™), and from SAP® R/3 systems. It works together with data stream transforms that Infoprint transform products provide.

Infoprint Server delivers efficient and economical printing with the flexibility for high-volume, high-speed printing from anywhere in the network.

Figure 1-1 shows how most of the components of Infoprint Server fit into your system. Following the figure is a description of the components.

1.1.1 Printer Inventory and Printer Inventory Manager

The Printer Inventory Manager controls the Printer Inventory. The Printer Inventory consists of files in the z/OS UNIX file system (HFS or zFS) that contain information about each defined printer and e-mail destination. The Printer Inventory also contains system configuration information for IP PrintWay™ and Print Services Facility™ (PSF Program Number 5655-M32) for z/OS.

An administrator creates objects in the Printer Inventory. The administrator uses Infoprint Server ISPF panels and the Printer Inventory Definition Utility (PIDU) to create and maintain the Printer Inventory. The PIDU is useful for creating many printer definitions at the same time and for backing up the Printer Inventory.
The Printer Inventory cannot be shared by Infoprint Servers running at the same or different levels on other systems.

**Objects in the Printer Inventory**
The administrator can create these types of objects in the Printer Inventory:

- **Printer definitions**
  Contains information about printers and e-mail destinations.

- **Printer pool definitions**
  Contains information about groups of printer definitions that you want to broadcast data to (applies only to NetSpool™).

- **FSS definitions**
  Contains configuration information for IP PrintWay basic mode functional subsystems (FSSes) and PSF FSSes.

- **FSA definitions**
  Contains configuration information for IP PrintWay basic mode functional subsystem applications (FSAs) and PSF FSAs.

- **Job selection rules**
  Contains rules that IP PrintWay extended mode uses to determine which print jobs to select from the JES spool for printing.

  PSF, as an option, uses printer configuration information that is specified in FSS and FSA definitions in the Printer Inventory. Alternatively the printer configuration information can be specified in the PSF startup procedures and exits. When the printer configuration information is specified in the Printer Inventory:
  
  - The information can be changed without restarting all FSAs in the FSS. Only the FSAs with changed configuration information need to be restarted.
  
  - Operators can use Infoprint Central to work with PSF-controlled printers. For example, operators can start and stop (that is, drain) printers, change JES work-selection criteria, pause or interrupt printers, and turn TCP/IP-connected printers online and offline.

### 1.1.2 IP PrintWay

IP Printway provides fast access to TCP/IP-connected printers and to Virtual Telecommunications Access Method (VTAM)-controlled printers.

The IP PrintWay component transmits output data sets from the JES spool to remote printers or print servers and to e-mail destinations. IP PrintWay can run either in basic mode or in IP PrintWay extended mode:

- **IP PrintWay basic mode**, the original mode of operation, uses the z/OS Functional Subsystem Interface (FSI) to obtain output data sets from the JES spool.

  **Note**: IBM does not plan additional enhancements to IP PrintWay basic mode.

- **IP PrintWay extended mode** uses the z/OS SYSOUT Application Programming Interface (SAPI) to obtain output data sets from the JES spool. It provides better performance, improved usability, and more function than IP PrintWay basic mode.
IP PrintWay protocols

IP PrintWay uses one of the following protocols to transmit output data sets to printers:

LPR
The LPR protocol is a TCP/IP protocol defined by RFC 1179. An LPD that adheres to RFC 1179 must be running in the remote printer or system.

Direct sockets
The direct sockets printing protocol is a TCP/IP protocol in which data is transmitted directly to a designated port. The remote printer or print server must support direct sockets printing.

The term “direct socket printing” refers to printing to network printers that maintain their own queues. To use such a device, one creates a socket, uses it to form a TCP connection to the printer, sends a file to be printed, and then closes the connection. The printer contains a CPU, standard TCP/IP protocols, and a storage device for spooling files. The printer maintains a queue of files waiting to be printed, and can accept additional files while it is printing.

VTAM
Virtual Telecommunications Access Method (VTAM) can be used with IP PrintWay to print on printers that are defined to VTAM as LU type 0, LU type 1, or LU type 3. Supported output data streams are SNA character string (SCS) and data stream compatible/data stream extended (DSC/DSE). Infoprint coaxial printer support for z/OS (5655-N62) is required to print on VTAM-controlled printers.

IPP
Internet printing protocol (IPP) uses the HTTP/1.1 protocol because HTTP/1.1 has the ability to perform multiple transfers over a single TCP connection and because it is the most widely accepted protocol in the Internet marketplace. An IPP Server is running in the remote printer or system.

E-mail
IP PrintWay can use the z/OS UNIX sendmail function to send print output to one or more e-mail addresses. IP PrintWay sends the output, which can be in any data format, as an e-mail attachment.
1.1.3 Internet Printing Protocol (IPP)

The Internet Printing Protocol (IPP) defines a standard protocol for printing as well as managing print jobs, media size, resolution, and so forth. IPP is a standard client-server TCP/IP protocol for printing over the Internet. IPP can be used locally or over the Internet to printers hundreds or thousands of miles away. Unlike other printing protocols, however, IPP also supports access control, authentication, and encryption, making it a capable and secure printing solution from remote workstations that have TCP/IP access. Print Interface can receive print requests from remote workstations that use the Internet Printing Protocol (IPP) or the SMB printing protocol, which is standard on Windows® systems.

The protocol allows the user to:
- Find out about a printer's capabilities
- Submit print jobs to a printer
- Find out the status of a printer or a print job
- Cancel a previously submitted print job

1.1.4 NetSpool

The NetSpool component of Infoprint Server intercepts print data from VTAM applications, such as CICS and IMS; transforms the data streams to EBCDIC line data, PCL, PDF, or other formats that the target printer accepts; and creates output data sets on the JES spool. You can configure NetSpool so that you do not need to change existing VTAM applications. That is, existing VTAM applications can send print requests to NetSpool in the same manner as they currently send print requests to SNA network printers.

VTAM applications establish communication sessions with NetSpool printer logical units (LUs) instead of with SNA-network printers. Each NetSpool printer LU must be defined to VTAM as an application logical unit.

NetSpool processes the following types of VTAM data streams:
- SNA character string (SCS) data over an LU type 1 session.
- 3270 data over an LU type 3 or LU type 0 session.
- A binary data stream over an LU type 0, type 1, or type 3 session. For this type of data, the NetSpool None formatting option in the Printer Inventory must be selected.

NetSpool can transform SCS and 3270 data streams to EBCDIC line data or ASCII PCL data streams without using other transform products. It can also use other Infoprint transform products to convert line data to PDF or PostScript® formats for printing or e-mailing. (NetSpool first transforms SCS and 3270 data to line data, and then uses the transforms).

Note: The AOP_APPLID is an environment variable and is the application program ID that IP PrintWay uses to establish a VTAM session with printers. This ID must match the name of the APPL statement defined to VTAM in the SYS1.VTAMLST data set. This environment variable is required to use IP PrintWay extended mode to print on VTAM-controlled printers.
1.1.5 Print Interface

The Print Interface component processes print requests received from both remote clients and local users. The Infoprint Print Interface receives print requests from clients that run on remote systems, such as Windows and UNIX systems, and directs data to the JES spool. It accepts data in a variety of formats, including the following:

- Advanced function presentation (AFP)
- Plain text, printer control language (PCL)
- Portable document format (PDF)
- PostScript formats

Print Interface protocols
Commands are also provided that let you print from z/OS UNIX System Services. Users submit print requests and query job status from remote clients in the TCP/IP network using TCP/IP protocols. The protocols include:

LPR protocol
The LPD/LPR protocol is a set of programs that provide printer spooling and network print server functionality for UNIX-like systems. The common UNIX printing system (or CUPS), which is more common on modern Linux distributions, borrows heavily from LPD. The LPR protocol is defined by RFC 1179.

Internet printing protocol
IPP is a standard protocol for printing over the Internet. The IPP client runs in the remote system. The Print Interface component implements the IPP server protocol.

Server message block
SMB printing protocol is the standard printing protocol that Windows systems use. The z/OS SMB server is installed on the z/OS system to receive print requests. The z/OS SMB server uses Print Interface callable services to allocate output data sets on the JES spool and return print job status to the client.

Print requests
Users can submit print requests from the local z/OS system using:

- Print Interface subsystem: Using the Print Interface subsystem, you can transform and print output data created by a batch application with minimal changes to your JCL.
- AOPPRINT JCL: Using the AOPPRINT JCL procedure, provided in SYS1.PROCLIB, you can print existing MVS™ data sets and UNIX files. With the AOPPRINT procedure, you can also send files to an e-mail destination instead of a printer.
- z/OS UNIX printing commands (lp, lpstat, and cancel) that Infoprint Server provides: Using these commands, you can print MVS data sets and UNIX files, query the status of a print job, and cancel a print job. You can run these commands from the z/OS UNIX command line or from a UNIX application.
- Infoprint Server SAP Output Management System (OMS): Using the Infoprint Server SAP OMS and the SAP R/3® Application Server for z/OS, SAP R/3 users can submit a print job and receive immediate notification about job events.

1.1.6 Infoprint Central

Infoprint Central, a component of InfoPrint Print Server, is a Web-based print management user interface that allows help desk operators, print administrators, and users to view print jobs, queues, and printers. Depending upon their security authorization, they can hold, move,
cancel, and release jobs; turn printers online and offline; pinpoint network connectivity problems; track Infoprint Central actions by user ID; and perform many other functions for job and printer management.

Infoprint Central requires the z/OS HTTP Server and a Web browser. No applications other than a Web browser need to be installed on users' workstations. Supported browsers include Microsoft® Internet Explorer® 5.5 (and later), Netscape Navigator 7.0 (and later), and IBM Home Page Reader 4.0 (and later).

With Infoprint Central you can:
► Work with print jobs
► Work with printers
► Work with NetSpool logical units
► Display printer definitions
► Check system status

1.1.7 Transform interface and Infoprint transforms

The transform interface component of Infoprint Server communicates with transforms that Infoprint transform products provide. Transforms convert print data from one format to another, for example, from PCL to AFP format and from AFP to PDF format. The Transform Interface can communicate with transforms that run on the z/OS system and on non-z/OS systems.

When processing a print request, Print Interface, NetSpool, and IP PrintWay extended mode invoke the transform interface to transform print data from one format to another when a transform filter is specified in the printer definition.

IBM provides several separately priced Infoprint transform products. Some of these products run on the z/OS system, while others run on other systems. The documentation for each transform product describes the supported functions and limitations of the transforms.

You can run the transforms as standalone commands from z/OS UNIX System Services, or they can run automatically as part of your print job processing.

1.1.8 Infoprint Port Monitor

Infoprint Port Monitor is software that runs on a Windows workstation. It lets you print files on z/OS printers just like you print files on local Windows printers. You can submit the files from any Windows application that has a printing function. After the Infoprint Port Monitor is installed and configured on the Windows system, it automatically sends documents to the Print Interface component of Infoprint Server.

The Infoprint Port Monitor runs on Windows 2000, Windows Server® 2003, and Windows XP. The Infoprint Port Monitor also runs on Windows 98, NT, and Millennium Edition (ME), although IBM no longer supports it on these systems because Microsoft no longer supports these operating systems.

Printing from a Windows system with the server message block (SMB) protocol and the Internet Printing Protocol (IPP) is also supported. To use these protocols, Windows users do not need to install the Infoprint Port Monitor.
1.2 Infoprint transforms

A transform is a program that converts a document (data stream) from one format to another, for example, from PCL to AFP, or PDF to AFP. Infoprint transforms provided by IBM are implemented as dynamic link library (DLL) filters. The Print Interface subsystem can transform data created by a batch application from one data format to another and then write the transformed data to an output data set on the JES spool. The output data set on the JES spool can then be printed on any printer or sent to an e-mail destination.

For a description of the Infoprint Server transform products, see “Transform products” on page 26.

1.2.1 Print data formats

This section describes the different data streams that are supported with Infoprint Server. Infoprint Server can, in many cases, automatically detect the format of the data stream in print jobs. It can then make sure that the selected printer can print the data stream, thus saving paper and time. With an Infoprint transform product, Infoprint Server supports printing data streams such as PCL, PDF, and PostScript on IBM AFP printers. It also supports printing data in AFP format on ASCII printers. Infoprint Server protects your investment in printer hardware while providing you with printing enhancements.

**AFP document format**

Advanced function presentation (AFP) MO:DCA architecture is a device-independent data stream that governs the interchange of documents. Without such an architecture, information exchange is difficult and unpredictable. A subset of MO:DCA is mixed object document content architecture for presentation (MO:DCA-P), which defines presentation documents.

A mixed object document is the collection of data objects that comprise the document's content and the resources and formatting specifications that dictate the processing functions performed on the content.

An MO:DCA-P document can contain a mixture of presentation data objects. Each data object type has unique processing requirements. An object content architecture (OCA) has been established for each IBM data object to define its respective syntax and semantics. MO:DCA-P documents can contain data and data objects governed by these OCAs, as follows:

- Bar code object content architecture (BCOCA) describes and generates bar code symbols.
- Font object content architecture (FOCA) supports the digital presentation of character shapes by defining their attributes, such as shape definitions, shape dimensions, and positioning information.
- Graphics Object Content Architecture (GOCA) represents pictures generated by a computer, commonly referred to as computer graphics.
- Image Object Content Architecture (IOCA) represents image information such as scanned pictures.
- Presentation Text Object Content Architecture (PTOCA) defines text information.

MO:DCA-P documents can also contain or reference some non-OCA data objects that are registered in the MO:DCA-P architecture. Such data objects can be carried in a generic MO:DCA-P object envelope called an object container. Some examples of data objects that can be carried in an object container are image objects in Encapsulated PostScript (EPS).
format, Portable Document Format (PDF) single-page objects, and color mapping tables (CMT).

MO:DCA-P data is composed into pages before it is sent to the printer and includes data placement and presentation information (such as which font to use), along with the data to be printed. The data to be printed is comprised of embedded OCA objects and object containers, and included resources.

Figure 1-3  AFP printing system relationships

A resource is a collection of printing instructions and sometimes data to be printed. A resource is stored in a resource library and can be retrieved by PSF for printing or for Infoprint Server transforms.

Resources in the libraries include:
- Overlays
- Fonts
- Form definitions
- Page definitions
- Page segments, images, and graphics

Methods to generate MO:DCA-P output:
- Use a text-formatting program that generates page data, such as Document Composition Facility (DCF).
- Generate MO:DCA-P data in your application program, using the AFP Toolbox (Program Number 5655-A25). This program product lets you generate MO:DCA-P output from an application without having to have detailed knowledge of the MO:DCA-P data stream and syntax.
- Use one of many products from IBM business partners to generate MO:DCA-P output and, with the provided design tools that are both flexible and easy to use, create personalized, customer-oriented documents.
**HP Printer Command Language (PCL)**

Printer Command Language, commonly referred to as PCL, is a page description language (PDL) developed by HP as a printer protocol that has become a de facto industry standard. HP created PCL to provide an efficient way to control printer features across many different printing devices.

PCL commands are compact escape sequence codes that are embedded in the print job before it is sent to the printer. HP PCL formatters and fonts are designed to quickly translate application output into high-quality, device-specific, rasterprint images.

Printer commands provide an efficient way to control printer features across many different printing devices. There are four general types of HP printer language commands:

- **Control codes**
  A control code is a character that initiates a printer function (for example, carriage return (CR), line feed (LF), form feed (FF), and so forth).

- **PCL commands**
  PCL commands provide access to the printer's PCL control structure. The PCL structure controls all of the printer's features except those used for vector graphics, which are controlled by the HP-GL/2 commands.

  PCL commands (other than single-character control codes) are also referred to as "escape sequences."

  Once a PCL command sets a feature of the printer, that feature remains set until that PCL command is repeated with a new value, or the printer is reset to default.

- **HP-GL/2 commands**
  HP-GL/2 (vector graphic) commands are two letter codes that represent the function of the command (such as IN for initialize). After the two letter mnemonic, there may be one or more parameters that identify details of how to process the command.

- **PJL commands**
  Printer job language (PJL) commands provide job-level control, unlike PCL and HP-GL/2 (which control the placement of dots on the printed page). One of the main features PJL offers is the ability to switch printer languages (personalities) between jobs. Applications supporting PJL can print one job using PCL and then print the next job using PostScript (or another printer language) – without any operator intervention. PJL can also command two-way communications with the printer. PJL can request information from the printer such as printer model, configuration, printer status, and job status. PJL also can be used to change the printer’s control panel settings (change default settings without using the control panel) and modify the message displayed on the control panel.

**PostScript (PS)**

The PS language is a programming language designed to convey a description of virtually any desired page to a printer. It possesses a wide range of graphic operators that can be combined in any manner. It contains variables and allows the combining of operators into more complex procedures and functions.

PostScript page descriptions are programs to be run by an interpreter. PostScript programs are usually generated by application programs. Many PostScript printers also have an interactive state in which the user can program directly in PostScript.

The PostScript language syntax uses reverse Polish notation. Most operators (what other languages term functions) take their arguments from the stack, and place their results onto the stack. Literals (for example, numbers) have the effect of placing a copy of themselves on
the stack. Sophisticated data structures can be built on the array and dictionary types, but
cannot be declared to the type system, which sees them all only as arrays and dictionaries,
so any further typing discipline to be applied to such user-defined types is left to the code that
implements them.

PostScript has a large selection of graphics operators that allow it to precisely describe a
desired page. These operators control the placement of three types of graphics objects:

- Text in a wide variety of typefaces can be placed on a page in any position, orientation,
  and scale.
- Geometric figures can be constructed using PostScript graphics operators. These
describe the locations of straight lines and curves of any size, orientation, and width, as
well as filled spaces of any size, shape, and color.
- Sampled Images of digitized photographs, free-hand sketches, or any other image can be
  placed on a page in any scale or orientation.

All graphic objects can be easily rotated, scaled, and clipped to a specified portion of the
output page.

**Portable document format (PDF)**

PDF is a file format created by Adobe® Systems for desktop publishing. PDF is used for
representing two-dimensional documents in a device-independent and
display-resolution-independent fixed-layout document format. Each PDF file encapsulates a
complete description of a 2D document (and, with the advent of Acrobat® 3D, embedded 3D
documents) that includes the text, fonts, images, and 2D vector graphics that compose the
document.

PDF is primarily the combination of three technologies:

- A sub set of the PostScript page description programming language, for generating the
  layout and graphics.
- A font embedding/replacement system to allow fonts to travel with the documents.
- A structured storage system to bundle these elements and any associated content into a
  single file, with data compression where appropriate.

As a document format, PDF has several advantages over PostScript:

- PDF contains already tokenized and interpreted results of the PostScript source code, so
  there is a more direct correspondence between changes to items in the PDF page
description and changes to the resulting appearance of the page.
- PDF (starting from version 1.4) supports true transparency while PostScript does not.
- PostScript is an imperative programming language (with an implicit global state), so
  instructions accompanying the description of one page can affect the appearance of any
  following page. All the preceding pages must therefore be processed in order to determine
  the correct appearance of any given page. Each page in a PDF document is unaffected by
  any others.

**Extensible markup language (XML)**

XML provides a standard method for the exchange and processing of data across different
computing platforms. XML data is structured according to standards from the World Wide
Web Consortium (W3C). XML differs from HTML. While HTML contains information about
how data is formatted for presentation on Web pages, XML contains data with no
presentation information. A W3C standard stylesheet language, called extensible stylesheet
language (XSL), defines the formatting objects that describe how XML data is presented.
XML data containing XSL formatting objects is called XSL-FO. XSL-FO describes document presentation details, such as pagination, layout, and styling information.

XML incorporates many of the features of HTML, but it also addresses some of the limitations of HTML. The syntax of XML has more restrictions than HTML, which results in faster and cheaper browsing.

XML is used for publishing as well as for data storage and retrieval, data interchange between heterogeneous platforms, data transformations, and data displays.

**Line data**
A data format whose bytes map to characters. Line data is stored as records, for example, in sequential data sets. The records can contain carriage-control characters and table-reference characters. Line data is typically found in mainframe data sets.

**Text**
A data format whose bytes map to characters. Text data contains no control characters other than line feed (LF), carriage return (CR), horizontal tab (HT), vertical tab (VT), and form feed (FF). Text data is typically found in workstation files.

**SAP**
SAP Output Text Format (OTF) or SAP Advanced Business Application Programming (ABAP™) Version 1 or Version 2 data format, defined by SAP AG.

**JPEG**
Joint photographic experts group (JPEG) file information format (JFIF). JPEG/JFIF format is most used for storing and transmitting photographs on the World Wide Web.

JPEG is a commonly used method of compression for photographic images. The JPEG standard specifies both the code, which defines how an image is compressed into a stream of bytes and decompressed back into an image, and the file format used to contain that stream.

**TIFF**
Tagged image file format (TIFF) is a container format for storing images, including photographs and line art. It was originally created for use with what was called “desktop publishing.” The TIFF format is widely supported by image-manipulation applications, by publishing and page layout applications, and by scanning, faxing, word processing, optical character recognition and other applications.

Adobe Systems holds the copyright to the TIFF specification.

### 1.2.2 Transform interface

The Infoprint Server transform interface component communicates with transforms that Infoprint transform products provide. Transforms convert data from one format to another, for example, from PCL to AFP format and from AFP to PDF format.

Infoprint Server transforms are implemented as filters that you associate with the data formats. To request that Infoprint Server invoke a filter, the filter name and filter options are specified in the printer definition (in the Printer Inventory) for a printer.
The transform interface communicates with transforms that run on the z/OS system and on non-z/OS systems. Figure 1-4 shows how the transform interface fits into your system. An explanation of the steps follows the figure.

1. When processing a print request using definitions in the Printer Inventory, the Print Interface subsystem, NetSpool, and IP PrintWay extended mode each can call the transform interface to transform print data from one format to another if the administrator specifies a transform filter in the printer definition. A printer definition can specify a different transform filter for each data format supported by the printer. When a printer definition specifies a transform filter for a data format, Print Interface automatically calls the specified transform before it writes data to the JES spool.

When NetSpool intercepts print data from the VTAM logical unit defined to a printer, it transforms the print data as requested in the printer definition before the data is written to the JES spool.

IP PrintWay extended mode calls transforms directly, while IP PrintWay basic mode transforms data by resubmitting it to Print Interface. Calling transforms directly is more efficient because data is not written to the JES spool a second time.

Infoprint Server and Infoprint transforms provide the following transform filters:

- The filter DLL or UNIX filter used for documents that contain MO:DCA (AFP) data.
- A filter that can transform data from one format to another.
- The filter DLL or UNIX filter used for documents that contain PostScript data.

2. z/OS job submitters can use the following z/OS UNIX commands to transform data without printing it:

- `remotexf` command
  
  The `remotexf` command sends data to Infoprint Transform Manager for Linux for a transform to take place.

- An Infoprint transform command
  
  Some Infoprint transform products provide z/OS UNIX transform commands. For example, Infoprint Transform for AFP to Adobe PDF for z/OS provides the `afp2pdf` command, which transforms data from AFP to PDF format.
3. Transform Interface communicates with the transform, as follows:
   - If the transform runs on z/OS, the Infoprint Server Transform Manager (part of Transform Interface) starts and stops the transform daemons, using configuration information specified by the administrator. (The Infoprint Server Transform Manager does not start the SAP to AFP transform because this transform is not implemented as a daemon.)

   **daemon**: A process that runs in the background and performs a specified operation at predefined times or in response to certain events.

   - If the transform runs on a non-z/OS system, the transform interface sends the data to the transform.

4. The transform converts the data from one format to another.

**Filters**

A filter can transform data from one format to another.

- **aoprxf.so**: This filter sends data to Infoprint Transform Manager for Linux to be transformed.

  **Note**: Executables that have the .so suffix in their file names are called *user shared library programs*.

- **aoprform.dll**: This filter sends data to Infoprint Manager for AIX® or Windows for transform.

- **An Infoprint transform filter**: Some Infoprint transform products provide transform filters. For example, Infoprint Transform for AFP to Adobe PDF for z/OS provides the afp2pdf.dll filter, which transforms data from AFP to PDF format.

  **Note**: IP PrintWay basic mode transforms data by resubmitting it to the Print Interface component. This function is called the resubmit for filtering function.

**1.2.3 Using transforms with Infoprint Server**

The Infoprint Server transforms can be invoked by the following Infoprint Server components and features:

- **Print Interface subsystem**: The Print Interface subsystem transforms data before writing it to an output data set on the JES spool, when the printer...
definition in the request specifies filtering for the printed data format. IP PrintWay or PSF can then print the data, or IP PrintWay can send it to an e-mail destination.

To use the Print Interface subsystem, code the SUBSYS JCL parameter to request that the Print Interface subsystem process a print data set.

**IP PrintWay extended mode**

IP PrintWay extended mode transforms data in a spool data set before the data is send to a printer or to an e-mail destination.

**IP PrintWay basic mode**

IP PrintWay basic mode sends data in a spool data set to Print Interface. Print Interface transforms the data and writes the transformed data to a new output data set on the JES spool. IP PrintWay then prints the transformed data or sends it to an e-mail destination. As shown in Figure 1-8 on page 19, the **Resubmit for filtering** indicator in a printer definition controls the IP PrintWay basic mode transformation (filter) processing.

**AOPPRINT JCL procedure**

The AOPPRINT JCL procedure uses the enhanced z/OS UNIX `lp` command to invoke Print Interface to transform data in an existing MVS data set or UNIX file and to write the transformed data to an output data set on the JES spool. The output data set on the JES spool can be printed on any printer or sent to an e-mail destination.

**Transform commands**

The z/OS UNIX transform commands, which can be run using the AOPBATCH program or UNIX shell, transform data in existing MVS data sets or UNIX files. The transform commands write the transformed output to an MVS data set or UNIX file but do not write the data set to an output data set on the JES spool for printing.

### 1.2.4 Print data transforms using the Print Interface subsystem

A z/OS job submitter specifies the SUBSYS JCL parameter on the output DD JCL statement to request that the Print Interface subsystem process the output data created by a batch application.

The Print Interface subsystem invokes transforms before writing the data to the JES spool when a printer definition in a print request specifies filtering for the data format.

**SUBSYS JCL parameter**

The format of the SUBSYS JCL parameter is:

```
SUBSYS=(subsystem_name[,['printer_definition_name'][,'attribute=value ...']])
```

The positional subparameters are as follows:

- `subsystem_name` Specifies the name of the Print Interface subsystem. This name must be the same as the Inventory name that is specified in the Infoprint Server configuration file. The Printer Inventory name is usually AOP1.
  
Default: None.

- `printer_definition_name` Specifies the name of the printer definition in the Printer Inventory that the subsystem uses to process the data set. This
name is case-sensitive. If the name includes any lower case characters or special characters, enclose the name in single quotation marks. If you want to omit this parameter, but still specify job attributes, code a comma to indicate the printer definition name is omitted.

Default: The first printer definition name found, using this order:

a. The printer definition name specified in the FSSDATA parameter on the OUTPUT JCL statement.

b. The Infoprint Server default printer definition. The name of the Infoprint Server default printer definition is specified in the Printer Inventory. The default name is lp1.

attribute=value Specifies Infoprint Server job attributes that the subsystem uses to process and print the data set.

**Note:** If you want to specify more than 120 characters of job attributes, instead of job attributes use the corresponding JCL parameter. For example, specify the COPIES= JCL parameter instead of the copies= job attribute. Or, specify the job attributes in an attributes file.

**Print transform example**

In the example shown in Figure 1-5, the Print Interface subsystem, named AOP1, transforms the input line data to PDF format and writes the data to the JES spool. IP PrintWay then sends the spooled data to the e-mail printer which is specified on the OUTPUT JCL statement and shows an example of SUBSYS parameter for the Print Interface subsystem.

```
//USERJOB JOB (999,POK),EXPERT,MSGLEVEL=1,MSGCLASS=T,CLASS=A
//EX EXEC  PGM=IEBGENER
//SYSPRINT DD SYSOUT=T
//EMAIL OUTPUT DEST=EMAIL,FORMS=STD,
// MAILFROM='nn',MAILTO='vaini@fi.ibm.com',MAILFILE=FYI,
// TITLE='Mail from POK', NOTIFY=&SYSUID
//SYSUT2 DD SUBSYS=(AOP1,'EVAINI','duplex=yes copies=2'),
// OUTPUT=(*.EMAIL)
//SYSIN DD DUMMY
//SYSUT1 DD *
/*
```

*Figure 1-5  Print JCL example using the SUBSYS parameter*

Figure 1-6 shows the line data transform filter specification on the Infoprint Server: Printer Inventory Manager Processing ISPF panel for the EMAIL printer.
Figure 1-6   An EMAIL printer’s ISPF Processing panel filter definition

Figure 1-7 shows the help text for a DLL or UNIX filter definition used for documents that contain line data. Each data format that can be selected on the IP PrintWay Printer Definition Processing panel has its own help text.

HELP
The filter DLL or UNIX filter used for documents that contain line data.
A filter can transform data from one format to another. If you install an IBM transform product, you can specify these filters that transform line data formatted with an AFP page definition:

afp2pcl.dll Transforms line data to PCL format
afp2pdf.dll Transforms line data to PDF format
afp2ps.dll Transforms line data to PostScript format

For filter options, see "Infoprint Transforms from AFP for z/OS".
Format: Filter name followed by options. Specify the full path name of the filter unless the filter is in a directory named in the LIBPATH (filter DLLs) or PATH (UNIX filters) environment variable.

Example: afp2pdf.dll -c eu %filter-options

Figure 1-7   ISPF panel Help for the Filter Field for Line Data Format

1.2.5  IP Printway data transform processing

IP PrintWay extended mode transforms data in an output data set from one data format to another and then sends it to a remote printer or to an e-mail destination. IP PrintWay extended mode, in most cases, automatically detects the input data format and transforms data into the format required by the printer or e-mail destination.

IP PrintWay basic mode transforms data into the format required by the printer or e-mail destination by resubmitting the data stream to Print Interface. The resubmit for filtering is requested for a printer on the IP PrintWay Printer Definition Processing panel (Figure 1-8 on page 19).
You do not need to specify any special JCL parameters to request the IP PrintWay transform functions. When you print a data set, you can specify the same JCL parameters as you usually do when you print on a PSF-controlled printer. However, you must direct the data set to IP PrintWay instead of to the PSF printer.

To direct an output data set to IP PrintWay, you specify on the SYSOUT DD or OUTPUT JCL statement for the data set:

- The work-selection criteria that your administrator defined to JES for the IP PrintWay basic mode functional subsystem application (FSA)
- The job-selection criteria that your administrator defined in the Printer Inventory for IP PrintWay extended mode

Figure 1-8 on page 19 shows, on the IP PrintWay Printer Definition Processing panel, transform filter names and some filter options for line data and MO:DCA-P data formats. For a transform filter you can specify in the Filter field of a printer definition:

**-c transformclass**

Specifies the name of a transform class that is defined in the aopxfd.conf transform configuration file. The name is case-sensitive.

The transform configuration file contains information that the Infoprint Server Transform Manager uses to manage transforms that run on the z/OS system. In the aopxfd.conf file, you create a separate entry for each transform class. The sample transform configuration file (/usr/lpp/Printsrv/samples/aopxfd.conf) defines several transform classes.

If you want to specify different environment variables for the same transform for different printing situations, you can create different classes of the transform. For example, if you want to transform PCL data to AFP format for printers that have different resolutions or paper sizes, you can define a transform class for each combination of printer resolution and paper size.

**%filter-options**

The %filter-options specifications cause options to be passed to the transform. Figure 1-10 on page 20 is a complete list of Infoprint Server job attributes.
1.3 Submitting jobs

A job submitter can also specify the transform class in the filter options job attribute. You specify job attributes in different ways, depending on the method you use to submit a print job, as shown in Figure 1-9.

Some job submission methods (for example, the LPR command) do not support specification of job attributes or JCL parameters. You can use the PRTATTRS parameter on OUTPUT JCL statements to specify Infoprint Server job attributes that do not have corresponding JCL parameters for, for example, document-codepage or document-format.
This example shows how to direct an output data set to IP PrintWay and to a printer by specifying the printer definition name in the Printer Inventory. The PRTATTRS parameter on the DEST=IP1145 OUTPUT JCL statement requests five copies to be printed using a GT24 coded font. IP Printway transforms the line data dataset to PCL before it is sent to the IP1145 printer. Refer to Figure 1-8 on page 19 for the transformation filter specification.

Note: The attributes listed show attribute names and values in their complete form. Often, you can abbreviate attribute names and values by using the first letter of each word in the name or value. For example, you can use the abbreviation d-f-s for the document-formats-supported attribute. You can use po for the postscript value, and specify the attribute and value pair as d-f-s={po}. Sometimes specifying only the first letter in each word is ambiguous. For example, c might stand for class or copies. Specify enough of the name so that it is unique, as in cl or co.

JCL example
This example shows how to direct an output data set to IP PrintWay and to a printer by specifying the printer definition name in the Printer Inventory. The PRTATTRS parameter on the DEST=IP1145 OUTPUT JCL statement requests five copies to be printed using a GT24 coded font. IP Printway transforms the line data dataset to PCL before it is sent to the IP1145 printer. Refer to Figure 1-8 on page 19 for the transformation filter specification.

//USERZX JOB (999,POK),EXPERT,MSGLEVEL=1,MSGCLASS=T,CLASS=A
//E1 EXEC PGM=IEBGENER
//IP1145 OUTPUT DEST=IP1145,PRTATTRS='chars=GT24 copies=5'
//SYSUT2 DD SYSOUT=F,OUTPUT=(*.IP1145)
//SYSPRINT DD DUMMY
//SYSPIN DD DUMMY
//SYSUT1 DD *

Data changes cannot be saved in a View session. Use Edit if you want to be able to save your changes.
1.3.1 Printing with the AOPPRINT JCL procedure

The AOPPRINT JCL procedure, provided in SYS1.PROCLIB, lets you submit print requests from z/OS. It lets you to take advantage of all the features of Infoprint Server, including:

- You can specify job attributes.
- If an IBM Infoprint transform product is installed, your document to be printed is automatically transformed to the data format supported by the target printer.
- Infoprint Server validates that data can print on the selected printer.

Figure 1-12 shows the AOPPRINT JCL procedure.

```
// *-------------------------------------------------------------------
// | AOPPRINT - OS/390 Print Server batch print procedure           |
// *-------------------------------------------------------------------
//AOPPRINT PROC PRINTER='p1',OPTIONS=,OUTCLASS='*',ERRCLASS='*'     
//LP       EXEC PGM=AOPBATCH,                                       
// PARM='/lp -d &PRINTER -o "&OPTIONS" //DD:SYSIN'                  
//STDOUT   DD   SYSOUT=&OUTCLASS                                   
//STDERR   DD   SYSOUT=&ERRCLASS                                   
//* STDENV may point to a dataset containing environment variables.
//* Builtin values will work for the default installation.         
//* *STDENV   DD   DSN=USERID.JCL(ENVVARS),DISP=SHR                
```

Figure 1-12  AOPPRINT JCL procedure

**Note:** The length of the subparameters in the PARM= specification must not exceed 100 characters:

- *Including* any commas, which are passed to the processing program.
- *Excluding* any enclosing parentheses or apostrophes, which are not passed.

When you code more than one subparameter, separate the subparameters by commas and enclose the subparameters in parentheses or apostrophes. For example:

```
PARM='chars=GT24 copies=5'
```

**Note:** To continue a parameter that is enclosed in apostrophes:

1. Extend the parameter to column 71. Do not code an apostrophe in column 71 of a JCL statement that is continued. The system interprets the apostrophe in column 71 as the final character in the statement and ignores the continuation.
2. Code // in columns 1 and 2 of the following statement.
3. Continue the parameter in column 16 of the following statement even if this splits the parameter. Trailing blanks or commas within the apostrophes do not indicate a continued statement; the system treats them as part of the parameter.

The following example shows the specification of a long PARM= field:

```
//LP EXEC PGM=AOPBATCH,                                          
// PARM='/lp -d IP1145 -o "chars=GT10 copies=10 document-codepage=IBM-
// M-1047 duplex=yes" //DD:SYSIN'
```
AOPPRINT procedure parameters

Following are the AOPPRINT JCL procedure parameters:

ERRCLASS=class

Specifies the one character alphanumeric name of the JES SYSOUT class for error messages. The name of the data set where error messages are written is specified by the STDERR data definition name (DDname).

Default: Infoprint Server uses the class specified by your administrator.

OPTIONS='attribute=value...' Specifying job attributes and values to use in processing the job. Enter the attribute names in lower case. Separate attributes with a space.

By using an attributes file, you can store attributes and values in a UNIX file (such as an HFS file) or in an MVS data set. Use the attribute called attributes to specify the file or DD name from which attributes are to be read, as in this example:

OPTIONS='attributes=bigjob.att'

If the attributes are stored in an MVS data set, you must specify four single quotation marks before the data set name and four single quotation marks after it.

OUTCLASS=class

Specifies the one character alphanumeric name of the JES SYSOUT class for informational messages. The name of the data set where informational messages are written is specified by the STDOUT DDname.

Default: Infoprint Server uses the class specified by your administrator.

AOPPRINT JCL DD statements

Following are the possible DD statements that are specified in the AOPPRINT JCL procedure. (Notice that you cannot specify an OUTPUT JCL statement; if you specify one, it is ignored.)

STDERR

Specifies the system output data set where error messages are to be written. The ERRCLASS parameter defines the class of this data set.

STDOUT

Specifies the system output data set where informational messages are to be written. The OUTCLASS parameter defines the class of this data set.

SYSIN

Specifies the data set to be printed. You can concatenate data sets that have the same data format; for example, line data or AFP data. However, you cannot concatenate data sets that contain PDF data.

AOPPRINT examples

After Infoprint Server accepts the print job, AOPPRINT returns an Infoprint Server job ID, which you can use to query and cancel the job. The Infoprint Server job ID is returned in a message in the STDOUT data set. For example:

AOP007I Job 509 successfully spooled to IP1145.

Figure 1-13 shows the JCL for printing with the AOPPRINT JCL procedure to a printer.
With the AOPPRINT JCL procedure, you can send the file to an e-mail destination instead of to a printer. Figure 1-15 sends a file to an e-mail destination. This example assumes that:

- The administrator has set up a printer definition using the EMAIL Protocol panel in the Printer Inventory, as shown in Figure 1-14.
- The e-mail protocol is selected and an e-mail address is specified in the printer definition mail for the primary recipient. However, the e-mail address you specify in the mail-to-addresses job attribute overrides the e-mail address in the printer definition.

- The AFP to PDF transforms are specified for line data and MO:DCA-P data formats in the EMAIL printer definition.
- AFP resources libraries that the transforms use are specified either in the printer definition or in the aopxfd.conf transform configuration. If the AFP resource libraries are not specified elsewhere, you must specify the libraries in the resource-library job attribute.
1.4 z/OS transforms using commands

z/OS UNIX System Services users can use transform commands to convert document data formats. You can run the transform commands from the z/OS UNIX command line, or you can use the Infoprint Server AOPBATCH program to run them. The IBM to/from AFP transforms include:

- **IBM Infoprint Transforms to AFP for z/OS** (program number 5655-N60). The names of the transforms to AFP and the levels of data streams that they transform are shown in Figure 1-16. These data-stream transforms let you transform documents to AFP format from other data stream formats and print non-AFP data on IBM AFP printers, also known as Intelligent Printer Data Stream (IPDS) printers.

<table>
<thead>
<tr>
<th>Transform:</th>
<th>Transforms to AFP from levels:</th>
<th>Command:</th>
</tr>
</thead>
<tbody>
<tr>
<td>PCL to AFP</td>
<td>PCL 6 (XL, 5, 5e, 5c)</td>
<td>pcl2afp</td>
</tr>
<tr>
<td>PDF to AFP</td>
<td>PDF 1.4</td>
<td>pdf2afp</td>
</tr>
<tr>
<td>PostScript to AFP</td>
<td>PostScript Language Level 3</td>
<td>ps2afp</td>
</tr>
<tr>
<td>SAP to AFP</td>
<td>SAP R/3 SAPGOF Release</td>
<td>sap2afp</td>
</tr>
</tbody>
</table>

  *Figure 1-16  Infoprint Transform z/OS UNIX commands to convert text to AFP format*

- The following transforms convert AFP data streams to the specified formats:
  - **IBM Infoprint Transform for AFP to HP PCL for z/OS** (program number 5655-P19)
  - **IBM Infoprint Transform for AFP to Adobe PDF for z/OS** (program number 5655-P20)
  - **IBM Infoprint Transform for AFP to Adobe PostScript for z/OS** (program number 5655-P21)

  The names of the transform commands from AFP and the levels of data streams that they create are shown in Figure 1-17.

<table>
<thead>
<tr>
<th>Transform:</th>
<th>Transforms AFP to levels:</th>
<th>Command:</th>
</tr>
</thead>
<tbody>
<tr>
<td>AFP to PCL</td>
<td>PCL 5, 5e, or 5c</td>
<td>afp2pcl</td>
</tr>
<tr>
<td>AFP to PDF</td>
<td>PDF 1.4</td>
<td>afp2pdf</td>
</tr>
<tr>
<td>AFP to PostScript</td>
<td>PostScript Language Level 3</td>
<td>afp2ps</td>
</tr>
</tbody>
</table>

  *Figure 1-17  z/OS UNIX commands to transform AFP data streams*

These three products provide data-stream transforms that transform documents from Advanced Function Presentation (AFP) format to other formats that can be printed on non-AFP printers or viewed with Adobe Acrobat.

- **IBM Infoprint XML Extender for z/OS**

  The Infoprint XML Extender for z/OS (referred to as XML Extender) is an IBM licensed program that transforms Extensible Markup Language (XML) data to AFP or PDF format:

  - AFP for viewing, archiving, further transformation, or printing. You can specify the transform with a batch JCL procedure (XML2AFP) or you can use the z/OS UNIX System Services xml2afp command.

  - PDF for viewing, archiving, or e-mailing. You can specify the transform with a batch JCL procedure (CML2PDF) or you can use the z/OS UNIX System Services xml2pdf command.
Infoprint Server transforms

The Transform Interface communicates with Infoprint transform products to transform data to other data formats. It manages most of the Infoprint transforms that run on z/OS, and it sends data to Infoprint transforms that run on other systems, including Linux, AIX, and Windows systems. This chapter describes how to work with the Transform Interface component of Infoprint Server and some transform products.

The Transform Interface component of Infoprint Server communicates with transforms that Infoprint transform products provide. Transforms convert data from one format to another, for example, from PCL to AFP format and from AFP to PDF format. The Transform Interface can communicate with transforms that run on the z/OS system and on non-z/OS systems.

Print Interface can automatically transform data to and from the AFP data format if you have installed Infoprint transform products or another optional transform product, such as IBM Infoprint XML Extender for z/OS (5655-J66) or IBM Infoprint XT Extender for z/OS (5655-J65). Print Interface can also automatically transform data to the AFP format remotely on an AIX or Windows system using transforms that Infoprint Manager for AIX (5785-E42) and Infoprint Manager for Windows (5639-127) provide.
2.1 The Transform Interface

The Transform Interface communicates with transforms that IBM transform products provide. Transforms convert data from one format to another, for example, from PDF to AFP format, or from AFP to PCL format. The IBM Infoprint transform products are all separately priced products and require customization when using them. Purchase only those transforms that you will use.

2.1.1 Transform products

These IBM Infoprint transform products work with Infoprint Server. You can run the transforms as standalone commands from z/OS UNIX System Services, or you can run them automatically when you submit a job to print. The transform products described in this section work with the z/OS Infoprint Server.

**Infoprint Transforms to AFP for z/OS (5655-N60)**
This product transforms documents to Advanced Function Presentation (AFP) format from HP Printer Control Language (PCL), Adobe Portable Document Format (PDF), Adobe PostScript, or SAP R/3 System Generic Output Format (SAPGOF).

**Infoprint Transform for AFP to HP PCL for z/OS (5655-P19)**
This product transforms documents from AFP format to PCL.

**Infoprint Transform for AFP to Adobe PDF for z/OS (5655-P20)**
This transform transforms documents from AFP format to PDF.

**Infoprint Transform for AFP to Adobe PostScript for z/OS (5655-P21)**
This product transforms documents from AFP format to PostScript.

**Infoprint Coaxial Printer Support for z/OS (5655-N62)**
This product transforms line data to Data Stream Compatibility/Data Stream Extended (DSC/DSE) or SNA Character String (SCS). IP PrintWay requires this transform to print on VTAM-controlled printers in an SNA network.

**Infoprint XML Extender for z/OS (5655-J66)**
This product transforms Extensible Markup Language (XML) files to AFP or PDF format for printing or e-mailing. To transform XML files to PCL or PostScript, you can transform XML to AFP, then transform AFP to PCL or PostScript.

**Infoprint XT Extender for z/OS (5655-J65)**
This product transforms Xerox files to AFP format for printing or e-mailing. The Xerox files can be line-conditioned data streams (LCDS) or metacode data streams.

**Infoprint Transform Manager for Linux (5639-P51)**
This product is a cost-effective, Linux-based (PDF, PS, PCL, GIF, TIFF and JPEG to AFP) transform offering. It works with InfoPrint Server for z/OS by allowing compute-intensive transforms to be off-loaded from the z/OS print server to a less expensive, dedicated server.
On the z/OS system the `remotexf` command can be used to send files to Infoprint Transform Manager for Linux (without printing them). A printer can also be defined to specify the `aoprxf.so` filter in printer definition to automatically transform files before they are printed.

Infoprint Transform Manager for Linux can return the transformed files and spool them for printing on AFP printers.

**Infoprint Manager for AIX (5785-E42)**

This product is a flexible and scalable print management solution on IBM System p™, providing many choices of how to expand and manage your print environment. This product transforms PCL to AFP, PDF to AFP, and PostScript to AFP. It provides several options for printing to or from host systems, depending on how much functionality you need and where you want to control printing from (the host system or Infoprint Manager):

- **MVS download**
  
  The Infoprint MVS download daemon allows MVS users to print data sets from the MVS JES spool on Infoprint Manager for AIX-managed printer devices.
  
  The MVS Download receiver/daemon in Infoprint Manager for AIX supports both Download for z/OS and AFP Download Plus for z/OS. If you use the MVS Download feature of PSF for z/OS or the AFP Download Plus feature of PSF for z/OS, you can spool jobs to remote locations, so that the remote print operator can control printing.

- **Infoprint Server on z/OS**
  
  If you use Infoprint Server on z/OS, you can submit jobs on the host for spooling to a remote Infoprint Manager server. The IP PrintWay component of Infoprint Server, which contains an LPR command, and the Infoprint Manager LPD must be running. The remote print operator controls printing after Infoprint Manager receives the job.
  
  On the z/OS system, the `aoprxform.dll` filter can be used to send data to Infoprint Manager for AIX, or Windows, to be transformed.

- **LPR**
  
  Remote systems that support LPR (RFC 1179) can also submit output to Infoprint Manager for AIX through its LPD gateway using the TCP/IP protocol.

- **AFP Upload**
  
  AFP Upload consists of an AIX client program (as part of Infoprint Manager) and an MVS or host server program (available as a priced feature of PSF/MVS). The AIX client program accepts AFP files or files it can transform to AFP and sends them to the MVS server, which places them on the JES spool.

**Infoprint Manager for Windows (5639-I27)**

If you use Infoprint Server on z/OS, you can submit jobs on the host for spooling to a remote Infoprint Manager server. The IP PrintWay component of Infoprint Server, which contains an LPR, and the Infoprint Manager LPD must be running. (If you run the Infoprint Manager Server on a Windows system, the Infoprint Manager LPD replaces the Microsoft LPD for Windows.) This product transforms PCL to AFP, PDF to AFP, and PostScript to AFP.

The remote print operator controls printing after Infoprint Manager receives the job.

### 2.1.2 Software requirements for transforms on z/OS

The transforms to AFP require the following IBM software:

- A current release of z/OS.
The transforms from AFP require following IBM software:

- A current release of z/OS.
- z/OS Infoprint Server.
- AFP Font Collection Version 1 (5648-113) or Version 2 (5648-B33), which contains 300-pel raster fonts and outline fonts. (If your installation has only 240-pel font libraries, you can use the AOXCF30 program to scale 240-pel single-byte or double-byte fonts to 300-pel fonts.)
- Sonoran Equivalent Fonts PRPQ 8A5061 (5799-FLK) if documents contain Sonoran Serif or Sonoran Sans Serif fonts.
- An AFP page definition, form definition, and font for formatting transform error messages.
- AFP resources that are not included in line in the AFP documents.
- AFP to PDF transform.

Note: The Open Cryptographic Services Facility (OCSF), with the Security Level 3 Feature installed, is required to encrypt PDF documents. OCSF is part of z/OS Cryptographic Services. The use of 128-bit encryption is export-controlled by the United States Government.

Adobe Acrobat 5.0 (or higher), or a comparable PDF viewer, is required to view the PDF documents.

IBM Infoprint XML Extender for z/OS
The IBM Infoprint XML Extender for z/OS runs on OS/390 V2R10 or any release of z/OS or z/OS.e. It can be installed as a managed transform with the Infoprint Server feature of OS/390 or z/OS. It can also be installed and used without the Infoprint Server feature of OS/390 or z/OS.

IBM Infoprint XT Extender for z/OS
The IBM Infoprint XT Extender for z/OS can be installed on all z/OS releases.

2.2 Infoprint Server environment variables

The following environment variables affect the behavior of the Transform Interface:

**AOPRXF_CONF**

The full path name of the remote transform configuration file, aoprxf.conf. This environment variable is optional. If you did not create this configuration file, or if you created it in the default location, you do not need to set it.

Default: /etc/Printsrv/aoprxf.conf

**AOPXFD_CONF**

The full path name of the transform configuration file, aopxfd.conf. This environment variable is optional. If you did not create this configuration file, or if you created it in the default location, you do not need to set it.

Default: /etc/Printsrv/aopxfd.conf
Chapter 2. Infoprint Server transforms

2.3 Infoprint Server configuration files

You can use the sample job provided in SYS1.SAMPLIB(AOPCPETC) to copy all Infoprint Server sample configuration files, including the transform configuration file, from /usr/lpp/Printsrv/samples to the /etc/Printsrv directory, as shown in Figure 2-2 on page 30. Review the comments in the sample job to determine which configuration files your installation requires, and then run the job to create the files. A user with an effective UID of 0 should run this job. For using Infoprint transforms, the aopxfd.conf file must be made available.

If you choose not to use the AOPCPETC job, then you can copy configuration files as shown in this chapter in the examples that follow.
Note: There are three files that can be copied from /usr/lpp/Printsrv/samples to /etc/Printsrv as shown in Figure 2-2. If you plan to use Infoprint Transform Manager for Linux, then you should also copy aoprf.conf.
Sample aopxfd.conf file

Figure 2-3 displays the sample transform entries in the /usr/lpp/Printsrv/samples/aopxfd.conf file that can be used to create an installation /etc/Printsrv/aopxfd.conf file.

```
transform afp2pc1
transform afp2pc1_eu
transform afp2pc1_us
transform afp2pdf
transform afp2pdf_eu
transform afp2pdf_us
transform afp2ps
transform afp2ps_eu
transform afp2ps_us
transform pcl2afp
transform pcl2afp_a4_240
transform pcl2afp_a4_300
transform pcl2afp_a4_600
transform pcl2afp_legal_240
transform pcl2afp_legal_300
transform pcl2afp_legal_600
transform pcl2afp_letter_240
transform pcl2afp_letter_300
transform pcl2afp_letter_600
transform ps2afp
```

Figure 2-3  Sample transform entries in /usr/lpp/Printsrv/samples/aopxfd.conf

2.4 Infoprint Server configuration file (aopd.conf)

To start the Infoprint Server transforms, you must add or edit the Infoprint Server aopd.conf configuration file, shown in Figure 2-4 on page 32, by adding the following specification xfd to the existing values in the start-daemons attribute to start the Infoprint Server transform daemon, aopxfd.

```
start-daemons = { xfd }
```

The default Infoprint Server configuration file is located in /etc/Printsrv/aopd.conf. This file is optional. The file named in the AOPCONF environment variable takes precedence over this file.
After you edit the aopd.conf file, you then need to customize the transform configuration file, aopxfd.conf, and restart the transform daemon. You do not need to stop Infoprint Server before restarting. Also, if you run Infoprint Central, restart the z/OS HTTP Server.

**Products using the transform daemon**

The following products require services provided by the transform daemon:

- Infoprint Transforms to AFP for z/OS
- Infoprint Transform for AFP to HP PCL for z/OS
- Infoprint Transform for AFP to Adobe PDF for z/OS
- Infoprint Transform for AFP to Adobe PostScript for z/OS
- Infoprint XML Extender for z/OS
- Infoprint XT Extender for z/OS

**Note:** You do not need to start the Infoprint Server transforms to use the SAP to AFP transform that Infoprint Transforms to AFP for z/OS provides.

### 2.5 Transform configuration file (aopxfd.conf)

The transform configuration file, aopxfd.conf, contains information that the Infoprint Server transforms use for transforms that run on the z/OS systems.

**Note:** To use Infoprint Transform Manager for Linux or Infoprint Manager for AIX/Windows transforms, you do not need to create the aopxfd.conf file.

In a transform entry in the aopxfd.conf file you can specify the following:

- Attributes that the Infoprint Server Transform Interface uses to manage the transform.
- Environment variables for the transform. Each transform uses different environment variables.
2.5.1 Create an aopxfd.conf file

This file is used by the Infoprint Server Transform Manager. The Transform Manager manages transforms, which convert data streams from one format to another. Create an entry for each transform. If you define transform classes, create a separate entry for each transform class. A sample aopxfd.conf is provided in /usr/lpp/Printsrv/samples as shown in Figure 2-2 on page 30.

Copy the sample aopxfd.conf file

The transform configuration file, aopxfd.conf, contains information that the Infoprint Server Transform Manager uses to manage transforms that run on the z/OS system. You must create the transform configuration file before you start the Infoprint Server Transform Manager. A sample aopxfd.conf file is available in /usr/lpp/Printsrv/samples/aopxfd.conf.

This file must be copied as a sample to /etc/Printsrv.

Create an aopxfd.conf file by using the supplied sample.
1. From ISPF Option 6 command line, type the ISHELL command; the panel shown in Figure 2-5 is displayed.
2. Type /usr/lpp/Printsrv/samples/aopxfd.conf and press Enter.

3. Place the cursor under File and press Enter; the panel in Figure 2-6 on page 34 is displayed.
4. Select Option 8 to copy the file; the panel in Figure 2-7 is displayed.

5. Select Option 1 that a file is being created. Press Enter. The panel shown in Figure 2-8 on page 35 is displayed.

6. The panel shown in Figure 2-8 on page 35 is displayed. Change the pathname from /usr/lpp/Printsrv/samples/aopxfd/conf to /etc/Printsrv/aopxfd.conf. Press Enter; the panel in Figure 2-9 on page 35 is displayed.
2.6 Creating a transform configuration file using ISPF shell

This section shows the TSO/E UNIX System Services ISPF shell dialog steps that can be used to create the aopxfd.conf configuration file for transforms. The transform configuration file, aopxfd.conf, contains information that the Infoprint Server Transform Manager uses to manage transforms that run on the z/OS system. You must create the transform configuration file before you start the Infoprint Server Transform Manager.
2.6.1 Create an aopxfd.conf file

Use the following steps to create the aopxfd.conf configuration file for the management of the Infoprint Server transforms:

1. On the ISPF command line (option 6 on the ISPF Primary Option Menu) type the ISHELL command. You then enter the UNIX System Services ISPF Shell displayed in Figure 2-10. Type the pathname /etc/Printsrv/aopxfd.conf and press Enter.

   ![Figure 2-10 UNIX System Services ISPF Shell panel](image)

2. If the effective UID (EUID=) is 0, as shown in Figure 2-10, proceed to step 4 on page 37.

3. Place the cursor under Setup and press Enter. On the Setup pull-down window select option 7 Enable superuser mode (SU), as shown in Figure 2-11 on page 37. When you press Enter, ISHELL issues the SU command to switch your UID to superuser.

   ![Figure 2-11 UNIX System Services ISPF Shell panel](image)

   **Note:** You must be a z/OS UNIX superuser, (UID=0), to create this file. Therefore, if your effective UID is not 0, (field EUID= at the bottom of the display in Figure 2-10), switch to an effective UID of 0.

   To use the z/OS UNIX su command, you must be permitted to the RACF® profile BPX.SUPERUSER in the FACILITY class.
4. If the aopxfd.conf file does not exist when you press the Enter key, the Create a New File panel will be displayed (Figure 2-12). Select 2 for the “File type,” 2 for “File source for regular file,” and Y for “View and set attributes.”

5. Press Enter. The Display File Attributes panel will be displayed (Figure 2-13 on page 38).

Note: Panel ISREDDE2 /etc/Printsrv/aopxfd.conf is displayed if the aopxfd.conf file already exists.
6. Select **Edit** from the action bar; the Edit File Attributes pull-down window will be displayed (Figure 2-14).

Select 2. **Owning user**.

7. The Change the File Owner pull-down window will be displayed (Figure 2-15 on page 39). Set the “UID number” to 0 or the “User ID” to a superuser’s user ID.
8. When you press Enter, the Display File Attributes panel is displayed again.

   Select **Edit** again and on the next Edit File Attributes pull-down window select option 3 **Owning group**. The Change the Owning Group pull-down will be displayed (Figure 2-16).

   On the pull-down window enter AOPADMIN in the “Group name” field. The group owner AOPADMIN is the suggested RACF group name for Infoprint Server administrators. However, you might have used a different name for this group.

9. When you exit (F3) from the Change the Owning Group pull-down window, the Replace a File panel will be displayed (Figure 2-17 on page 40).

   Select 1. **File** as the “Source for copy” and press Enter.
10. The Enter the Pathname window will be displayed (Figure 2-18).

Enter the sample aopxfd.conf configuration file name
/usr/lpp/Printsrv/samples/aopxfd.conf in the “Change this to the pathname of the
source file” field.

11. Press Enter; the Enter File Permissions pull-down window will be displayed.

Change the “Permissions” of the file to 644 so that it is readable by everyone and writable
only by UID 0 and members of the AOPADMIN group (Figure 2-19 on page 41).
12. Press Enter to return to the UNIX System Services ISPF Shell panel.

2.7 Edit an aopxfd.conf file

The aopxfd.conf file can now be edited. In the transform configuration file, create at least one entry for each transform. In the transform entry you can specify:

- Attributes that the Infoprint Server Transform Manager uses to manage the transform.
- Environment variables for the transform. Each transform uses different environment variables. For information about the environment variables you can specify for each transform, see “Infoprint Server environment variables” on page 28.

Using the ISPF shell

The following steps show the UNIX System Services ISPF shell dialog used to edit the aopxfd.conf configuration file for transforms.

1. On the ISPF Command Shell panel (option 6 on the ISPF Primary Option Menu) enter ISHELL. The UNIX System Services ISPF Shell is displayed as shown in Figure 2-20 on page 42.
2. Enter /etc/Printsrv as shown in Figure 2-20.
   Make sure you have an effective UID (EUID=0). If EUID= is not zero, perform the procedure described in step 3 on page 33.
3. Press Enter; the /etc/Printsrv/ Directory List panel is displayed (Figure 2-21).

Note: You can skip this step by selecting /etc/Printsrv/aopxfd.conf in Step 2.

4. The following action codes are provided for working with files on the Directory list panel:

- **A** - Attributes
- **D** - Delete
- **R** - Rename
- **E** - Edit
- **B** - Browse text
- **V** - View records
- **C** - Copy to
- **I** - Replace From
- **P** - Print
- **F** - Find
- **X** - Run
- **U** - File system
- **G** - Edit records
- **M** - Compare

To edit the aopxfd.conf file, enter action code **E** for the file and press Enter. The Select Edit or Browse Options panel shown in Figure 2-22 on page 43 will be displayed.
Figure 2-22 Select Edit or Browse Options pull-down

5. To avoid seeing this panel in the future, select “Bypass this panel on edit” by placing a slash (/) beside it in the Select additional options field. Press Enter; you can now start editing the aopxfd.conf file, as shown in Figure 2-23.

Figure 2-23 Edit the aopxfd.conf file

2.7.1 Activate the aopxfd.conf file

Once the file has been edited to meet your installation’s standards or requirements, it has to be activated before it can be used. To activate the aopxfd.conf file changes, you have to stop and restart the Infoprint Server Transform Interface. The aopstart command starts Infoprint Server daemons. If a daemon is already started, aopstart does not start that daemon again.

For example, enter the MVS console commands to run the AOPSTOP JCL procedure to stop any active xfd daemon and then start the xfd daemon with your first configuration file or an updated configuration file which uses the AOPSTART JCL procedures, as follows:

```
S AOPSTOP,OPTIONS=' -d xfd'
S AOPSTART
```
2.7.2 Infoprint Server transform error messages

After restarting the xfd daemon, check for error messages in the stderr file of each transform daemon. Each instance of a transform daemon writes error messages to its own stderr file, which is cleared when the Infoprint Server Transform Interface restarts. If you find any error messages, fix the errors and restart the Infoprint Server transforms.

To be able to read the stderr file, you must be a member of the AOPADMIN RACF group.

Use either of the following methods to view an stderr file:

► Use the XFDERRE REXX™ program to browse transform stderr error files.
► Use the ISHELL to view the stderr error files.

Using XFDERRE REXX program for stderr files

The XFDERRE REXX TSO/E command invokes the UNIX System Services ISPF shell (ISHELL) for the transform stderr files that are not empty.

1. Go to the ISPF Option 6 command line and type XFDERRE, as shown in Figure 2-24.

Figure 2-24   Check for error messages in the stderr file using the XFDERRE REXX exec

2. The panel shown in Figure 2-25 on page 45 is displayed showing several stderr files.

To browse one of the files, place a b next to the file pathname.
Chaprer 2. Infoprint Server transforms

1. The panel in Figure 2-27 is displayed, showing the stderr file for the ps2afp transform. Place a b next to the pathname to display the error messages.

2. Figure 2-28 shows the stderr file containing the error messages.

Using the ISHELL for the stderr files

Another option to view the stderr file is to find the transform's stderr file by looking in directory base-directory/xfd, where base-directory is the directory specified in the base-directory attribute of the Infoprint Server aopd.conf configuration file. The default base directory is /var/printsrv.

The stderr file naming convention is: transform[_class].n.stderr
The first number n in the stderr file name is 0; n is increased by one for each new transform instance.
Use the UNIX System Services ISPF shell dialog steps for viewing transform error messages, as shown in Figure 2-29 through Figure 2-31.

1. Enter the ISHELL from ISPF Option 6, using the ISHELL command.
   Enter /var/printsrv/xfd as the pathname, as shown in Figure 2-29.

   ![Figure 2-29 Result of Infoprint Server transform error messages - Step 1](image)

   Enter the ISHELL from ISPF Option 6, using the ISHELL command.
   Enter /var/printsrv/xfd as the pathname, as shown in Figure 2-29.

   ![Figure 2-29 Result of Infoprint Server transform error messages - Step 1](image)

2. Press Enter. The panel in Figure 2-30 is displayed showing the current files and directories.
   To browse the stderr file, place a b next to the file.

   ![Figure 2-30 Result of Infoprint Server transform error messages - Step 2](image)

3. Figure 2-31 shows a display of the stderr file that is being browsed.

   ![Figure 2-31 Result of Infoprint Server transform error messages - Step 3](image)
2.8 Create a remote transform configuration file (aoprxf.conf)

The remote transform configuration file contains information that Transform Interface uses to access Infoprint Transform Manager for Linux. If you use Infoprint Transform Manager for Linux, the aoprxf.conf configuration file is required. A remotexf command transforms data from one format to another remotely using Infoprint Transform Manager for Linux transforms. The command lets you convert files to another format without printing the files. The command automatically detects the format of the input file and uses the appropriate transform, such as PDF to AFP, for that data format. However, you can specify the document-format job attribute to override the input format.

**Note:** Use the z/OS UNIX remotexf command to transform a document. You can use the AOPBATCH JCL procedure to run the remotexf command.

2.8.1 File conversion to AFP format

Infoprint Transform Manager for Linux transforms let you convert files to Advanced Function Presentation (AFP) format from:

- Adobe Portable Document Format (PDF)
- Adobe PostScript
- Graphics interchange format (GIF)
- HP Printer Control Language (PCL)
- Joint Photographic Experts Group (JPEG) file information format (JFIF)
- Tagged image file format (TIFF)

The remotexf command and the aoprxf.so filter in the z/OS printer definitions send files to Infoprint Transform Manager for Linux.

Figure 2-32 on page 48 shows the components of the Infoprint Transform Manager for Linux.
Figure 2-32 illustrates how the components of Infoprint Transform Manager for Linux interact with one another and with the print servers and the following discussion provides details:

1. Print jobs are submitted to one or more print servers that have been configured to send transform requests to the Infoprint Transform Manager for Linux. A print server interface transmits the request to the coordinator. The coordinator component receives the request and information regarding the type of transformation to be performed.

2. The coordinator sends the transform request to a transform server component, which transforms the data with the appropriate transform, such as PostScript to AFP. If you are using more than one transform server, the coordinator sends the request to the transform server with the lowest processing load.

3. To maximize print server performance in sending data to the printers, the transform servers sequence the transformed data in the correct order and return it to the print servers as the data completes processing, instead of waiting for the entire job to finish.

4. The user interface component, which you access from a Web browser, lets you define which print servers and transform servers the Transform Manager communicates with. With the user interface, you can start and stop transform servers, check the operating status of the transform servers, determine the percentage of work pending for a transform server, view the message log, configure a trace to help diagnose a problem (only when requested by IBM Software Support), and see what transforms are installed on a transform server.

**Using Infoprint Transform Manager for Linux**

If you submit numerous transform requests from a print server through the Infoprint Transform Manager for Linux, those transform requests might not return to the print server in the same order. The order in which the transformed data returns to the print server depends on the kind of transform (PDF, PS, GIF, JPEG, or TIFF) and the size of the request.
For example, imagine that you send a PS file and then you send a PDF file that is the same size. The PS file can only be transformed on a job basis. The PDF file, on the other hand, can be transformed in parallel on a page-by-page basis over multiple processors. Therefore, the PDF file might be transformed first and the Transform Manager might return it to the print server before the PS file.

The Infoprint Server remote transform configuration file contains information that Transform Interface uses to access Infoprint Transform Manager for Linux. If you use Infoprint Transform Manager for Linux, this configuration file is required.

The default directory for the remote transform configuration file is /etc/Printsrv/ and the file name is aoprxf.conf.

The format of the remote transform configuration file is:

```
[ #comment ]
remote-transform-manager-ip-address = hostname
remote-transform-manager-port-number = portnumber
```

Where:

- `remote-transform-manager-ip-address = hostname`
  - The host name or dotted-decimal IP address of the Infoprint Transform Manager for Linux transform server. This attribute is required.
  - Default: None

- `remote-transform-manager-port-number = portnumber`
  - The port number of the Infoprint Transform Manager for Linux transform server. This attribute is optional.
  - Default: remote-transform-manager-port-number = 6986

The sample remote configuration file is /usr/lpp/Printsrv/samples/aoprxf.conf.

The environment variable for the remote configuration file is AOPRXF_CONF. It defines the full path name of the remote transform configuration file aoprxf.conf. This environment variable is optional. If you did not create this configuration file, or if you created it in the default location, you do not need to set the AOPRXF_CONF environment variable. The default is /etc/Printsrv/aoprxf.conf. See “Infoprint Server environment variables” on page 28.

Transforms that run on non-z/OS systems might use transform resources that reside on the z/OS system. For example, Infoprint Transform Manager for Linux transforms use a PostScript halftone dictionary, which can reside on the z/OS system. If a resource resides on the z/OS system, set the z/OS UNIX permissions for the file so that the user ID that starts Infoprint Server can read it.

**Customizing the TCP/IP network firewall**

If you have configured a TCP/IP firewall, make sure that the z/OS system is allowed to communicate using the TCP/IP protocol with Infoprint transforms that run on remote systems.

If you use Infoprint Transform Manager for Linux to transform data, the z/OS system must be able to connect to each Infoprint Transform Manager transform server that the administrator has defined.
The z/OS system must be able to connect to each transform server at both these ports:

- The transform server's defined port. The default port is 6986.
  
  You specify the defined port in the remote-transform-manager-port-number attribute in the remote transform configuration file, aoprxf.conf.
- Port 6984, which is the port to which Infoprint Server sends data.

**Creating a code page conversion table**

Transform Interface converts messages from Infoprint Transform Manager for Linux from the ISO8859-1 code page to the code page for the z/OS locale. Therefore, if you use a custom code page for the z/OS locale, make sure that a conversion table exists.

*z/OS XL C/C++ Programming Guide*, SC09-4765 describes the code set conversion utilities supported by the z/OS XL C/C++ compiler. These utilities are as follows:

<table>
<thead>
<tr>
<th>Utility</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>genxlt utility</td>
<td>Generates a translation table for use by the iconv utility and iconv() functions.</td>
</tr>
<tr>
<td>iconv utility</td>
<td>Converts a file from one code set encoding to another.</td>
</tr>
<tr>
<td>iconv() functions</td>
<td>Perform code set translation. These functions are iconv_open(), iconv(), and iconv_close(). They are used by the iconv utility and may be called from any z/OS XL C/C++ program requiring code set translation.</td>
</tr>
<tr>
<td>uconvdef utility</td>
<td>Handles Universal-coded character sets. Creates binary conversion tables that define mapping between UCS-2 and multibyte code sets.</td>
</tr>
</tbody>
</table>

**2.9 Setting up security for transform resources**

The user ID that starts Infoprint Server must have permission to read any transform resource files that reside on the z/OS system. For example, the AFP to PCL, AFP to PDF, and AFP to PostScript transforms use AFP resources, such as form definitions and fonts, that reside on the z/OS system. The SAP to AFP transform also uses configuration files that reside on the z/OS system.

Security access must be provided as follows:

- The user ID that starts Infoprint Server must have RACF READ access to all system resource libraries and user resource libraries that the transform uses.
- Users who belong to the AOPOPER group should be given READ access to the resource libraries. If someone with a user ID of 0 who is not a member of the AOPOPER group can start Infoprint Server (for example, using the aopstart command), you must also give this user READ access to the resource libraries.
- Job submitters must have RACF READ access to all user AFP resource libraries that are specified in the printer definition, JCL statements, or job attributes.
2.9.1 Security for AFP resource libraries

The IBM Infoprint Transforms from AFP for z/OS require the installation’s security product, (for example RACF), to allow access to AFP system resource libraries and user resource libraries.

- System resource libraries are libraries specified in the aopxfd.conf transform configuration file for the AFP to PCL, AFP to PDF, and AFP to PostScript transforms.
- User resource libraries are libraries specified in the:
  - Resource library field in the printer definition
  - USERLIB parameter of the OUTPUT JCL statement
  - Resource library job attribute

**Note:** AOPOPER is the recommended group name for Infoprint Server operators. However, your installation can assign a different name to this group. Therefore, you should give the AOPOPER group READ access to the resource libraries.

- Users who submit print jobs that require from AFP transforms must have RACF READ access to all user AFP resource libraries that are specified in the printer definition, JCL statements, or job attributes.
  
In addition, print job submitters must use submission methods that authenticate their z/OS user IDs so that Infoprint Server can use the user ID to check RACF access to the resource libraries. Job submission methods that can authenticate z/OS user IDs include:

- lp, afp2pcl, afp2pdf, and afp2ps commands
- SAP R/3, using the Infoprint Server SAP Output Management System (OMS)
- Windows SMB protocol
- AOPPRINT and AOPBATCH JCL procedures
- z/OS JCL that submits a print job to the Infoprint Server subsystem
- z/OS JCL that submits a print job to IP PrintWay extended mode

**Note:** Print job submitters who use other job submission methods, such as the Infoprint Port Monitor for Windows, can only use user AFP resource libraries that have universal READ access.
2.9.2 Security for the file ps2afpd

Security checking in the ps2afp transform requires that the user identifier (UID) of the executable file for the ps2afpd transform is not 0 (zero). File ps2afpd is assigned a UID of 0 when it is installed. Therefore, you must change the owner of the file. The new owner must have other than 0 UID or the default UID. Also, the set-user-ID flag for the file must be turned on.

**Note:** This is the only transform file that requires this security change.

**Define an owner for the ps2afpd file using RACF**

Use RACF or another security product that follows system authorization facility (SAF) protocol to create a user and group profile for the owner of ps2afpd file. Then, change the owner of ps2afpd and turn on the set-user-ID flag, as follows:

1. Define a group to RACF. The group profile must have an OMVS segment and a group identifier (GID). You can use any group name. IBM recommends that you do not give this group any authority to the z/OS file system.
2. Define a user profile with an OMVS segment to RACF. It must have the following:
   - Not have UID=0
   - Not have a UID which is defined in the BPX.DEFAULT.USER profile in the RACF FACILITY class
3. Change this user to the owner of the ps2afpd file.
   IBM recommends that you do not give this user authority to the z/OS file system other than what is required by PostScript jobs to be transformed.

**Note:** Specify the name of a RACF group, defined in step 1, to be the default group for the user. You do not have to issue the CONNECT command to connect new users to their default groups.

The ps2afp transform, as well as PostScript jobs being transformed, will run with the file owner's UID.

Following are two different ways to create the file owner and set the set-user-ID flag. You can choose your preferred method when using z/OS UNIX.

**Using the OMVS shell to create the file owner**

Do the following to change the ps2afpd file owner and set-user-ID flag:

1. Switch to an effective UID of 0 and assign the previously defined user ID as the owner of the /usr/lpp/Printsrv/bin/ps2afpd file.
   ```bash
   chown new_owner /usr/lpp/Printsrv/bin/ps2afpd
   ```
2. Turn the set-user-ID flag on for file ps2afpd.
   ```bash
   chmod u+s /usr/lpp/Printsrv/bin/ps2afpd
   ```

**Using the ISHELL to create the file owner**

From the ISPF Option 6 command line, enter the ISHELL command.

1. On the UNIX System Services ISPF Shell panel (BPXWP99) enter pathname `/usr/lpp/Printsrv/bin` as shown in Figure 2-33. Press Enter.
2. The Directory List panel (BPXWP06A) is displayed, as shown in Figure 2-34.

Scroll down the list of commands until you see ps2afpd. Set action code (a) for the file ps2afpd and press Enter.

3. The panel shown in Figure 2-35 on page 54 is displayed. Place the cursor under Edit and press Enter.
4. The panel shown in Figure 2-36 is displayed where you then Select Option 2 to change the owning user.

5. The panel shown in Figure 2-37 on page 55 is displayed. Here you can specify an owning user and set that user’s UID.
6. Now, to set the set-UID bit, go back to access the panel shown in Figure 2-36 on page 54. Select Option 1 to set the set-UID bit.

7. The panel shown in Figure 2-39 on page 56 is displayed. There you then make sure the permissions are okay and place a 1 in the Set UID bit field as shown.
2.10 Customizing transforms

IBM provides several separately priced Infoprint transform products that convert data from one format to another. Some of these products run on the z/OS system, while others run on other systems. The documentation for each transform product describes the supported functions and limitations of the transforms. This section describes how to customize the various z/OS Infoprint Server transforms.

Transform daemons

Transform daemons are programs that run unattended to perform continuous or periodic functions. The Infoprint Server Transform Manager (part of the Transform Interface) starts and stops transform daemons, as shown in Figure 2-42 on page 61. These transform daemons require configuration customization by an administrator.

The transform daemons can be found in the file system at /usr/lpp/Printsrv/bin. These transform daemons are the executable programs that do the data stream transformation. The transform daemons are:

- afp2pcld
- afp2pdfd
- pcl2afpd
- ps2afpd
- sap2afpd
- xml2pdfd
2.10.1 Using transform classes

You must create at least one transform entry in the Infoprint Server aopxfd.conf transform configuration file for each transform you plan to use. Each transform entry consists of a set of attributes, starting with the transform attribute and ending with a semicolon. The general format of all transform class entries is shown in Figure 2-40. In the transform entry, you can specify:

- Environment variables that control the transform
- Attributes that control how the Infoprint Server Transform Interface manages the transform

If you want to specify different environment variables for the same transform for different printing situations, you can create different classes for the transform. For example, if you want to transform PCL data to AFP format for printers that have different resolutions or paper sizes, you can define a transform class for each combination of printer resolution and paper size.

The aopxfd.conf file contains entries with the format shown in Figure 2-40. The default Transform Manager configuration file can be accessed from /etc/Printsrv/aopxfd.conf. This file is required if you start the Transform Manager daemon. Any aopxfd.conf file named in the AOPXFD_CONF environment variable takes precedence over this file.

```plaintext
# comment
transform transformname[_transformclass]
  start-command = "daemon [ option ]..."
  [ environment = {name -> value [ name -> value ]...} ]
  [ maximum-active = number ]
  [ maximum-idle-time = seconds ]
  [ minimum-active = number ]
;
```

*Figure 2-40  Parameter attributes to define a transform class*

**Transform class attributes**

The transform entry attributes and environment variables apply whether Infoprint Server invokes a transform automatically or the user invokes a transform with a z/OS UNIX transform command, as shown in Figure 1-16 and Figure 1-17 on page 24.

In the aopxfd.conf file each transform class has its own separate entry. The sample transform configuration file (/usr/lpp/Printsrv/samples/aopxfd.conf) defines several transform classes.
In the transform configuration file, create at least one entry for each transform. In the transform entry you can specify attributes that the Infoprint Server Transform Manager uses to manage the transform. In addition, you also specify environment variables for the transform. Each transform uses different environment variables.

The attributes in the aopxfd.conf file are defined in the following sections.

**transform transformname[ _transformclass]**
This attribute begins a transform entry. The attribute is required and must be the first attribute in the entry, as follows:

- **transformname** The name of the transform, for example, pcl2afp or ps2afp.
- **transformclass** The name of an optional transform class. Specify from 1 to 63 characters, including letters, numbers, or special characters. The transform name is case-sensitive. To use the transform class, you must specify the transform class when you invoke the transform.

  Default: No transform class. This transform entry is used when no transform class option (-c) is specified either in the printer definition or by the job submitter.

  Example: transform pcl2afp_letter_300

**start-command = "daemon [option]... "**
The name of the transform daemon and options. Enclose the value in single or double quotation marks if you specify an option. This attribute is required.

- **daemon** The name of the transform daemon, for example, pcl2afpd or ps2afpd. If the transform daemon is not in a directory identified in the PATH environment variable in the aopstart EXEC, specify the full directory path name of the daemon.

  **Note:** The aopstart command is a REXX program that sets environment variables that Infoprint Server daemons use. It also starts the Printer Inventory daemon and any other daemons specified in the start-daemons attribute in the Infoprint Server configuration file (aopd.conf).

  The default aopstart EXEC is in the /usr/lpp/Printsrv/bin/ directory.

- **option** One or more options supported by the transform daemon.

  **Note:** -m nnn[K|M]

  The maximum number of bytes of memory the transform can use. Specify the number of bytes in either kilobytes (K) or megabytes (M). The amount of memory required depends on the compression type, the image being compressed, fonts, and so on.

  **Example:** start-command = "pcl2afpd -m 4M"

  **Example:** start-command = "ps2afpd -m 246M"
**environment = {name -> value [ name -> value ]... }**

Environment variables define the transform environment. Enclose the entire set of environment variables in braces. See the documentation for each transform product for information about the environment variables you can specify for the transform.

These environment variables override environment variables with the same names that you set in the aopstart EXEC. The aopstart EXEC also picks up selected environment variables from the run-time environment.

You can specify the _BPX_JOBNAME environment variable in each transform class entry to assign a different job name to each class of transform daemons. By default, the job name of each transform daemon is AOPXFD. When you assign a different job name to each class of transform daemon, the operator can manage the transform daemons more effectively.

**maximum-active = number**

This is the maximum number of transform daemons that the Infoprint Server Transform Interface activates concurrently for a transform class. Specify a number greater than 0 and greater than or equal to the number specified in minimum-active. When this number is reached, the Infoprint Server Transform Interface does not start a new transform daemon to do transforms in the class, and print jobs wait until a transform daemon becomes available. This attribute is optional.

**Recommendations:** A value of 2 or 3 is suitable if you transform small print jobs and do not transform many print jobs at the same time. However, if you might need to transform large print jobs at the same time, consider setting a higher number. Very large print jobs might take several minutes or even an hour to transform. As a result, a transform daemon might not be available to service other transform requests. If the other transform requests are delayed long enough, the requesting programs might time-out and fail.

If you use the Print Interface subsystem in your installation, do not specify a low value because jobs that use the subsystem and write multiple data sets might not be able to complete.

To avoid having many transform daemons active, consider specifying either maximum-active or maximum-idle-time, or both.

Default: No maximum number. Transform daemons are started when needed.

**maximum-idle-time = seconds**

This is the number of seconds before an idle transform daemon is shut down and system resources are freed. Specify a number greater than 0.

This attribute is optional. However, to avoid having many transform daemons active, consider specifying either this attribute or the maximum-active attribute, or both.

Default: An idle transform daemon is not shut down.

**minimum-active = number**

This is the minimum number of transform daemons that are concurrently active for a transform class. It is also the minimum number of transform daemons that are active (that is, not shut down) even when the maximum-idle-time expires for an idle transform daemon. Specify a number less than or equal to the number specified in the maximum-active attribute. This attribute is optional.
Do not specify a very high number because active transform daemons use some system resources. However, starting transform daemons uses additional CPU resources, so you should keep some transform daemons active even when they are idle.

Default: minimum-active = 0

2.10.2 Example of a transform entry

Figure 2-41 shows one sample entry in the aopxfd.conf configuration file for the PCL to AFP transform.

When the Infoprint Server transform starts, it starts one transform daemon for the PCL to AFP transform class _letter_300: minimum-active = 1. No more than two transform daemons are active at any time for class _letter_300: maximum-active = 2. Transform daemons that are idle 5 minutes are shut down: maximum-idle-time = 300.

Because the transform transformname_transformclass attribute specifies a transform class, this entry is used only when the transform class is specified as a transform command option or a filter option.

```plaintext
transform pcl2afp_letter_300
start-command = "pcl2afpd"
environment = { _BPX_JOBNAME -> PCL2AFPD }
minimum-active = 1
maximum-active = 2
maximum-idle-time = 300  # 5 minutes
environment = {
    AOP_RESOLUTION -> 300
    _BPX_JOBNAME -> PCL2AFPD
}
;
```

Figure 2-41  Defining a transform entry

Using the transform class

To use the transform class shown in Figure 2-41, the administrator or the job submitter must specify the transform class in one of the following ways:

- In the printer definition using the -c filter option. For example, in the filter field for the PCL data format, specify:
  
  `pcl2afp.dll -c letter_300`

- On the z/OS UNIX transform command using the -c option. For example:
  
  `pcl2afp -c letter_300 -o myfile.afp myfile.pcl`

- In the filter-options job attribute on the lp command. For example:
  
  `lp -d IAZFSS -o "filter-options=-'c letter_300'" myfile.pcl`

2.10.3 Infoprint Server transforms to be customized

IBM provides several separately priced Infoprint transform products that convert data from one format to another. The documentation for each transform product describes the supported functions and limitations of the transforms.
Details about customizing the transforms are presented in the following sections:

- 2.11, “PCL to AFP transform customization” on page 62 as well as in transform daemon pcl2afpd discussion
- 2.12, “Customizing the PDF to AFP and PS to AFP transforms” on page 66 and in transform daemon ps2afpd discussion
- 2.13, “Customizing the SAP to AFP transform” on page 74
- 2.14, “Customizing AFP to PCL transform” on page 79.
- 2.15, “Customizing the AFP to PDF transform” on page 85
- 2.16, “Customizing the AFP to PostScript transform” on page 103

**Transform Manager**

The Transform Interface component of Infoprint Server communicates with transforms that Infoprint transform products provide. Transforms convert data from one format to another, for example, from PCL to AFP format and from AFP to PDF format.

When processing a print request, Print Interface (shown in Figure 2-42 on page 61), NetSpool, and IP PrintWay extended mode call Transform Interface to transform data from one format to another if the administrator specifies a transform filter in the printer definition. The administrator can specify a different transform filter for each data format.

When the transform runs on z/OS, the Infoprint Server Transform Manager (part of Transform Interface) starts and stops the transform daemons, using configuration information specified by the administrator.

The Infoprint Server Transform Manager manages the transforms. When you update the transform configuration file, you must restart the Infoprint Server Transform Manager.

**Note:** The Infoprint Server Transform Manager does not start the SAP to AFP transform because this transform is not implemented as a daemon.
2.11 PCL to AFP transform customization

To customize the PCL to AFP transform all you need to do is specify transform options, a required task. The PCL to AFP transform converts Printer Control Language (PCL) 6 (XL, 5, 5e, 5c) format documents to AFP format. The transform can accept PCL documents in color. However, it always creates a monochrome AFP image.

The PCL to AFP transform creates an AFP Image Object Content Architecture (IOCA) image for each page in the PCL document. IOCA provides a way to represent documents in a device-independent format, which allows them to be interchangeable across environments. IOCA uses a consistent set of constructs, called self-defining fields, to describe the characteristics of the image data.

The PCL to AFP transform can produce a printable AFP document, or an AFP overlay or page segment that you can print as part of other documents. The AFP document can be a compressed IOCA image (recommended for faster printing) or an uncompressed image.

You can specify the type of IOCA image and type of document the transform creates in options on the transform command and in options in the printer definition. The Infoprint Server administrator can specify the height and width of the output page and the resolution of the IOCA image in environment variables in the aopxfd.conf transform configuration file.

2.11.1 Customization tasks for transforms

Customizing the PCL to AFP transform involves the following tasks:

- Specify transform options - Required
  See “Transform classes for PCL to AFP” on page 62.
- Start Infoprint Server with sufficient memory - Required
  See “Starting Infoprint Server with sufficient region size” on page 71.
- Set up security - Required
  See “Setting up security for transform resources” on page 50.

These tasks are required by all installations.

2.11.2 Transform configuration file aopxfd.conf

For information about how to create and edit the Infoprint Server transform configuration file, see “Create an aopxfd.conf file” on page 36. After you update the transform configuration file, you must restart the Infoprint Server Transform Interface.

Once this file is created, you then add transform entries for this transform, as shown in “Edit an aopxfd.conf file” on page 41 and then “Activate the aopxfd.conf file” on page 43.

Once the transforms have restarted, you should check for error messages in the transform stderr files. If you find any error messages, fix the errors and restart the Transform Interface. For more information about how to find the transform message logs, see “Infoprint Server transform error messages” on page 44.

2.11.3 Transform classes for PCL to AFP

When editing the aopxfd.conf configuration file, you can create different transform classes for different transform options and for different printers. For example, you could create classes
for printers that print on different paper sizes. To define transform classes in the aopxfd.conf transform configuration file, see “Using transform classes” on page 57.

To use a given transform class, job submitters specify the class name in the -c option on the pcl2afp transform command or in the filter-options job attribute. Infoprint Server administrators specify the class name in the -c filter option in the printer definition. For example, these z/OS UNIX commands use the letter_300 transform class:

```
pcl2afp -c letter_300 -o myfile.afp myfile.pcl
lp -d myprinter -o "filter-options=f-c letter_300f" myfile.pcl
```

The sample transform configuration file, /usr/lpp/Printsrv/samples/aopxfd.conf, has examples of transform classes.

**PCL to AFP transform entry pcl2afp**

See “Using transform classes” on page 57 for the general format of transform entries in the transform configuration file (aopxfd.conf).

Figure 2-43 shows the pcl2afp_letter_300 transform class entry in the /usr/lpp/Printsrv/samples/aopxfd.conf file.

```
transform pcl2afp_letter_300
   start-command = pcl2afpd
   min-active = 0
   max-active = 2
   maximum-idle-time = 300  # 5 minutes
   environment = {
      AOP_RESOLUTION -> 300
      _BPX_JOBNAME -> PCL2AFPD
   }
;
```

*Figure 2-43  PCL to AFP transform pcl2afp_letter_300 transform class entry*

**Transform class parameters**

The syntax for the start-command attribute in the PCL to AFP transform entry in the transform configuration file (aopxfd.conf) is:

```
start-command = "pcl2afpd [-m nnn[K|M]]"
```

This attribute names the transform daemon and option. Enclose the value in single or double quotation marks if you specify the option.

**pcl2afpd**  The name of the transform daemon. If the transform daemon is not in a directory identified in the PATH environment variable in the aopstart EXEC, specify the full directory path name of the daemon. (The pcl2afpd daemon is installed in /usr/lpp/Printsrv/bin.) This attribute is required.

**-m nnn[K|M]**  The number of bytes of storage the transform can use. Specify the number of bytes in either kilobytes (K) or megabytes (M). The amount of storage required depends on the compression type, the image being compressed, fonts, and so on.

You can determine the number of megabytes with the following computation:

```
Megabytes = \((width \text{ in pels} / 8) \times height \text{ in pels}\) \times 3
width \text{ in pels} = (\text{width in inches} \times \text{resolution})
height \text{ in pels} = (\text{height in inches} \times \text{resolution})
```
Instead of performing the calculation, you can use Table 2-1 to obtain the storage amount. The table shows the amount of storage for different paper sizes and resolutions (240, 300, and 600 pels per inch), using the previous algorithm.

<table>
<thead>
<tr>
<th>Paper size</th>
<th>Resolution 240</th>
<th>Resolution 300</th>
<th>Resolution 600</th>
</tr>
</thead>
<tbody>
<tr>
<td>Letter</td>
<td>2.0M</td>
<td>3.2M</td>
<td>12.6M</td>
</tr>
<tr>
<td>Legal</td>
<td>2.6M</td>
<td>4.0M</td>
<td>16.1M</td>
</tr>
<tr>
<td>A4</td>
<td>2.1M</td>
<td>3.3M</td>
<td>13.1M</td>
</tr>
</tbody>
</table>

Default: 6M
If the transform fails due to lack of storage, increase the amount of storage.
Example: start-command = "pcl2afpd -m 4M"

### 2.11.4 Environment variables for the PCL to AFP

Infoprint Server transform environment variables let you specify the following options:

- Height and width of the page
- Margins of the page
- Resolution of the printer
- Record length of AFP images
- Tracing options

Attention: All environment variables are optional. The environment variables that you can specify are identified in Appendix C, “Transform environment variables” on page 203.

#### Example transform configuration file entries for PCL to AFP

Following are examples of aopxfd.conf configuration file entries:

- **Print on letter size paper on 300-pel printers**
  
  This transform class can be used for printers that print on letter size paper (8.5 inches wide, 11 inches high) and that have a resolution of 300 pels per inch. The transform class is called `letter_300`:
Chapter 2. Infoprint Server transforms

2.11.5 Supported fonts for PCL to AFP

The PCL to AFP transform contains a set of single-byte built-in fonts, which it uses to create a rasterized image of the data. The list of fonts that are built into the PCL to AFP transform is shown in Figure 2-46 on page 66.
DBCS fonts are not built into the transform, but the transform can process DBCS fonts that are embedded in the PCL document.

If a PCL document specifies a font that is not built in and is not embedded in the PCL document, the transform substitutes another font that is the best match for the specified font. You cannot customize the way the transform performs font substitution.

The administrator cannot add fonts to the transform. However, you can embed fonts in the PCL document.

<table>
<thead>
<tr>
<th>Font Name</th>
<th>Font Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Albertus Extra Bold</td>
<td>Albertus Medium</td>
</tr>
<tr>
<td>Antique Olive</td>
<td>Antique Olive Bold</td>
</tr>
<tr>
<td>Antique Olive Italic</td>
<td>Arial</td>
</tr>
<tr>
<td>Arial Bold</td>
<td>Arial BoldItalic</td>
</tr>
<tr>
<td>Arial Italic</td>
<td>CG Omega</td>
</tr>
<tr>
<td>CG Omega Bold</td>
<td>CG Omega BoldItalic</td>
</tr>
<tr>
<td>CG Omega Italic</td>
<td>CG Times</td>
</tr>
<tr>
<td>CG Times Bold</td>
<td>CG Times BoldItalic</td>
</tr>
<tr>
<td>CG Times Italic</td>
<td>Clarendon Condensed Bold</td>
</tr>
<tr>
<td>Coronet Courier</td>
<td>Courier Bold</td>
</tr>
<tr>
<td>Courier BoldItalic</td>
<td>CourierItalic</td>
</tr>
<tr>
<td>Garamond Antiqua</td>
<td>Garamond Halbfett</td>
</tr>
<tr>
<td>Garamond Kursiv</td>
<td>Garamond Kursiv Halbfett</td>
</tr>
<tr>
<td>Letter Gothic</td>
<td>Letter Gothic Bold</td>
</tr>
<tr>
<td>Letter Gothic Italic</td>
<td>Marigold</td>
</tr>
<tr>
<td>Symbol</td>
<td>Times New Roman</td>
</tr>
<tr>
<td>Times New Roman Bold</td>
<td>Times New Roman BoldItalic</td>
</tr>
<tr>
<td>Times New Roman Italic</td>
<td>Univers Bold</td>
</tr>
<tr>
<td>Univers BoldItalic</td>
<td>Univers Condensed Bold</td>
</tr>
<tr>
<td>Univers Condensed BoldItalic</td>
<td>Univers Condensed Medium</td>
</tr>
<tr>
<td>Univers Condensed MediumItalic</td>
<td>Univers Medium Univers MediumItalic</td>
</tr>
</tbody>
</table>

Figure 2-46  Fonts that are built into the PCL to AFP transform

2.11.6  PCL to AFP limitations

Each Infoprint transform has some limitations on what can be transformed. For PCL to AFP transform limitations, see C.7, “AFP architecture limitations for transforms” and C.7.1, “PCL to AFP limitations” on page 230.

2.12  Customizing the PDF to AFP and PS to AFP transforms

The customization tasks are the same for both PDF to AFP and PostScript to AFP transforms. Furthermore, the same transform daemon (ps2atpd) transforms both PDF and PostScript data streams.

To customize the PDF to AFP and PostScript to AFP transform involves the following steps:

- Specify transform options - Required
  See “Transform classes for PDF to AFP and PostScript to AFP” on page 67 and “Environment variables for the PDF to AFP and PostScript to AFP” on page 70.
- Start Infoprint Server with sufficient memory - Required
  See “Starting Infoprint Server with sufficient region size” on page 71.
Set up security - Required
See “Setting up security for transform resources” on page 50.

Add fonts - Optional
To add custom fonts, see “Adding fonts for PDF to AFP and PostScript to AFP” on page 72.

These tasks are required for all installations.

2.12.1 Transform configuration file aopxfd.conf

For information about how to create and edit the Infoprint Server transform configuration file, see “Create an aopxfd.conf file” on page 36. After you update the transform configuration file, you must restart the Infoprint Server Transform Interface.

Once this file is created, you then add transform entries for this transform, as shown in “Edit an aopxfd.conf file” on page 41 and then “Activate the aopxfd.conf file” on page 43.

Once the transforms have restarted, you should check for error messages in the transform stderr files. If you find any error messages, fix the errors and restart the Transform Interface. For more information about how to find the transform message logs, see “Infoprint Server transform error messages” on page 44.

2.12.2 Transform classes for PDF to AFP and PostScript to AFP

To define transform classes in the aopxfd.conf transform configuration file, see “Using transform classes” on page 57.

If you want to specify different transform options for different transform situations, you can create different transform classes. For example, you could create a separate class that turns tracing on. You could use this transform class if IBM requests that you trace the transform.

For each transform class, you must create a separate transform entry. You specify a name for the transform class in the transform entry. Figure 2-47 on page 67 shows the default ps2afp transform entry in the /usr/lpp/Printsrv/samples/aopxfd.conf file.

```bash
# PS or PDF -> AFP:
transform ps2afp_classn
    start-command = ps2afpd # ps2afp V2R1
    min-active = 1
    max-active = 2
    maximum-idle-time = 300 # 5 minutes
    environment = {
        _BPX_JOBNAME -> PS2AFPD
    }

Figure 2-47  PDF to AFP and PostScript to AFP transform entry
```

To use a transform class, job submitters must specify the class name, classn, in the -c option on the pdf2afp or ps2afp z/OS UNIX transform commands or in the filter-options job attribute on the z/OS UNIX lp command as shown in the following examples:

```
ps2afp -c classn -o myfile.afp myfile.ps
```
lp -d myprinter -o "filter-options='-c classn'" myfile.ps

The Infoprint Server administrator can specify a class name in the -c filter option in the printer definition, as shown in Figure 2-48.

```
Figure 2-48   Processing panel for a printer defined in the Printer Inventory
```

### Transform class parameters

In the PDF to AFP and PostScript to AFP transform class entry in the aopxfd.conf transform configuration file, the syntax for the `start-command` attribute is:

```
start-command = "ps2afpd [-m nnnnnn[K|M]]"
```

This attribute names the transform daemon and option. Enclose the value in single or double quotation marks if you specify the option.

- **ps2afpd**
  - The name of the transform daemon. If the transform daemon is not in a directory identified in the PATH environment variable in the aopstart EXEC, specify the full directory path name of the daemon. (The ps2afpd daemon is installed in /usr/lpp/Printsrv/bin.) This attribute is required.

- **-m nnnnnn[K|M]**
  - The number of bytes of memory the transform daemon uses to do transforms. Specify the number in either kilobytes (K) or megabytes (M). Valid values are:
    - 15M to 256M
    - 15360K to 262144K
  - The amount of memory required depends on the complexity of the documents to be transformed. If you specify too low a value, the transform of complex PDF or PostScript documents might fail with undefined PostScript errors. If this occurs, try increasing the value in this option.

  Also, specify a maximum region size that is at least 10M greater than the value you specify in this option. For example, if you specify -m 32M, set the region size to at least 42M (32M + 10M). IBM, however, recommends a region size of at least 256M. If the region size is too low, the transform will fail immediately due to lack of memory.

    - Default: 32M
2.12.3 PDF to AFP transform

The PDF to AFP transform converts documents in PDF 1.4 format to AFP format. The transform can accept PDF documents in color. However, it always creates a monochrome AFP image.

The transform creates an AFP Image Object Content Architecture (IOCA) image for each page in the PDF document. The image can be a compressed IOCA image (recommended for faster printing) or an uncompressed image. The transform can produce a printable AFP document, or an AFP overlay or page segment that you can print as part of other documents.

You can specify the type and resolution of the IOCA image, the height and width of the output page, and the type of document the transform creates in options on the transform command and in options in the printer definition.

2.12.4 PDF to AFP supported fonts

The PDF to AFP transform provides a set of fonts, which it uses to create a rasterized image of the data. For a list of the fonts that are provided with the transform, see Appendix B.1, “PDF to AFP and PostScript to AFP fonts” on page 200.

If a PDF document specifies a font that is not embedded in the PDF document and is not provided with the transform, the transform substitutes another font that is the best match for the specified font. You cannot customize the way the transform performs font substitution.

The administrator can add single-byte ASCII fonts to the transform.

2.12.5 PDF to AFP transform limitations

Each Infoprint transform has some limitations on what can be transformed. For PDF to AFP transform limitations, see C.7, “AFP architecture limitations for transforms” and C.7.2, “PDF to AFP limitations” on page 231.

2.12.6 PostScript to AFP transform

The PostScript to AFP transform converts documents in PostScript Language Level 3 format to AFP format. The transform can accept PostScript documents in color. However, it always creates a monochrome AFP image.

The transform creates an AFP Image Object Content Architecture (IOCA) image for each page in the PostScript document. The image can be a compressed IOCA image (recommended for faster printing) or an uncompressed image. The transform can produce a printable AFP document, or an AFP overlay or page segment that you can print as part of other documents.

You can specify the type, height, width, and resolution of the IOCA image, and the type of document the transform creates in options on the transform command and in options in the printer definition.
2.12.7 PostScript to AFP supported fonts

The PostScript to AFP transform provides a set of fonts, which it uses to create a rasterized image of the data. For a list of the fonts that are provided with the transform, see B.1, “PDF to AFP and PostScript to AFP fonts” on page 200.

If a PostScript document specifies a font that is not embedded in the PostScript document and is not provided with the transform, the transform substitutes another font that is the best match for the specified font. You cannot customize the way the transform performs font substitution.

The administrator can add single-byte ASCII fonts to the transform.

2.12.8 PostScript to AFP transform limitations

Each Infoprint transform has some limitations on what can be transformed. For PS to AFP transform limitations, see C.7, “AFP architecture limitations for transforms” on page 230 and C.7.3, “PS to AFP limitations” on page 231.

2.12.9 Environment variables for the PDF to AFP and PostScript to AFP

Environment variables let you specify:

- Directory that contains transform resources
- Record length of AFP images
- Tracing options

**Note:** All environment variables are optional. The environment variables you can specify are shown in Appendix C-3, “Environment variables for PDF and PS to AFP” on page 206 and Appendix C-4, “Environment variables added in z/OS V1R8” on page 207.

Example transform entries for the PDF to AFP and PostScript to AFP

This transform entry specifies a resource directory that your installation created to contain custom fonts. This transform does not create a transform class. Therefore, to use this transform, do not specify a transform class in the -c transform option. Specify a transform resource path as shown.

```plaintext
transform ps2afp
  start-command = ps2afpd
  maximum-active = 2
  maximum-idle-time = 300 # 5 minutes
  minimum-active = 1
  environment = {
    _BPX_JOBNAME -> PS2AFPD
    AOP_RESOURCE_PATH -> /etc/Printsrv/ps2afpv2/fonts
  }
;  
Figure 2-49  Transform for PS to AFP
```

This transform entry traces transform jobs and writes the trace to the /var/Printsrv/xfd directory. This transform entry creates transform class trace.
Chapter 2. Infoprint Server transforms

2.12.10 Starting Infoprint Server with sufficient region size

The region size available for Infoprint Server must be at least 10M larger than the number of bytes specified in the -m option in the transform configuration file. The -m option can be specified in the transform configuration file entries on the start-command attribute for the to-AFP transforms. The default for the -m option is 32M. Therefore, the minimum region size is 42MB. However, IBM recommends a region size of 256 MB or more so that you can transform large or complex data streams.

Depending on which method is used to start Infoprint Server, you can make sure that sufficiently storage is available, when you start the transform with:

**AOPSTART procedure** Specify a region size of 256MB in the REGION parameter on the EXEC statement in the AOPSTART JCL procedure. If the REGION parameter is not specified, the default region size defined for your installation is used.

**aopstart command** If you enter the aopstart command on the z/OS UNIX command line, make sure the MAXASSIZE parameter in the BPXPRMxx member of SYS1.PARMLIB is at least 256 MB. The MAXASSIZE parameter determines the system-wide maximum region size for an address space. (You can set the MAXASSIZE parameter dynamically with the SETOMVS command.)

**Examples:** This statement in the BPXPRMxx member sets the maximum region size to 256 MB:

```
MAXASSIZE(268435456) /* 256*1024*1024 = 256MB */
```

This command sets the maximum region size to 256 MB:

```
SETOMVS MAXASSIZE=268435456
```
If you enter the `aopstart` command on the TSO UNIX shell command line, make sure the `SIZE` parameter on the TSO/E LOGON panel specifies at least 256 MB. The `SIZE` parameter determines the maximum region size for an address space.

### 2.12.11 Adding fonts for PDF to AFP and PostScript to AFP

The PDF to AFP and PostScript to AFP transforms provide fonts in the `/usr/lpp/Printsrv/ps2afpv2/fonts` directory. If the PDF and PostScript files to be transformed use other fonts, you can add the fonts to the transforms. You can add font types as follows:

- **Type 0**
  - Type 0 is a “composite” font format. A composite font is composed of a high-level font that references multiple descendant fonts.
  - **CID** - CID Font (also known as CID-keyed Font, CID-based Font) is a PostScript font file format designed to address a large number of glyphs. CID-keyed font format can be used with the Type 1 font format for standard CID-keyed fonts, or Type 2 for CID-keyed OpenType® fonts.

- **Type 1**
  - Type 1 (also known as PostScript, PostScript Type 1, PS1, T1 or Adobe Type 1) is the font format for single-byte Roman fonts for use with Adobe Type Manager® software and with PostScript printers. It can support font hinting.

- **Type 2**
  - Type 2 is a character string format that offers a compact representation of the character description procedures in an outline font file. The format is designed to be used with the Compact Font Format (CFF).
  - CFF is designed to use less storage space than Type 1 fonts, by using operators with multiple arguments, various predefined default values, more efficient allotment of encoding values and shared subroutines among FontSet (family of fonts).

- **Type 3**
  - Type 3 font (also known as PostScript Type 3 or PS3, T3 or Adobe Type 3) consists of glyphs defined using the full PostScript language, rather than just a subset. Because of this, a Type 3 font can do some things that Type 1 fonts cannot do, such as specify shading, color, and fill patterns. However, it does not support hinting.

- **Type 4**
  - Type 4 is a format that was used to make fonts for printer font cartridges and for permanent storage on a printer’s hard disk. The character descriptions are expressed in the Type 1 format.

- **Type 42**
  - Type 42 font format is a TrueType font embedded in PostScript file format, which allows the file to print on PostScript-capable printers containing a TrueType rasterizer. Support for multi-byte CJK TrueType fonts is included.

- **MultiMaster fonts**

  - **MultiMaster** - Multiple master fonts (or MM fonts) are an extension to Adobe Systems’ Type 1 PostScript fonts, now mostly superseded by the advent of OpenType.
2.12.12 Steps for adding PDF and PostScript to AFP fonts

1. Create a directory for the fonts. Set the z/OS UNIX permissions so that the owner can read, write, and execute the directories, and everyone can read and execute them. For example:
   
   ```bash
   mkdir /etc/Printsrv/ps2afpv2
   chmod 755 /etc/Printsrv/ps2afpv2
   mkdir /etc/Printsrv/ps2afpv2/fonts
   chmod 755 /etc/Printsrv/ps2afpv2/fonts
   ```

2. Add fonts to the fonts directory created in step 1. Set the z/OS UNIX permissions so that the owner can read and write the font files, and everyone can read them. For example:
   
   ```bash
   chmod 644 /etc/Printsrv/ps2afpv2/fonts/*
   ```

3. (Optional) If the names of the font files are not the same as the font names, edit the Fontmap file. First, copy the file from directory `/usr/lpp/Printsrv/ps2afpv2/lib` to the fonts directory created in step 1. For example:
   
   ```bash
   cp /usr/lpp/Printsrv/ps2afpv2/lib/Fontmap /etc/Printsrv/ps2afpv2/fonts/Fontmap
   ```
   
   Then follow the instructions in the file to specify the names and file names of the fonts you added.

   The data in the Fontmap file must be in ASCII representation. To edit this file in EBCDIC on the z/OS system, you can use the `iconv` command to convert between ASCII and EBCDIC.

   The following are examples of how to use the `alias` command to create `a2e` and `e2a` commands that convert data between EBCDIC and ASCII code pages, and then use these commands to convert data in the Fontmap file:

   a. Convert data in Fontmap from ASCII to EBCDIC and create file `Fontmap.e`:
      
      ```bash
      alias a2e="iconv -f iso8859-1 -t ibm-1047"
      a2e Fontmap > Fontmap.e
      ```

   b. Edit file `Fontmap.e`.

   c. Convert data in `Fontmap.e` from EBCDIC to ASCII and replace data in file `Fontmap`:
      
      ```bash
      alias e2a="iconv -f ibm-1047 -t iso8859-1"
      e2a Fontmap.e > Fontmap
      ```

4. Specify the directory that contains the fonts you added and the modified Fontmap file in the AOP_RESOURCE_PATH environment variable in the transform configuration file, `aopxfd.conf`. For example:
   
   ```bash
   environment = {AOP_RESOURCE_PATH -> /etc/Printsrv/ps2afpv2/fonts}
   ```

Notes: Fonts must be in ASCII representation. Postscript font files in both PFA and PFB format are supported.

The font file name can have a suffix. However, a suffix is not required.

If the font file name is not exactly the same as the font name, you must specify the font file name in the Fontmap file. (If the font file name is the same as the font name, the transform finds the font even if the font name is not in the Fontmap file.)

The data in the Fontmap file must be in ASCII representation.
5. Restart the Transform Manager to pick up the changes to the Fontmap and aopxfd.conf files. For example, use the AOPSTOP and AOPSTART JCL procedures:

```
START AOPSTOP,OPTIONS=''-d xfd'
START AOPSTART
```

### 2.13 Customizing the SAP to AFP transform

Infoprint Server provides printing support for users and application programs in the SAP R/3 environment. Infoprint Transforms to AFP for z/OS, 5655-N60 lets you transform documents to Advanced Function Presentation (AFP) format from HP Printer Control Language (PCL), Adobe Portable Document Format (PDF), Adobe PostScript, or SAP R/3 System Generic Output Format (SAPGOF).

The SAP to AFP transform allows documents generated by SAP R/3 applications to be printed on high-speed IBM AFP printers with no changes to the SAP R/3 applications. Figure 2-51 on page 75 illustrates the steps described in the following process:

1. Using the SAP R/3 GUI, a user submits a document for printing on an SAP R/3 output device, which the SAP R/3 administrator has associated with a printer defined to Infoprint Server. The SAP R/3 application server converts the document to PCL format and sends it to the spool work process of the SAP R/3 Application Server for z/OS. The spool work process must run on the same system as Infoprint Server.

2. The SAP R/3 Application Server's spool work process submits the document (in PCL format) to the Infoprint Server SAP Output Management System (OMS), which is part of the Print Interface component.

3. Print Interface determines that the document must be transformed into AFP format for printing. Print Interface calls the PCL to AFP transform.

4. The PCL to AFP transform converts the data to AFP format.

5. Print Interface creates an output data set on the JES spool.

6. PSF selects the output data set from the JES spool and directs it to the selected AFP printer.

7. When the data set finishes printing or is deleted from the JES spool, the Print Interface SAP Callback daemon sends notification back to the SAP application server that submitted the print request. The notification indicates whether the data set printed successfully.
2.13.1 Customization tasks

All the SAP to AFP transform tasks are optional. These include:

- Accessing the aopsapd.conf configuration file - Optional
  
  This file should be moved to `/etc/Printsrv/aopsapd.conf` as shown in Figure 2-2 on page 30. This file is the default SAP Callback daemon configuration file. This file is required if the SAP Callback daemon receives a print request from SAP R/3. The file named in the AOPSAPD_CONF environment variable takes precedence over this file.

- Setting environment variables - Optional
- Installing multibyte conversion tables - Optional
- Customizing SAP to AFP configuration files - Optional

2.13.2 SAP to AFP transform customization

The SAP to AFP transform requires customization to be able to set up your printer definitions so that Infoprint Server automatically calls the SAP to AFP transform when it processes SAP documents. The SAP to AFP transform converts SAP R/3 Release 4.6C Output Text Format (OTF) and Advanced Business Application Programming (ABAP) documents to AFP format:

- SAP R/3 OTF data streams are converted into AFP Presentation Text Object Content Architecture (PTOCA) data streams.
- SAP R/3 ABAP data streams are converted into line data.
aopsapd.conf configuration file

The sample aopsapd.conf configuration file for the Infoprint Server SAP OMS V1.0 included with Infoprint Server for z/OS has one system entry required for each R/3 application server system to which the callback daemon is to connect.

The initial-target attribute names the server to which the callback daemon will initially connect for the system. This initial connection will be used to automatically retrieve configuration information for the Infoprint Server LOMSs configured on that system. Although the initial-target may be a non-z/OS server, it is necessary that the initialization instance for the ROMS and the tasking target for the LOMS all be located on an z/OS server where the Infoprint Server is running.

```
    system EZO
    initial-target = boefcs1_EZ0_00
    client = 000
    userid = mike
    password = xxxxxxx
    language = EN
    
    system EX0
    initial-target = boefcs5_EX0_00
    client = 000
    userid = mike
    password = xxxxxxx
    language = EN

Figure 2-52   aopsapd.conf transform entries
```

2.13.3 AFP resources for transform

The transform selects the appropriate AFP resources (form definition, page definition, and font) to use when PSF prints the document. The transform selects the resources based on the value of the PJFORM keyword in the header of the SAP data stream.

The default configuration files are in the /usr/lpp/Printsrv/sap2afp directory. If you choose to install the SAP to AFP configuration files in a different directory, you must specify the new directory in environment variable AOP_SAP2AFP_RESOURCES. All configuration files must be in the same directory.

2.13.4 SAP to AFP transform configuration files

Following are the SAP to AFP transform configuration files that can be customized:

- **barcode.tab**
  The barcode.tab configuration file contains a table that maps bar code names used by SAP to the matching bar codes defined by the AFP Bar Code Object Content Architecture (BCOCA).

- **defcp.tab**
  The defcp.tab configuration file contains a table that maps the Open Systems EBCDIC 1047 code page to the code page associated with the ABAP coded fonts specified in the pagedef.tab file.

- **fonts.tab**
  The fonts.tab configuration file contains a table that maps the fonts used in the SAP OTF data stream to AFP fonts.
image.tab  The image.tab configuration file contains a table that defines values used to print image data.

pagedef.tab  The pagedef.tab configuration file contains a table that maps SAP R/3 Format names to the names of AFP form definitions, AFP page definitions, and fonts. The SAP to AFP transform uses the form definition name when it transforms either SAP OTF or ABAP data. However, the transform uses the page definition and font names only for SAP ABAP data.

xxxx0000.tab  These configuration files contain tables that map characters of an individual SAP code page into an AFP code page.

Note: For more information, see IBM Infoprint Transforms to AFP for z/OS Version 2.1, G550-0443: Customizing SAP to AFP configuration files.

2.13.5 Supported input code pages for SAP to AFP transform

The SAP to AFP transform supports these input code pages:

- 0000 EBCDIC ISO-1 (Latin 1)
- 0120 EBCDIC ISO-1 (Latin 1)
- 1100 ASCII ISO 8859/1 (Latin 1)
- 4001 OCR-A (ASCII)
- 4004 OCR-B (ASCII)
- 8000 Japanese ISO Shift-JIS

For information about the 4-byte codes (such as 0120, 1100), see the SAP R/3 4.6C specifications.

2.13.6 SAP to AFP transform limitations

The following limitations apply when using the SAP to AFP transform:

- The transform does not produce color output. It produces monochrome output only.
- The transform cannot create FS45 image objects, which some IBM color printers require.

2.13.7 Setting environment variables for SAP to AFP

The AOP_SAP2AFP_RESOURCES environment variable specifies the directory that contains all of the SAP to AFP transform resources. If you need to set this environment variable, set it both in the aopstart EXEC and in the /etc/profile file.

<table>
<thead>
<tr>
<th>Environment variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AOP_SAP2AFP_RESOURCES</td>
<td>The directory that contains all of the SAP to AFP transform resources. You can specify only one directory name. If you installed Infoprint Server Transforms resources in the default directory, /usr/lpp/Printsrv/sap2afp, you do not need to set this environment variable. Default: /usr/lpp/Printsrv/sap2afp</td>
</tr>
</tbody>
</table>
2.13.8 Installing multibyte conversion tables

To transform and print SAP R/3 data streams that contain multibyte special characters, such as data streams produced by Asian versions of the SAP R/3 system, you must install binary conversion tables for the code pages used by the SAP to AFP transform in the /usr/lib/nls/locale/uconvTable directory. See “Steps for creating binary tables” for details.

The z/OS UNIX uconvdef command (Create binary conversion tables) can create the required conversion tables. You can use source files in the /usr/lpp/Printsrv/sap2afp directory as input to the uconvdef command. These source files define the mapping between UCS-2 and multibyte code sets, as follows:

**IBM-932-SAP2AFP.ucmap**
Defines the mapping for the SAP R/3 8000 (Japanese) system code page. The transform uses code page IBM-932-SAP2AFP as the iconv source code page.

**IBM-1030-SAP2AFP.ucmap**
Defines the mapping for the IBM-1030-SAP2AFP code page. This mapping is based on IBM-1027.ucmap, with the addition of the 11 box drawing characters. To use the IBM-1030-SAP2AFP code page, customize the 80000000.tab configuration file.

**IBM-1031-SAP2AFP.ucmap**
Defines the mapping for the IBM-1031-SAP2AFP code page, which the transform uses as the default output code page. This mapping is based on IBM-290.ucmap, with the addition of the 11 box drawing characters. You can change the output code page in the 80000000.tab configuration file.

2.13.9 Steps for creating binary tables

The uconvdef utility handles universal-coded character sets and creates binary conversion tables, uconvTable, that define a mapping between UCS-2 and multibyte code sets. The uconvdef utility reads the input source file to create the binary conversion table.

**Note:** UCS-2 is the Universal Multiple-Octet Coded Character Set defined by ISO/IEC 10646-1:1993(EE), while multibyte code sets consist of one or more bytes per character. uconvTable is in a format that can be opened and read by iconv conversion functions.

Perform the following steps from the OMVS shell:

1. Switch to an effective UID of 0 using the su command:
   ```
su
   ```
   **Note:** To use the z/OS UNIX su command, you must be permitted to the BPX.SUPERUSER FACILITY class profile in RACF.

2. Change to the /usr/lib/nls/locale/uconvTable directory:
   ```
cd /usr/lib/nls/locale/uconvTable
   ```

3. To create the compiled uconvTable that defines the conversion table between IBM-1047 and UCS-2, issue the following commands:
   ```
uconvdef -f /usr/lpp/Printsrv/sap2afp/IBM-932-SAP2AFP.ucmap -v IBM-932-SAP2AFP
uconvdef -f /usr/lpp/Printsrv/sap2afp/IBM-1030-SAP2AFP.ucmap -v IBM-1030-SAP2AFP
uconvdef -f /usr/lpp/Printsrv/sap2afp/IBM-1031-SAP2AFP.ucmap -v IBM-1031-SAP2AFP
```
4. Set the `p` extended attribute for the binary conversion tables:
   
   ```bash
   extattr +p IBM-*-SAP2AFP
   ```

5. Change the access permissions:
   
   ```bash
   chmod 644 IBM-*-SAP2AFP
   ```

6. If you switched to an effective UID of 0 in step 1, switch back to your own user ID:
   
   ```bash
   exit
   ```

   For more information about these z/OS UNIX commands, see the z/OS UNIX System Services Command Reference.

### 2.14 Customizing AFP to PCL transform

When Print Interface determines that a document must be transformed into PCL format for printing, Print Interface calls the AFP to PCL transform.

The AFP to PCL transform customization tasks include:

- Specifying transform options - Required
- Security for AFP resource libraries - Required
  
  See “Security for AFP resource libraries” on page 51.
- Adding paper sizes - Optional
  
  See “Adding paper sizes” on page 116.
- Adding fonts for font-mapping - Optional
  
  See “Adding fonts for font-mapping” on page 118.
- Scaling 240-pel to 300-pel fonts - Optional
  
  See “AOXCF30 program” on page 121.

#### 2.14.1 AFP to PCL transform

The AFP to PCL transform creates PCL 5, 5e, or 5c output. The transform can create monochrome or color output. You can print the PCL output on printers that support PCL 5, 5e, or 5c. A printer that supports PCL 5c is required to print color output.

The AFP to PCL transform converts AFP resources (resources can be inline in the AFP documents or in AFP resource libraries) into PCL format and includes them in the PCL data stream. In addition, the transform converts AFP formatting options, such as paper size and duplexing, to PCL commands. The interpretation of these PCL commands depends on your printer, and unpredictable results can occur if the AFP document requests formatting options that are not installed in the PCL printer.

The AFP to PCL transform uses 300-pel raster fonts (single-byte and double-byte). The transform includes all raster fonts in the PCL data stream to make sure they are available. The transform can map single-byte outline fonts to equivalent 300-pel raster fonts. Font-mapping supports single-byte fonts provided in the IBM AFP Font Collection (program number 5648–B33). The Infoprint Server administrator can customize font-mapping to add custom fonts.
AFP to PCL transform options

AFP to PCL transform options are specified in the aopxfd.conf transform configuration file. To define transform classes in the aopxfd.conf transform configuration file, see “Using transform classes” on page 57. The options include output page size, how to map AFP input tray numbers to paper sizes and PCL tray numbers, and whether to produce monochrome or color output.

Note: The sample transform configuration file, /usr/lpp/Printsrv/samples/aopxfd.conf, shows examples of transform entries with different transform classes.

Specifying transform options for AFP to PCL

Figure 2-53 on page 80 shows the sample afp2pcl_eu transform class in the /usr/lpp/Printsrv/samples/aopxfd.conf file.

```bash
# AFP to PCL transform with typical European paper sizes
# NOTE: This example only has a raster font library specified, because this
#       works for installations which do not have outline fonts installed.
#       But if you have outline fonts, it is recommended that you modify this
#       example to include both the raster and outline font libraries with
#       the AOP_FONTLIB variable.
# NOTE: An outline font library is needed for font mapping. If font mapping
#       is wanted, add your outline font library (e.g. SYS1.FONTOLN) to the
#       AOP_FONTLIB concatenation. If font mapping is not wanted, AOP_FONTMAP
#       should be set to "no".
transform afp2pcl_eu
   start-command = afp2pcld
   min-active = 0
   max-active = 2
   maximum-idle-time = 300 # 5 minutes
   environment = {
      AOP_CHARS -> 60D9
      AOP_COLOR -> no
      AOP_CUTSHEET -> no
      AOP_FONTLIB -> "sys1.font300"
      AOP_FONTMAP -> yes
      AOP_FORMDEF -> F1CP0110
      AOP_FORMDEFLIB -> "sys1.fdeflib"
      AOP_MSGFORMDEF -> F1CP0110
      AOP_MSGPAGEDEF -> "sys1.overlib"
      AOP_PAGEDEFLIB -> P1Q09182
      AOP_PAPER -> "a4 a3 a4 a4 a4 a4 a4 c5"
      AOP_PJL -> yes
      AOP_TRAYID -> "1 4 0 0 0 0 0 0 2"
      _BPX_JOBNAME -> AFP2PCLD
   }
```

Figure 2-53   AFP to PCL transform afp2pcl_eu transform entry

Transform classes for AFP to PCL

If you want to specify different transform options for different printers, you can create different classes of the AFP to PCL transform. For example, you could create classes for printers that print on different paper sizes. For each transform class, you must create a separate transform
entry in the aopxfd.conf file. You select a name for the transform class in the transform entry. See “Using transform classes” on page 57.

To use a transform class, job submitters specify the class name as an option (-c) on the afp2pcl transform command or in the filter-options job attribute. Administrators specify the class name as a filter option (-c) in the printer definition in the Printer Inventory.

For example, the z/OS UNIX afp2pcl command uses the eu transform class:

```
afp2pcl -c eu -o aopfa200 pcl pcl2.afp
```

### 2.14.2 Supported objects for AFP to PCL

See Appendix D.1.3, “Supported objects for AFP transforms” on page 235.

### 2.14.3 AFP to PCL transform limitations

There is a list of items in the AFP architecture that the AFP to PCL transform does not currently support. IBM suggests that you test how your AFP applications print on your non-AFP printers to make sure the output is acceptable. See C.7, “AFP architecture limitations for transforms” on page 230 and C.7.4, “Transform limitations” on page 232 for a list of items.

Other limitations for AFP to PCL

The transform might create bar codes that are larger or smaller than the same bar codes printed on AFP printers. The transform creates bar codes with a resolution of 300 pels per inch. Even though the size of bar codes might be different, the bar codes will scan properly.

The transform does not check all AFP structured fields in the AFP documents for validity. PSF might check additional structured fields for validity when it prints the AFP documents. Therefore, do not use the AFP to PCL transform to determine if PSF can print an AFP document.

### 2.14.4 Environment variables for the AFP to PCL transform

You can use environment variables to:

- Specify default AFP resources, such as the default form definition and page definition
- Specify AFP system resource libraries
- Select transform functions, such as whether to create color output
- Name the paper sizes in AFP input trays, and map AFP input tray IDs to PostScript input trays.

**Note:** All environment variables are optional. The environment variables are shown in C.3, “Environment variables for AFP to PCL” on page 208.

### 2.14.5 Default AFP resources for AFP to PCL

AFP resources are collections of data and control information that the transforms use to create PCL documents. You can specify default AFP resources (such as a default font, form definition, and page definition) in the transform configuration file transform entries and in printer definitions. AFP resources specified in a printer definition override default resources.
specified in transform entries. If only a few printers require unique default AFP resources, you might want to specify these in the printer definitions.

In most situations, you should specify the same default AFP resources that your AFP printers use. These default resources are specified either in the PSF startup procedures (in the PRINTDEV statements) or, if PSF is configured to obtain PRINTDEV values from the Printer Inventory, in the PSF FSA definitions in the Printer Inventory.

Note: See D.1.4, “AFP resources for transforms” on page 236 for details.

**AFP to PCL environment variables**

Environment variables that specify default AFP resources in the AFP to PCL transform configuration file AFP to PCL transform entries are as follows:

- **AOP_CHARS** The default font used for transform error messages and for line data and AFP data that does not specify another font.
- **AOP_FORMDEF** The default form definition used when no other form definition is specified.
- **AOP_PAGEDEF** The default page definition used when no other page definition is specified.

### 2.14.6 Execution of the transform

During the execution of the transform, it is necessary to determine where some of the resources to be used exist. Since the same resource can be specified in many different specifications, a search is done in a specific order to determine where the resource to be used is. This search is done for the following resources:

- Form definitions
- Page definitions
- Fonts to be used
- Message fonts

Attention: For a description of the search order for these resources, see C.6, “Search hierarchies for forms and page definitions” on page 228 and C.6.1, “Search hierarchy for the AFP resource libraries” on page 230.

### 2.14.7 Font-mapping options AFP to PCL

The AFP to PCL transform uses only 300-pel raster fonts (single-byte and double-byte fonts). The transform can map single-byte outline fonts to equivalent 300-pel raster fonts. Double-byte outline fonts are not supported.

To map outline fonts to raster fonts, the transform uses an internal font-mapping table that supports the Expanded Core Fonts feature of IBM AFP Font Collection V2, which provides single-byte fonts in both raster and outline formats. If the input document uses an outline font and the transform cannot find an equivalent raster font in the font-mapping table, the transform writes an error message and stops transforming the document.

You can customize the font-mapping table to add custom fonts used by your installation.
The AOP_FONTMAP environment variable controls font-mapping in the transform configuration file AFP to PCL transform entries:

**AOP_FONTMAP**

The **yes** option (default) enables font-mapping. You should enable font-mapping because the AFP to PCL transform cannot use outline fonts.

The **no** option disables font-mapping.

Font X060D9 is the default and is coded in the transform.

### 2.14.8 AFP system resources libraries AFP to PCL

In the transform entry, you can specify from 1 to 8 AFP system resource libraries for fonts, page definitions, form definitions, page segments, and overlays. You can specify from 1 to 8 AFP user resource libraries in printer definitions.

In most situations, you should specify the same AFP system resource libraries in the transform configuration file as you currently specify in your PSF startup procedures. If only a few printers require unique resource libraries, you can specify those libraries in the printer definitions for the printers.

The environment variables that specify system resource libraries in the AFP to PCL transform configuration file are:

- **AOP_FONTLIB**
  - The names of 1 to 8 system font libraries.
- **AOP_FORMDEFLIB**
  - The names of 1 to 8 system form definition libraries.
- **AOP_OVERLAYLIB**
  - The names of 1 to 8 system overlay libraries.
- **AOP_PAGEDEFLIB**
  - The names of 1 to 8 system page definition libraries.
- **AOP_PAGESEGLIB**
  - The names of 1 to 8 system page segment libraries.

AFP user resource libraries can also be specified in job attributes, JCL parameters on an OUTPUT JCL statement, and in the printer definition.

### 2.14.9 Examples -- Transform configuration file entries for the AFP to PCL

This section shows sample transform entries in the Infoprint Server aopxfd.conf configuration file for the APF to PCL transform.

### Print on letter and legal size paper

This transform entry, shown in Figure 2-54 on page 84, can be used for printers that print on letter and legal size paper. To use this entry, do not specify a transform class.
transform afp2pcl
    start-command = afp2pcl
    maximum-active = 2
    maximum-idle-time = 300 # 5 minutes
    minimum-active = 1
    environment = {
        _BPX_JOBNAME -> AFP2PCLD
        AOP_CHARS -> 60d9
        AOP_COLOR -> no
        AOP_CUTSHEET -> yes
        AOP_FONTLIB -> "sys1.font300"
        AOP_FONTMAP -> yes
        AOP_FORMDEF -> F1CP0110
        AOP_FORMDEFLIB -> "sys1.fdeflib"
        AOP_MSGFORMDEF -> F1CP0110
        AOP_MSGPAGEDEF -> P1P08682
        AOP_OVERLAYLIB -> "sys1.overlib"
        AOP_PAGEDEF -> P1P08682
        AOP_PAGEDEFLIB -> "sys1.pdeflib"
        AOP_PAGESEGLIB -> "sys1.pseglib"
        AOP_PAPER -> "letter legal letteree letter letter letter letter letter letter"
        AOP_PJL -> yes
        AOP_POSITIONING_METHOD-> cell
        AOP_TRAYID -> "1 4 1 1 1 1 1 1 1 1 2"
    };

Figure 2-54  Transform for AFP to PCL on letter and legal size paper
Print on A3, A4, and C5 paper
This transform entry, shown in Figure 2-55, can be used for printers that print on A3, A4, and C5 paper. This transform entry creates transform class eu.

```
transform afp2pcl_eu
   start-command = afp2pcl
d  maximum-active = 2
d  maximum-idle-time = 300 # 5 minutes
d  minimum-active = 1
environment = {
   AOP_CHARS -> 60d9
   AOP_COLOR -> no
   AOP_CUTSHEET -> yes
   AOP_FONTLIB -> "sys1.font300"
   AOP_FONTMAP -> yes
   AOP_FORMDEF -> F1CP0110
   AOP_FORMDEFLIB -> "sys1.fdeflib inst.fdeflib"
   AOP_MSGFORMDEF -> F1CP0110
   AOP_MSGPAGEDEF -> P1P08682
   AOP_OVERLAYLIB -> "sys1.overlib inst.overlib"
   AOP_PAGEDEF -> P1P08682
   AOP_PAGDEFLIB -> "sys1.pdeflib inst.pdeflib"
   AOP_PAGESEGLIB -> "sys1.pseglib inst.pseglib"
   AOP_PAPER -> "a4 a3 a4ee a4 a4 a4 a4 a4 c5"
   AOP_PJL -> yes
   AOP_POSITIONING_METHOD -> cell
   AOP_TRAYID -> "1 4 1 1 1 1 1 1 2"
} ;
```

Figure 2-55  Transform for AFP to PCL to print on A3, A4, and C5 paper

Transform and print a file
To use the eu transform class, specify the class in the -c transform option as shown in the following example, where the OMVS shell user transforms the file from AFP to PCL using the afp2pcl UNIX command and then prints the file using the lp command:

```
afp2pcl -c eu -o myfile.pcl myfile.afp
lp -o "filter-options='-c eu'" -d myprinter myfile.afp
```

2.15 Customizing the AFP to PDF transform

The AFP to PDF transform creates PDF 1.4 output. The transform always creates color output if the AFP document contains color images. For PDF document viewing and printing you can use Adobe Acrobat 5.0 (or higher), or a comparable PDF viewer.

The AFP to PDF transform converts AFP resources (resources can be inline in the AFP documents or in AFP resource libraries) into PDF format and includes them in the PDF data stream.

The AFP to PDF transform customization involves:
- Specifying transform options - Required
- Security for AFP resource libraries - Required

See “Security for AFP resource libraries” on page 51.
► Customizing Open Cryptographic Services Facility (OCSF) - Optional
  See “Customizing OCSF” on page 98 for PDF document encryption.
► Writing a Password exit - Optional
  See “Writing a Password exit” on page 98 for PDF document encryption with passwords.
► Adding paper sizes - Optional
  See “Adding paper sizes” on page 116.
► Adding fonts for font-mapping - Optional
  See “Adding fonts for font-mapping” on page 118.
► Scaling 240-pel to 300-pel fonts - Optional
  See “AOXCF30 program” on page 121.

Note: The transform does not include some AFP formatting options (such as copies, duplex, input and output bins, finishing, and copy marking and offset stacking (jogging)) in the PDF document. When you print PDF documents, you can use the printer driver options to select options such as copies, duplex, and so on.

With the AFP to PDF transform, the following functions and options are available:
► Uses 300-pel raster fonts (single-byte and double-byte) and AFP outline fonts (single-byte only).
► The transform includes all raster fonts in the PDF document to make sure they are available.
  The Infoprint Server administrator specifies whether to include outline fonts or only the names of outline fonts in the transformed PDF document.
  The size of PDF documents is reduced when only the names of outline fonts are included.
► The transform can map single-byte 300-pel raster fonts to equivalent outline fonts.
  Outline fonts provide superior viewing and printing.
► Font-mapping supports the Expanded Core Fonts feature of IBM AFP Font Collection (program number 5648–B33), which provides single-byte fonts in both raster and outline formats. The administrator can customize font-mapping to add custom fonts.

aopxfd configuration file
The transform options of the AFP to PDF transform are specified in the aopxfd.conf transform configuration file. The options include such characteristics as the output page size, whether to create enhanced PDF documents (enhanced PDF documents are easier to view and navigate), and whether to encrypt PDF documents. See “Using transform classes” on page 57.

2.15.1 Specifying transform options for AFP to PDF
To define transform classes in the aopxfd.conf transform configuration file, see “Using transform classes” on page 57. To activate the transform configuration file updates, you must restart the Infoprint Server transforms. After you restart the transforms, check for error messages in the transforms stderr file. If you find any error messages, fix the errors and restart the transforms. For more information about how to find the transform message logs, see “Infoprint Server transform error messages” on page 44.
Figure 2-56 shows the sample `afp2pdf_eu` transform class in the `/usr/lpp/Printsrv/samples/aopxfd.conf` file.

```plaintext
# AFP to PDF transform with typical European paper sizes
#
# NOTE: This example only has a raster font library specified, because this
#       works for installations which do not have outline fonts installed.
#       But if you have outline fonts, it is recommended that you modify this
#       example to include both the raster and outline font libraries with
#       the AOP_FONTLIB variable.
#
# NOTE: An outline font library is needed for font mapping. If font mapping
#       is wanted, add your outline font library (e.g. SYS1.FONTOLN) to the
#       AOP_FONTLIB concatenation. If font mapping is not wanted, AOP_FONTMAP
#       should be set to "no".
#
transform afp2pdf_eu
  start-command = afp2pdfd
  min-active = 0
  max-active = 2
  maximum-idle-time = 300 # 5 minutes
  environment = {  
    AOP_ANNOTATIONS -> no
    AOP_CHARS -> 60D9
    AOP_COLOR -> yes
    AOP_CUTSHEET -> no
    AOP_FLATE -> yes
    AOP_FONTLIB -> "sys1.font300 sys1.fontoln"
    AOP_FONTLIB -> "sys1.font300"
    AOP_FONTMAP -> yes
    AOP_FORMDEF -> F1CP0110
    AOPFORMDEFLIB -> "sys1.fdeflib"
    AOP_MSGFORMDEF -> F1CP0110
    AOP_MSGPAGEDEF -> P1Q09182
    AOP_OVERLAYLIB -> "sys1.overlib"
    AOP_OUTLINES -> yes
    AOP_PAGEDEF -> P1Q09182
    AOP_PAGEMAPDEFLIB -> "sys1.pdeflib"
    AOP_PAGEMAPDEFLIB -> "sys1.pseglib"
    AOP_PAPER -> "a4 a3 a4 a4 a4 a4 a4 a4 a4 c5"
    AOP_TRAYID -> "1 1 1 1 1 1 1 1 1 1"
    _BPX_JOBNAME -> AFP2PDFD
  }
```

Figure 2-56  AFP to PDF transform with typical European paper sizes

**Transform classes for AFP to PDF**

If you want to specify different sets of transform options for different uses, you can create different classes of the AFP to PDF transform. For example, you might create a separate class for documents that require encryption.

For each transform class, you must create a separate transform entry. You select a name for the transform class in the transform entry (see the transform attribute).

To use a transform class, job submitters must specify the class name in the `-c` option on the `afp2pdf` transform command or in the `filter-options` job attribute. Infoprint Server
administrators must specify the class name as a -c filter option in printer definitions. Examples of z/OS UNIX commands that use the encrypt transform class are:

afp2pdf -c encrypt -o myfile.pdf myfile.afp
lp -d myprinter -o "filter-options= -c encrypt " myfile.afp

Examples of transform entries with different transform classes are in the /usr/lpp/Printsrv/samples/aopxfd.conf transform configuration file.

2.15.2 Enhancing PDF documents

The AFP to PDF transform can create PDF documents that are easier to view and navigate, as follows:

- Create PDF bookmarks to help readers navigate through PDF documents. Adobe Acrobat displays PDF bookmarks in its navigation pane. For example, if the AFP document contains a table of contents, the transform converts it to a set of PDF bookmarks.

- Create PDF links to let readers jump to another location in a PDF document or to a Web site. For example, if the AFP document contains references to other locations, the transform converts them to PDF links. If the AFP document contains references to Web sites, the transform converts them to URLs that link to the Web sites.

- Optimize PDF documents for fast viewing from the Web. This means that Adobe Acrobat can display the first page of the PDF document before the entire document has been loaded from a Web site or a network.

- Rotate PDF documents for easier viewing. For example, some pages (such as those that contain tables) might require the PDF document to be turned sideways to be read. You can select auto-rotation so that each page is rotated in the same direction as the first character on the page.

2.15.3 Encrypting PDF documents

The AFP to PDF transform can encrypt PDF documents. Encrypting PDF documents provides enhanced security for sensitive documents that you send over a TCP/IP network. In addition, the transform can associate user and owner passwords with encrypted PDF documents to prevent unauthorized access, and it can restrict copying, updating, and printing of encrypted PDF documents.

Following are the PDF document encryption options:

- Encrypting with passwords

  You can associate a user password with the PDF document. The user password lets someone open an encrypted PDF document. A user password is optional. However, it prevents unauthorized users from opening PDF documents. In Adobe Acrobat, the user password is called an open password. You can associate an owner password with the PDF document. The owner password lets someone open an encrypted PDF document and bypass restrictions. An owner password is optional. However, it is required to restrict actions in PDF documents. In Adobe Acrobat, the owner password is called a permissions password. You can restrict actions in the PDF document, or you can allow all actions. Adobe Acrobat prevents users from doing restricted actions unless the user knows the owner password.

  You can select the level of encryption as follows:

  - A high level of encryption (a 128-bit encryption key) provides enhanced security.
A low level of encryption lets you send encrypted PDF documents to countries that do not use a high level of encryption or to users with earlier Adobe Acrobat versions.

**Encrypting without passwords**

Anyone can open the PDF document because no user password is associated with it. The transform generates an owner password that it uses to restrict actions in the PDF document. Because this owner password is secret, no one can use it to bypass restrictions. You must restrict at least one action in the encrypted PDF document. The transform uses a low level of encryption (a 40-bit encryption key). You cannot select the level of encryption. You might want to encrypt PDF documents without passwords so that the administrators do not need to maintain a password database.

**Specifying user and owner passwords**

For security reasons, job submitters cannot specify user and owner passwords during job submission, and administrators cannot specify passwords in printer definitions. Instead, job submitters and administrators specify user and owner identifiers.

The administrator can decide what identifiers to use. For example, identifiers can be z/OS user IDs, e-mail addresses, or a combination of different types of identifiers. Identifiers can contain any combination of 1 to 256 letters, numbers, blanks, and special characters.

The administrator must write a Password exit that returns a password to the transform for each user and owner identifier. The password exit can obtain these passwords from a password database. The password database can be in any format that your Password exit can use.

Job submitters can specify user and owner identifiers in job attributes `pdf-user-identifier` and `pdf-owner-identifier`, as shown in the following example when using the `afp2pdf` command:

```
afp2pdf -j "pdf-user-identifier=SMITH pdf-owner-identifier=LEE" -o myfile.pdf myfile.afp
```

As an alternative, the administrator can specify user and owner identifiers in printer definitions, by specifying them in the **AFP to PDF Transform Encryption** section of the Processing panel shown in Figure 2-57 on page 90.

**User identifier**

This field links the user to the user password. The user enters the user password when opening the encrypted PDF document. For example, a nurse encrypts a patient's test results with the AFP to PDF transform and e-mails them to the doctor as a PDF document that only the doctor can read. Enter the doctor's identifier in this field. If this field is blank, any user can open the PDF document without a password. The `pdf-user-identifier` job attribute overrides this field. The format can be any combination of letters, numbers, blanks, and special characters that the Password exit allows. This field might be case-sensitive, depending on the Password exit, for example, smith or drsmith@hospital.com.

**Owner identifier**

This field links the owner to the owner password. For example, a nurse encrypts a patient's test results with the AFP to PDF transform and e-mails them to the doctor as a PDF document that no one can change. Enter the nurse's identifier in this field. This field is required if you restrict actions in the Protected Actions field, unless the Password exit provides a default owner password. The `pdf-owner-identifier` job attribute overrides this field. The format can be any combination of letters, numbers, blanks, and special characters that the Password exit allows.
This field might be case-sensitive, depending on the Password exit, for example, lee or Nurse-Lee@hospital.com.

Restricting actions for encryption

When you encrypt PDF documents with or without passwords, you can restrict copying, updating, and printing in the PDF documents. Adobe Acrobat does not permit users to do the restricted actions when they open the PDF document. However, users who open the PDF document with the owner password bypass restrictions.

In Adobe Acrobat restricted actions are not available. For example, if you restrict printing, the Adobe Acrobat “Print” menu action is not available. To check what restricted menu actions Adobe Acrobat has made not available, open the transformed PDF. PDF viewers other than Adobe Acrobat might interpret restricted actions in different ways.

You can restrict slightly different sets of actions when you encrypt documents with and without passwords. In addition, the way you specify restricted actions differs.

Encrypting with passwords: When you encrypt PDF documents with passwords, job submitters can specify the restricted actions in the pdf-protect job attribute. For example, you can specify this afp2pdf command:

```
afp2pdf -j "pdf-user-identifier=SMITH pdf-owner-identifier=LEE
         pdf-protect={copy print update}" -o myfile.pdf myfile.afp
```

As an alternative, the administrator can specify restricted actions in printer definitions.

For more information about bits in the encryption dictionary, see the Adobe PDF Reference, which is available on the Adobe Web site.

Encrypting without passwords: When you encrypt PDF documents without passwords, the administrator must specify restricted actions in the AOP_PROTECT environment variable in the transform configuration file. For example, the administrator could create a transform class
called “nomodify” that restricts users from modifying the PDF documents. To do this, the administrator would specify this environment variable for the transform class:

\[ \text{AOP\_PROTECT} \rightarrow \text{"nomodify"} \]

When you encrypt PDF documents without passwords, job submitters cannot specify restricted actions. However, job submitters can submit transform jobs to the transform class that has the restrictions they want. For example, you can specify this \texttt{afp2pdf} command:

\[ \text{afp2pdf -c nomodify -o myfile.pdf myfile.afp} \]

As an alternative, the administrator can specify a transform class that restricts actions in printer definitions.

2.15.4 **Supported objects and controls for AFP to PDF**

See D.1.3, “Supported objects for AFP transforms” on page 235 and D.1.4, “AFP resources for transforms” on page 236 for more information.

2.15.5 **AFP to PDF transform limitations**


**Other transform limitations**

The transform cannot scale outline fonts. Therefore, the page definition you use for line-data and XML documents must specify a RATIO of 100 or omit the RATIO keyword.

The IBM Document Composition Facility (DCF) program produces AFP documents that contain BookMaster® fonts (this is the default). Because BookMaster fonts do not have equivalent outline fonts, the transform cannot map them to outline fonts. To improve the readability of the documents with Adobe Acrobat, IBM recommends that you use these two DCF options when you create AFP documents:

- \[ @\text{COREFNT}(\text{YES}) \]
- \[ @\text{BOOKFNT}(\text{3}) \]

These options tell DCF to create AFP documents that use raster fonts in the AFP Font Collection instead of BookMaster fonts. All raster fonts in the AFP Font Collection have outline equivalents.

**Using Adobe Acrobat**

There are some imitations on viewing and searching with Adobe Acrobat, as follows:

- Text that has been generated using AFP GOCA output graphic characters cannot be found.
- To preserve the output fidelity of a document, corresponding placement of characters is done in the PDF output document, which can result in additional “space” characters in a character string. This restricts the operation of the Adobe find function.
- The AFP representation of a code page might not match its ASCII representation. This can cause problems searching, especially with raster fonts.
- The printed output might be smaller or larger than expected. For example, the output might be 90% of the original size if the Fit to Page or Fit to paper option is selected in the Adobe Acrobat Print dialog. To solve this problem, deselect the option.
Transformed PDF images might look different from the original AFP image, depending on your monitor and printer.

The transform uses the AFP font encoding to create the PDF document. The characters contained in the document might not be the same as the ASCII character mapping available on the keyboard used for PDF viewing.

The actual appearance of raster fonts can differ from the printed output. For example, some characters might not be aligned on the character baseline. The appearance might change as higher magnification levels are chosen in Adobe Acrobat.

In some versions of Acrobat you might need to deselect **Use Greek text below xx pixels** to see all output.

PDF documents contain a unique page number identification that is assigned during the creation of the PDF document. This page number might not correspond to the page numbers used in the AFP input document.

Other limitations include:

- The transform might create bar codes that are larger or smaller than the same bar codes printed on AFP printers. The transform creates bar codes with a resolution of 300 pels per inch. Even though the size of bar codes might be different, the bar codes will scan properly.
- The transform does not check all AFP structured fields in the AFP documents for validity. PSF might check additional structured fields for validity when it prints the AFP documents. Therefore, do not use the AFP to PDF transform to determine if PSF can print an AFP document.
- The transform formats all output for the size of the paper in the first input tray selected.

### 2.15.6 Environment variables for the AFP to PDF transforms

Environment variables in the AFP to PDF transform entry include:

- Specify default AFP resources, such as the default form definition and page definition.
- Specify AFP system resource libraries.
- Select transform functions, such as whether to enable encryption.
- Name the paper sizes in AFP input trays.

**Note:** All environment variables are optional. Some specific environment variables are discussed in the next few sections. For a complete description of the environment variables for AFP to PDF transforms, see C.4, “Environment variables for AFP to PDF” on page 214.

### 2.15.7 Font-mapping options for AFP to PDF

The AFP to PDF transform can use 300-pel raster fonts (single-byte or double-byte) and AFP outline fonts (single-byte only). In addition, the transform can map single-byte 300-pel raster fonts to equivalent outline fonts. Outline fonts provide superior viewing and printing. Double-byte outline fonts are not supported.

To map raster fonts to outline fonts, the transform uses an internal font-mapping table that supports the **Expanded Core Fonts** feature of **IBM AFP Font Collection V2**, which provides single-byte fonts in both raster and outline formats. If the input document uses a raster font that does not have an equivalent outline font in the font-mapping table, the transform uses the
raster font. You can customize the font-mapping table to add custom fonts used by your installation. For more information, see “Adding fonts for font-mapping” on page 118.

Environment variables for font-mapping

The following environment variables control font-mapping in the afp2pdf transform in the configuration file.

**AOP_FONTMAP**

The **yes** option (default) enables font-mapping. You should enable font-mapping because outline fonts provide superior viewing and printing.

The **no** option disables font-mapping. If your installation has not installed outline font libraries, you can disable font-mapping to avoid error messages. If the transform does not find an outline font in the system font libraries, the transform automatically disables font-mapping for the raster font. However, the transform writes an error message in the transform error log once for each missing outline font.

**AOP_OUTLINES**

The **yes** option (default) causes the transform to include outline fonts in the output data.

The **builtin** option causes the transform to include only the names of outline fonts in the PDF document.

**Note:** Select the **builtin** option if you want to reduce the size of your PDF documents.

Environment variables to enhance viewing

The AFP to PDF transform lets you select options that enhance viewing and navigation in PDF documents. To select these options, use these environment variables:

**AOP_INDEX**

Creates bookmarks in PDF documents for improved navigation.

**AOP_INDEX_LANG**

Specifies the code page for the language for converting text in bookmarks.

**AOP_LINEARIZE**

Optimizes PDF documents for fast viewing from the Web.

**AOP_LINKS**

Creates links in PDF documents for improved navigation.

**AOP_ROTATE_PDF**

Specifies how to rotate PDF documents for easier viewing.

Environment variables for encryption

The AFP to PDF transform can encrypt PDF documents. For an overview of encryption, see “Encrypting PDF documents” on page 88. The environment variables that specify encryption options in the transform configuration file for AFP to PDF are:

**AOP_ENCRYPT**

Enables encryption. When you enable encryption, the transform encrypts documents if either of these conditions is met:

- The job submitter specifies a user or owner identifier in a job attribute, or submits a print job to a printer definition that specifies a user or owner identifier.

- The AOP_PROTECT environment variable is specified in the AFP to PDF transform class.

**AOP_PASSWORD_EXIT**

Names your Password exit and its optional arguments. The transform calls this exit to obtain PDF owner and user passwords when an owner or user identifier is specified.
This environment variable is used if encryption is enabled and a user or owner identifier is specified. Otherwise it is ignored.

**AOP_PROTECT**

Specifies the actions to be restricted in all PDF documents when no identifiers are specified. You can restrict copying, printing, and updating PDF documents.

This environment variable is ignored if either a user or owner identifier is specified.

Consider specifying the AOP_PROTECT environment variable in a transform class of its own and use this transform class only for documents you want to restrict actions on. You might want to set up several transform classes with different restrictions. For example, you could set up a class that restricts printing PDF documents, and another class that restricts changing PDF documents.

AFP resources are collections of data and control information that the transforms use to create PDF documents. You can specify default AFP resources (such as a default font, form definition, and page definition) in the transform configuration file transform entries and in printer definitions. AFP resources specified in a printer definition override default resources specified in the transform entry. If only a few printers require unique default AFP resources, you might want to specify these AFP resources in the printer definitions.

In most situations, you should specify the same default AFP resources that your AFP printers use. These default resources are specified either in the PSF startup procedures (in the PRINTDEV statements) or, if PSF is configured to obtain PRINTDEV values from the Printer Inventory, in the PSF FSA definitions in the Printer Inventory.

Environment variables that specify default AFP resources in the transform configuration file are:

- **AOP_CHARS**
  The default font used for transform error messages and for line data and AFP data that does not specify another font.

- **AOP_FORMDEF**
  The default form definition used when no other form definition is specified.

- **AOP_PAGEDEF**
  The default page definition used when no other page definition is specified.

**Note:** For a complete description of the AFP to PDF environment variables, see C.4, "Environment variables for AFP to PDF" on page 214.

### 2.15.8 Execution of the transform

During the execution of the transform, it is necessary to determine where some of the resources to be used exist. Since the same resource can be specified in many different specifications, a search is done in a specific order to determine where the resource to be used is. This search is done for the following resources:

- Form definitions
- Page definitions
- Fonts to be used
- Message fonts
2.15.9 AFP system resources libraries for AFP to PDF

In a AFP to PDF transform entry, you can specify from 1 to 8 AFP system resource libraries for fonts, page definitions, form definitions, page segments, and overlays. You can specify from 1 to 8 AFP user resource libraries in printer definitions. In most situations, you should specify the same AFP system resource libraries in the transform configuration file as you currently specify in your PSF startup procedures. If only a few printers require different resource libraries, you can specify those libraries in the printer definitions for the printers.

The following environment variables specify system resource libraries in the transform configuration file:

- **AOP_FONTLIB**  
The names of 1 to 8 system font libraries
- **AOP_FORMDEFLIB**  
The names of 1 to 8 system form definition libraries
- **AOP_OVERLAYLIB**  
The names of 1 to 8 system overlay libraries
- **AOP_PAGEDEFLIB**  
The names of 1 to 8 system page definition libraries
- **AOP_PAGESEGLIB**  
The names of 1 to 8 system page segment libraries

AFP user resource libraries can be specified in job attributes, JCL parameters, and in the printer definition.

Note: For a description of the search order for these resources, see C.6, “Search hierarchies for forms and page definitions” on page 228.

2.15.10 AFP to PDF transform entry afp2pdf

The afp2pdf command converts an Advanced Function Presentation (AFP) data file into an Adobe Portable Document Format (PDF) data stream file for printing or e-mailing.

Examples of AFP to PDF transforms

Following are two examples of AFP to PDF transforms:

1. Enhance viewing of PDF documents

This transform entry enables transform options that enhance viewing of PDF documents. The environment variables to enable these functions are shown in bold text. For more information about these environment variables, see “Environment variables to enhance viewing” on page 93.
transform afp2pdf
start-command = afp2pdfd
minimum-active = 1
maximum-active = 2
maximum-idle-time = 300 # 5 minutes
environment = {
  BPX_JOBNAME => AFP2PDFD
  AOP_ANNOTATIONS => yes
  AOP_CHARS => 60D9
  AOP_CUTSHEET => yes
  AOP_ENCRYPT => no
  AOP_FLATE => yes
  AOP_FONTLIB => "sys1.font300 sys1.font1n"
  AOP_FONTMAP => yes
  AOP_FORMDEF => F1CP0110
  AOP_FORMDEFLIB => "sys1.fdeflib"
  AOP_INDEX => yes
  AOP_INDEX_LANG => 037
  AOP_LINEARIZE => yes
  AOP_LINKS => yes
  AOP_MSGFORMDEF => F1CP0110
  AOP_MSGPAGEDEF => P1P08682
  AOP_OUTLINES => yes
  AOP_OVERLAYLIB => "sys1.overlib"
  AOP_PAGEDEF => P1P08682
  AOP_PAGEDEFLIB => "sys1.pdeflib"
  AOP_PAGESEGLIB => "sys1.pseglib"
  AOP_PAPER => "letter letter letter letter letter letter letter letter letter letter"
  AOP_POSITIONING_METHOD => cell
  AOP_ROTATE_PDF => auto
  AOP_TRAYID => "1 1 1 1 1 1 1 1 1 1"
}
;

Figure 2-58  AFP to PDF transform to enhance PDF documents
2. Encrypt PDF documents

This transform entry enables encryption and also restricts actions in all PDF documents. The environment variables used for encryption are shown in bold text. For more information about these environment variables, see “Environment variables for encryption” on page 93. This transform entry creates transform class encrypt.

```plaintext
transform afp2pdf_encrypt
  start-command = afp2pdfd
  minimum-active = 1
  maximum-active = 2
  maximum-idle-time = 300 # 5 minutes
  environment = {
_BPX_JOBNAME -> AFP2PDFD
  AOP_ANNOTATIONS -> yes
  AOP_CHARS -> 60D9
  AOP_CUTSHEET -> no
  AOP_ENCRYPT -> yes
  AOP_FLATE -> yes
  AOP_FONTLIB -> "sys1.font300 sys1.fontoln"
  AOP_FONTMAP -> yes
  AOP_FORMDEF -> F1CP0110
  AOP_FORMDEFLIB -> "sys1.fdeflib"
  AOP_INDEX -> yes AOP_INDEX_LANG -> 037
  AOP_LINEARIZE -> yes
  AOP_LINKS -> yes
  AOP_MSGFORMDEF -> F1CP0110
  AOP_MSGPAGEDEF -> P1P08682
  AOP_OUTLINES -> yes
  AOP_OVERLAYLIB -> "sys1.overlib"
  AOP_PAGEDEF -> P1P08682
  AOP_PAGEDEFLIB -> "sys1.pdeflib"
  AOP_PAGESEGLIB -> "sys1.pseglib"
  AOP_PAPER -> "letter letter letter letter letter letter letter letter letter"
  AOP_PASSWORD_EXIT -> "/usr/lpp/Printsrv/lib/aoppdfexit.dll"
  AOP_PROTECT -> "modify print select"
  AOP_POSITIONING_METHOD -> cell
  AOP_ROTATE_PDF -> no
  AOP_TRAYID -> "1 1 1 1 1 1 1 1 1"
  }
```

Figure 2-59  AFP to PDF transform for encryption of PDF documents

Using the AFP to PDF transform

To use the encrypt transform class, specify the class in the -c transform option as shown in these two examples:

afp2pdfd -c encrypt -o myfile.pdf myfile.afp
lp -o "filter-options='-c encrypt'" -d myprinter myfile.afp
2.15.11 Customizing OCSF

The OCSF architecture consists of a set of layered security services and associated programming interfaces designed to furnish an integrated set of information and communication security capabilities. Each layer builds on the more fundamental services of the layer directly below it. The OCSF implementation of the Common Data Security Architecture (CDSA) needs z/OS RACF (or an equivalent security product) to authorize the use of its services. The OCSF services are intended to be used by z/OS UNIX System Services based application servers or daemons.

To encrypt PDF documents, you must install and customize Open Cryptographic Services Facility (OCSF) with the Security Level 3 feature of OCSF. OCSF is part of z/OS Cryptographic Services.

For information about how to install and configure OCSF, see “Chapter 1. Configuring and Getting Started” in z/OS Open Cryptographic Services Facility Application Programming, SC24-5899.

OCSF requires that the user that starts the Infoprint Server daemons have READ access to following RACF FACILITY class resource profiles:
- BPX.SERVER
- CDS.CSSM
- CDS.CSSM.CRYPTO
- CDS.CSSM.DATALIB

Users who start Infoprint Server must be members of the AOPOPER RACF group, or have a UID of 0. (AOPOPER is the default group name for Infoprint Server operators. However, your installation can assign a different name to this group.) Therefore, you should give the AOPOPER group READ access to RACF profiles listed. If someone with a user ID of 0 who is not a member of the AOPOPER group can start Infoprint Server (for example, using the aopstart command), you must also give this user READ access to the profiles.

RACF commands

The following RACF commands are examples to give the AOPOPER group READ access to the required profiles, and refresh the FACILITY class:

```
PERMIT BPX.SERVER CLASS(FACILITY) ACCESS(READ) ID(AOPOPER)
PERMIT CDS.CSSM CLASS(FACILITY) ACCESS(READ) ID(AOPOPER)
PERMIT CDS.CSSM.CRYPTO CLASS(FACILITY) ACCESS(READ) ID(AOPOPER)
PERMIT CDS.CSSM.DATALIB CLASS(FACILITY) ACCESS(READ) ID(AOPOPER)
SETROPTS RACLIST(FACILITY) REFRESH
```

2.15.12 Writing a Password exit

PDF's standard security handler allows access permissions and up to two passwords to be specified for a document, as follows:

- An owner password
- A user password

The decision to encrypt a document is based on whether the user creating the document specifies any passwords or access restrictions. When the AFP to PDF transform encrypts a PDF document with passwords, a Password exit is required to provide passwords to the
transform. A Password exit is not required to encrypt PDF documents without passwords. For details, see “Environment variables for encryption” on page 93.

The two types of PDF passwords are:

**User password**  
Lets someone open an encrypted PDF document. A user password is optional. If a user password is not specified, anyone can open the encrypted PDF document.

Opening the document with the correct user password (or opening a document that does not have a user password) allows additional operations to be performed according to the user access permissions specified in the document's encryption dictionary.

**Owner password**  
Lets you restrict actions in an encrypted PDF document. The owner password also lets someone open an encrypted PDF document and bypass restrictions. An owner password is optional. However, it is required to restrict actions.

Opening the document with the correct owner password (assuming it is not the same as the user password) allows full (owner) access to the document. This unlimited access includes the ability to change the document’s passwords and access permissions.

**Password exit functions**
The functions the Password exit can provide are as follows:

- **Provide user and owner passwords:**

  The AFP to PDF transform passes one or two identifiers to the Password exit as input – a user identifier and an owner identifier. Typically, job submitters specify these identifiers in job attributes. However, the Infoprint Server administrator can also specify identifiers in printer definitions. Your installation can decide what identifiers to use. For example, identifiers can be z/OS user IDs, e-mail addresses, or a combination of different types of identifiers.

  The Password exit must return a password for each identifier that is passed as input to the exit. To map identifiers to passwords, your Password exit can use a password database in any format. For example, your Password exit can use a password database that already exists for other purposes. Or, you could create a password database that only your Password exit uses.

  **Note:** Your Password exit can return a password even when an identifier is not specified as input. For example, your Password exit could provide a default owner password when no owner identifier is specified.

- **Specify restricted actions:**

  This function is optional. The AFP to PDF transform passes a list of restricted actions to the Password exit as input. The job submitter specifies the restricted actions in a job attribute, or the administrator can specify them in the printer definition. If no restricted actions have been specified for a job, the default is not to restrict actions. This default is passed to the exit.

  Your Password exit can change the restricted actions. However, if your Password exit restricts actions, it must also return an owner password.
Programming considerations
When coding the Password exit, keep the following considerations in mind:

- The Password exit must be program-controlled.
- The Password exit runs with UID 0.
- Do not send a long-term explicit or implied WAIT in the Password exit.
- The Password exit runs in 31-bit addressing mode, in problem state.
- Code the Password exit to be reentrant.
- Programming exceptions cause Infoprint Server to abend, so test your Password exit carefully. An ESTAE is in effect while the exit is running.
- Future maintenance might require that you recompile the exit.

Input to the Password exit
When using the Password exit, the information available to use in the exit is as follows:

- Arguments (optional):
  You can define arguments in the AOP_PASSWORD_EXIT environment variable in the transform configuration file. For example, you might want to pass the name of the password database as an argument to your Password exit.

- User identifier (optional):
  The user identifier. This identifier can be specified either in the pdf-user-identifier job attribute or in the User identifier field in the printer definition of the AFP to PDF Transform Encryption data on the Processing panel. Although the user identifier is optional, the transform calls the Password exit only if a user identifier or owner identifier is specified.

- Owner identifier (optional):
  The owner identifier. This identifier can be specified either in the pdf-owner-identifier job attribute or in the Owner identifier field in the printer definition of the AFP to PDF Transform Encryption data on the Processing panel. Although the owner identifier is optional, the transform calls the Password exit only if a user identifier or owner identifier is specified.

- Protected actions (optional):
  A list of actions that are to be restricted in the PDF document. These restricted actions can be specified either in the pdf-protect job attribute or in the Protected actions field of the AFP to PDF Transform Encryption data in the printer definition.

Rules for identifiers and passwords
When referencing the identifiers, consider the following:

- The identifier can contain any combination of 1 to 256 letters, numbers, blanks, and special characters.
- The identifier is passed to the exit in EBCDIC representation (IBM-1047 code page).
- The identifier is passed to the exit with the same case as it was specified. However, your password exit could ignore the identifier's case. The sample Password exit does not ignore the identifier's case.
- Passwords can be 1 to 256 characters.
- Adobe Acrobat allows passwords that contain blanks or special characters. However, if you use the sample Password exit, passwords cannot contain blanks or the pound sign (#).
If a user identifier or owner identifier is input to the Password exit, it must return a password for the identifier or the job fails.

If your Password exit returns a list of protected actions, it must also return an owner password or the job fails.

Output from the Password exit
When exiting the exit, consider the following parameters:

- User password (optional): The password for the user identifier.
- Owner password (optional): The password for the owner identifier.
- Protected actions: A list of actions that are to be restricted in the PDF document.
- The user and owner passwords must not be the same.
- Passwords must be in ASCII representation because PDF viewers expect ASCII passwords. For example, use ASCII code page ISO8859-1.

Sample Password exit
IBM provides a sample Password exit and sample password database, as follows:

- /usr/lpp/Printsrv/lib/aoppdfexit.dll: This sample Password exit looks up passwords in the sample password database and returns a password for each identifier that is passed to it as input. It does not change the restricted actions. The sample Password exit accepts the name of the password database as an argument. If no argument is specified for the sample exit in the AOP_PASSWORD_EXIT environment variable, the exit looks for the database in /etc/Printsrv/aoppdfexit.db.
- /usr/lpp/Printsrv/samples/aoppdfexit.db: A sample password database that maps identifiers to passwords. To use the sample database with the sample exit, copy it to /etc/Printsrv/aoppdfexit.db because the sample exit looks for the password database in that location.

Files that contain the IBM-provided source code for the sample Password exit (aoppdfexit.dll):

- /usr/lpp/Printsrv/samples/aoppdfexit.h: A header file that contains declarations and interface descriptions.

Important: Do not change the contents of this file.

- /usr/lpp/Printsrv/samples/aoppdfexit.c: The source code for the sample Password exit.

Steps for writing and installing a Password exit
To code and install the Password exit, use the following steps:

1. Modify file /usr/lpp/Printsrv/samples/aoppdfexit.c, which contains the source code for the sample Password exit. File /usr/lpp/Printsrv/samples/aoppdfexit.h contains declarations and interface descriptions.

2. Link your Password exit as a dynamic link library (DLL). Follow instructions in /usr/lpp/Printsrv/samples/aoppdfexit.h.

3. Mark the UNIX file or MVS data set that contains your Password exit program-controlled.
   If the Password exit is in a UNIX file, use the z/OS UNIX extattr command to do this.
   Example:
   
   ```bash
   extattr +p /etc/Printsrv/mypdfexit.dll
   ```
To use the `+p` option of the `extattr` command, you must have at least READ access to the BPX.FILEATTR.PROGCTL FACILITY class profile.

4. Set the permissions so that the file is owned and executable by UID 0 and with no group or other permissions, as follows:

```
su
chown 0 /etc/Printsrv/mypdfexit.dll
chmod 700 /etc/Printsrv/mypdfexit.dll
```

To use the `chown` command, you must have an effective UID of 0. You can use the `su` command to switch to an effective UID of 0 if you are permitted to the BPX.SUPERUSER profile in the FACILITY class in RACF.

Alternatively you could use the UNIX System Services ISPF Shell dialog to set the exit program-controlled and set the permissions so that the file is owned and executable by UID 0 and with no group or other permissions.

5. Specify the name of the Password exit and optional arguments in the AOP_PASSWORD_EXIT environment variable in the transform configuration file, `aopxfd.conf`. For information, see “Environment variables for the AFP to PDF transforms” on page 92. An example of the environment variable is:

```
environment = {AOP_PASSWORD_EXIT -> "/etc/Printsrv/mypdfexit.dll"}
```

To edit the transform configuration file, you must have an effective UID of 0.

6. If you switched to an effective UID of 0 in step 4, switch back to your own user ID.

```
exit
```

7. Restart the Transform Interface to pick up the changes to the transform configuration file and to load your Password exit. For example, use the AOPSTOP and AOPSTART JCL procedures to stop and restart the transforms:

```
START AOPSTOP,OPTIONS='-d xfd'
START AOPSTART
```

If you modify the Password exit in the future, restart the transforms to reload it.

**Steps for writing a Password database**

1. Create the password database. If you want to use the sample password database, copy it from `/usr/lpp/Printsrv/samples/aoppdfexit.db` to another directory and follow instructions in the sample database to add the user identifiers, owner identifiers, and passwords for your installation.

```
su
  cp /usr/lpp/Printsrv/samples/aoppdfexit.db /etc/Printsrv/aoppdfexit.db
```

To copy the sample password database, you must have an effective UID of 0. You can use the `su` command to switch to an effective UID of 0 if you are permitted to the BPX.SUPERUSER profile in the FACILITY class in RACF.

2. Set the z/OS UNIX permissions of the database so that it is owned by UID 0 and with no group or other permissions.

```
su
chown 0 /etc/Printsrv/mypdfexit.db
chmod 700 /etc/Printsrv/mypdfexit.db
```

To use the `chown` command, you must have an effective UID of 0.

3. (Optional) If your password exit accepts the name of the password database as an argument, specify the argument in the AOP_PASSWORD_EXIT environment variable in the transform configuration file, `aopxfd.conf`. 
Example:

```plaintext
environment = {AOP_PASSWORD_EXIT -> "/etc/Printsrv/mypdfexit.dll \ 
/etc/Printsrv/mypdfexit.db"}
```

The backslash in this example indicates that the text within the quotation marks continues on the next line.

To edit the transform configuration file, you must have an effective UID of 0.

4. If you switched to an effective UID of 0 in step 1 or step 2, switch back to your own user ID.
```
exit
```

5. If you modified the transform configuration file, restart the Transform Interface to pick up the changes.

For example, use the AOPSTOP and AOPSTART JCL procedures to stop and restart the transforms.

If you modify the Password database without changing the transform configuration file, you do not need to restart the transforms.

## 2.16 Customizing the AFP to PostScript transform

The AFP to PostScript transform creates PostScript Language Level 3 output. The transform can create monochrome or color output. You can print the PostScript output on printers that support PostScript Language Level 3.

The AFP to PostScript transform converts all AFP resources (resources can be inline in the AFP documents or in AFP resource libraries) into PostScript format and includes them in the PostScript data stream. In addition, the transform converts formatting options in the AFP document, such as paper size and duplexing, to corresponding PostScript commands. The interpretation of these commands depends on your printer, and unpredictable results can occur if the AFP document requests formatting options that are not installed in the PostScript printer.

To customize the AFP to PostScript transform involves:

- Specifying transform options - Required
- Setting up security for AFP resource libraries - Required
  
  See “Security for AFP resource libraries” on page 51.
- Adding paper sizes - Optional
  
  See “Adding paper sizes” on page 116.
- Adding fonts for font-mapping - Optional
  
  See “Adding fonts for font-mapping” on page 118.
- Scaling 240-pel to 300-pel fonts - Optional
  
  See “AOXCF30 program” on page 121.

### 2.16.1 Transform classes AFP to PostScript

To define transform classes in the aopxfd.conf transform configuration file, see “Using transform classes” on page 57. If you need to specify different transform options for different
printers, you can create different classes for the AFP to PostScript transforms. For example, you could create classes for printers that print on different paper sizes.

Each transform class requires a separate transform entry. You give a name for the transform class in the transform entry (see the transform attribute).

To use a transform class, job submitters must specify the class name as a -c option on the `afp2ps` transform command or in the `filter-options` job attribute. The Infoprint Server administrators specify the class name as a `-c` filter option in the printer definitions. Examples on z/OS UNIX commands that use the `eu` transform class:

```
afp2ps -c eu -o myfile.ps myfile.afp
lp -d myprinter -o "filter-options='-c eu'" myfile.afp
```

Figure 2-60 is the sample `afp2_eu` transform entry.

```
#---------------------------------------------------------------
# AFP to PS transform with typical European paper sizes
#
# NOTE: This example only has a raster font library specified, because this
#       works for installations which do not have outline fonts installed.
#
# NOTE: An outline font library is needed for font mapping. If font mapping
#       is wanted, add your outline font library (e.g. SYS1.FONTOLN) to the
#       AOP_FONTLIB concatenation. If font mapping is not wanted, AOP_FONTMAP
#       should be set to "no".
#---------------------------------------------------------------
transform afp2ps_eu
  start-command = afp2psd
  min-active = 0
  max-active = 2
  maximum-idle-time = 300 # 5 minutes
  environment = {
    AOP_CHARS -> 60D9
    AOP_COLOR -> no
    AOP_CUTSHEET -> no
    AOP_FONTLIB -> "sys1.font300"
    AOP_FONTMAP -> yes
    AOP_FORMDEF -> F1CP0110
    AOP_FORMDEFLIB -> "sys1.fdeflib"
    AOP_MSGFORMDEF -> F1CP0110
    AOP_MSGPAGEDEF -> F1Q09182
    AOP_OVERLAYLIB -> "sys1.overlib"
    AOP_PAGEDEF -> F1Q09182
    AOP_PAGEDEFLIB -> "sys1.pdeflib"
    AOP_PAGESEGLIB -> "sys1.pseglib"
    AOP_PAPER -> "a4 a3 a4 a4 a4 a4 a4 a4 c5"
    AOP_TRAYID -> "1 2 0 0 0 0 0 0 2"
  _BPX_JOBNAME -> AFP2PSD
};
```

Figure 2-60  AFP to PS transform with typical European paper sizes

**AFP to PostScript transform**

The AFP to PostScript transform can use 300-pel raster fonts (single-byte and double-byte) and AFP outline fonts (single-byte only). The transform includes all fonts in the PostScript data stream to make sure they are available.
The transform can map single-byte 300-pel raster fonts to equivalent outline fonts. Outline fonts provide superior viewing and printing. Font-mapping supports single-byte fonts provided in the IBM AFP Font Collection (program number 5648-B33). You can customize font-mapping to add custom fonts.

The Infoprint Server administrator can specify transform options in the aopxfd.conf transform configuration file. The options include such characteristics as the output page size, how to map AFP input tray numbers to paper sizes and PostScript tray numbers, and whether to produce monochrome or color output.

Supported objects and controls for AFP to PostScript
For MO:DCA-P objects, see “Supported objects for AFP transforms” on page 235. For AFP resources, see “AFP resources for transforms” on page 236.

2.16.2 AFP to PostScript transform limitations


Following is the list of items in the AFP architecture that the AFP to PostScript transform does not currently support. Because the AFP architecture and PSF continue to be enhanced with new functional capabilities, this list might be incomplete after the publication date. IBM suggests that you test how your AFP applications print on your non-AFP printers to make sure the output is acceptable.

Limitations include the following:

- The transform might create bar codes that are larger or smaller than the same bar codes printed on AFP printers. The transform creates bar codes with a resolution of 300 pels per inch. Even though the size of bar codes might be different, the bar codes will scan properly.

- The transform does not check all AFP structured fields in the AFP documents for validity. PSF might check additional structured fields for validity when it prints the AFP documents. Therefore, do not use the AFP to PostScript transform to determine if PSF can print an AFP document.

2.16.3 Font-mapping options for AFP to PostScript

The AFP to PostScript transform can use 300-pel raster fonts (single-byte or double-byte) and AFP outline fonts (single-byte only). The transform can map single-byte 300-pel raster fonts to equivalent outline fonts. Outline fonts provide superior viewing and printing. Double-byte outline fonts are not supported.

To map raster fonts to outline fonts, the transform uses an internal font-mapping table that supports the Expanded Core Fonts feature of IBM AFP Font Collection V2, which provides single-byte fonts in both raster and outline formats. If the input document uses a raster font that does not have an equivalent outline font in the font-mapping table, the transform uses the raster font. You can customize the font-mapping table to add custom fonts used by your installation. For information, see “Adding fonts for font-mapping” on page 118.

Important: The AOP_FONTMAP environment variable controls font-mapping in the transform configuration file. For a description of this environment variable, see C.5, “Environment variables for AFP to PS” on page 223.
2.16.4 Environment variables for the AFP to PostScript transform

You can use environment variables to:

- Specify default AFP resources, such as the default form definition and page definition.
- Specify AFP system resource libraries.
- Select transform functions, such as whether to create color output.
- Name the paper sizes in AFP input trays, and map AFP input tray IDs to PostScript input trays.

**Note:** All environment variables are optional. For a complete description of the environment variables, see C.5, “Environment variables for AFP to PS” on page 223.

2.16.5 Default AFP resources for AFP to PostScript

AFP resources are collections of data and control information that the transforms use to create PostScript documents. You can specify default AFP resources (such as a default font, form definition, and page definition) in the transform configuration file and in printer definitions. AFP resources specified in a printer definition override default resources specified in the transform entries. If only a few printers require unique default AFP resources, you might specify these AFP resources in the printer definitions.

In most situations, you should specify the same default AFP resources that your AFP printers use. These default resources are specified either in the PSF startup procedures (in the PRINTDEV statements) or, if PSF is configured to obtain PRINTDEV values from the Printer Inventory, in the PSF FSA definitions in the Printer Inventory.

The environment variables to specify default AFP resources in the aopxfd.conf transform configuration file are:

- **AOP_CHARS** The default font used for transform error messages and for line data and AFP data that does not specify another font.
- **AOP_FORMDEF** The default form definition used when no other form definition is specified.
- **AOP_PAGEDEF** The default page definition used when no other page definition is specified.

2.16.6 Execution of the transform

During the execution of the transform, it is necessary to determine where some of the resources to be used exist. Since the same resource can be specified in many different specifications, a search is done in a specific order to determine where the resource to be used is. This search is done for the following resources:

- Form definitions
- Page definitions
- Fonts to be used
- Message fonts

**Attention:** For a description of the search order for these resources, see C.6, “Search hierarchies for forms and page definitions” on page 228.
**AFP system resources libraries for AFP to PostScript**

In the transform entry, you can specify from 1 to 8 AFP system resource libraries for fonts, page definitions, form definitions, page segments, and overlays. You can specify from 1 to 8 AFP user resource libraries in printer definitions.

In most situations, you should specify the same AFP system resource libraries in the transform configuration file as you currently specify in your PSF startup procedures. If only a few printers require different resource libraries, you can specify those libraries in the printer definitions for the printers.

Use these environment variables to specify system resource libraries in the transform configuration file:

- **AOP_FONTLIB** The names of 1 to 8 system font libraries.
- **AOP_FORMDEFLIB** The names of 1 to 8 system form definition libraries.
- **AOP_OVERLAYLIB** The names of 1 to 8 system overlay libraries.
- **AOP_PAGEDEFLIB** The names of 1 to 8 system page definition libraries.
- **AOP_PAGESEGLIB** The names of 1 to 8 system page segment libraries.

**Note:** For a complete description of these environment variables, see C.5, “Environment variables for AFP to PS” on page 223.

**Search order for AFP resources**

AFP user resource libraries can be specified in job attributes, JCL parameters, and in the printer definitions.

**Attention:** For a description of the search order for these resources, see C.6.1, “Search hierarchy for the AFP resource libraries” on page 230.

**AFP to PostScript transform entry afp2ps**

See “Using transform classes” on page 57 for the general format of a transform entry in the aopxfd.conf transform configuration file.

### 2.16.7 Sample transform configuration entries for AFP to PostScript

This section shows sample transform entries in the Infoprint Server transform configuration file (aopxfd.conf) for the AFP to PostScript transform.
Print on letter and legal size paper
This transform entry (Figure 2-61) can be used for printers that print on letter and legal size paper. To use this entry, do not specify a transform class.

```plaintext
transform afp2ps
  start-command = afp2psd
  minimum-active = 1
  maximum-active = 2
  maximum-idle-time = 300 # 5 minutes
  environment = {
    _BPX_JOBNAME -> AFP2PSD
    AOP_CHARS -> 60d9
    AOP_COLOR -> no
    AOP_CUTSHEET -> yes
    AOP_FONTLIB -> "sys1.font300"
    AOP_FONTMAP -> yes
    AOP_FORMDEF -> F1CP0110
    AOP_FORMDEFLIB -> "sys1.fdeflib"
    AOP_MSGFORMDEF -> F1CP0110
    AOP_MSGPAGDEFLIB -> P1P08682
    AOP_OVERLAYLIB -> "sys1.overlib"
    AOP_PAGEDEFLIB -> P1P08682
    AOP_PAGEDEF -> "sys1.pdeflib"
    AOP_PAGESEGLIB -> "sys1.pseglib"
    AOP_PAPER -> "letter legal letteree letter letter letter letter letter letter letter"
    AOP_POSITIONING_METHOD -> cell
    AOP_TRAYID -> "1 4 1 1 1 1 1 1 2"
  }
;
```

Figure 2-61  Transform to print on letter and legal size paper
Print on A3, A4, and C5 paper
This transform entry (Figure 2-62) can be used for printers that print on A3, A4, and C5 paper. This transform entry creates transform class eu.

```
start-command = afp2psd
minimum-active = 1
maximum-active = 2
maximum-idle-time = 300 # 5 minutes
environment = {
    _BPX_JOBNAME -> AFP2PSD
    AOP_CHARS -> "60d9"
    AOP_COLOR -> no
    AOP_CUTSHEET -> yes
    AOP_FONTLIB -> "sys1.font300"
    AOP_FONTMAP -> yes
    AOP_FORMDEF -> F1CP0110
    AOP_FORMDEFLIB -> "sys1.fdeflib"
    AOP_MSGFORMDEF -> F1CP0110
    AOP_MSGPAGEDEF -> P1P08682
    AOP_OVERLAYLIB -> "sys1.overlaylib"
    AOP_PAGEDEF -> P1P08682
    AOP_PAGEDEFLIB -> "sys1.pdeflib"
    AOP_PAGESEGLIB -> "sys1.pseglib"
    AOP_PAPER -> "a4 a3 a4ee a4 a4 a4 a4 c5"
    AOP_POSITIONING_METHOD -> cell
    AOP_TRAYID -> "1 1 1 1 1 1 1 2"
}
```

Figure 2-62 Transform to print an A3, A4, and C5 paper

To use the eu transform class, specify the class in the -c transform option as shown in examples as follows:

```
afp2ps -c eu -o myfile.ps myfile.afp
lp -o "filter-options='-c eu'" -d myprinter myfile.afp
```

2.17 Customizing the Infoprint XML Extender for z/OS

Extensible Markup Language (XML) provides a standard method for the exchange and processing of data across different computing platforms. XML data is structured according to standards from the World Wide Web Consortium (W3C). It is specifically designed for Web applications. XML differs from HTML in that, while HTML contains information about how data is formatted for presentation on Web pages, XML contains data with no presentation information. A W3C standard stylesheet language, called Extensible Stylesheet Language (XSL), uses formatting objects to describe how XML data is presented. XML data containing XSL formatting objects is called XSL-FO. XSL-FO describes document presentation details, such as pagination, layout, and styling information.

The XML Extender for z/OS lets you transform XML files to AFP or PDF format for printing or e-mailing. To transform XML files to another format, you can first transform XML to AFP, and then transform AFP to PCL or PostScript.
2.17.1 XML Extender transforms

XML Extender transforms XML data to an AFP or PDF data stream. Input to the XML Extender transforms is:

- XML files (or data sets), which do not contain data placement and presentation information, and an accompanying XSL style sheet
- XSL-FO files (or data sets), which contain XML data with XSL formatting objects

Figure 2-63 shows the transform process for XML data to AFP or PDF.

![Figure 2-63 Process for transforming XML data to AFP or PDF](image)

2.17.2 Infoprint Server customization tasks for XML Extender

To customize Infoprint Server to run XML Extender as a managed transform, the following tasks must be performed:

1. Specify a transform filter for the XML data format. Infoprint Server uses a filter program to transform XML data and then send it to an AFP printer or an e-mail destination. When the data you want to print or e-mail contains XML data, Infoprint Server calls the associated filter and invokes the transform before it writes data to the JES spool.

2. Customize the Infoprint Server Transform Interface. The Transform Interface is customized by editing the aopxfd.conf transform configuration file, which is shipped with Infoprint Server. The transform configuration file contains information that the Transform Manager uses to manage transform daemons.

The format of the XML to AFP transform entry

The XML to AFP transform shown in Figure 2-64 specifies the transform daemon and options. Enclose the value in single or double quotation marks if you specify any options.
Figure 2-64  Format of an XML to AFP transform

```
start-command = "xml2afpd [-T] [-d destinationtype] [-l pagelength[| mm]]
               [-s {stylesheet | none}] [-w pagewidth[| mm]]"
```

Where:

- **xml2afpd**
  The name of the XML Extender transform daemon. Specify the absolute path name (/usr/lpp/Printsrv/bin/xml2afpd) if the transform daemon is not in a directory named in the PATH environment variable.

- **-T, -d, -l, -s, -w**
  These options have the same meanings as the options for the `xml2afp` command. The options you specify are used by default as each input file is transformed. The administrator can override these options in the printer definition, or the job submitter can specify filter options to override the options.

  - **-T**
    requests a trace for diagnostic purposes. Trace output is written to stderr.

  - **-d destinationtype**
    indicates the type of destination, which affects the output generated. For example, different types of destinations can support different font technologies and different color specifications.

  - **-l pagelength[| mm]**
    indicates the length of the page on which the output is printed.

  - **-s {stylesheet | none}**
    specifies whether a style sheet is used to transform XML data to XSL-FO data or whether a specified style sheet is suppressed when the transform is managed by Infoprint Server.

  - **-w pagewidth[| mm]**
    indicates the width of the page on which the output is printed.

**The format of the XML to PDF transform entry**

An XML to PDF transform is shown in Figure 2-65 and specifies the transform daemon and options. Enclose the value in single or double quotation marks if you specify any options.

```
start-command = "xml2pdf [-T] [-s {stylesheet | none}]"
```

Figure 2-65  Format of an XML to PDF transform
Where:

\[\text{start-command} = \text{"xml2pdfd [-T] [-s \{stylesheet | none\}]"}\]

**xml2pdfd** The name of the XML Extender transform daemon. Specify the absolute path name (/usr/lpp/Printsrv/bin/xml2pdfd) if the transform daemon is not in a directory named in the PATH environment variable.

**-T, -s** These options have the same meanings as the options for the xml2pdf command. The options you specify are used by default as each input file is transformed. The administrator can override these options in the printer definition, or the job submitter can specify filter options to override the options.

- **-T** requests a trace for diagnostic purposes. Trace output is written to stderr.

- **-s \{stylesheet | none\}** specifies whether a style sheet is used to transform XML data to XSL-FO data or whether a specified style sheet is suppressed when the transform is managed by Infoprint Server.

3. Set up the security environment for the XML to AFP transform. Security checking performed in the transform requires that the user identifier (UID) of the XML Extender transform daemons, xml2afpd and xml2pdfd, are not 0 (zero). The installation of the product sets the daemon UIDs to 0.

4. Create the XML Extender configuration files, xml2afp.conf and xml2pdf.conf, which let you configure XML Extender. To create the files, copy the sample configuration files:

   `/usr/lpp/xml2afp/samples/xml2afp.conf` to `/etc/xml2afp.conf`
   `/usr/lpp/xml2afp/samples/xml2pdf.conf` to `/etc/xml2pdf.conf`

   The UNIX commands are:

   ```bash
   cp /usr/lpp/xml2afp/samples/xml2afp.conf /etc/xml2afp.conf
   cp /usr/lpp/xml2afp/samples/xml2pdf.conf /etc/xml2pdf.conf
   ```

   Edit the file using your preferred editor, such as `oedit /etc/xml2afp.conf`

   On the XML to AFP configuration file FONTLIB keyword, specify the location of the APF font libraries to be used by XML Extender, for example:

   ```
   FONTLIB="/SYS1.FONTOLN:/SYS1.FONT300"
   ```

   The XML to AFP transform requires the outline fonts.

5. Run the installation verification process (IVP).

**Security checking**

Security checking in the xml2afpd and xml2pdfd transforms requires that the user identifier (UID) of the executable files is not 0 (zero). Files xml2afp and xml2pdf are assigned a UID of 0 when it is installed. Therefore, you must change the owner of the file. The new owner must have other than a UID=0 or a default UID. Also, the set-user-ID flag for the file must be turned on.

### 2.17.3 Transform XML data to AFP or PDF

To transform XML data to AFP or PDF, XML Extender takes the input data you supplied and does the following:

- Uses the XSL style sheet to transform the XML files into XSL-FO and then completes the transformation to AFP or PDF.
An XSL style sheet processor accepts a document or data in XML and an XSL style sheet and produces the presentation of that XML source content that was intended by the designer of that style sheet. There are two aspects of this presentation process: first, constructing a result tree from the XML source tree and second, interpreting the result tree to produce formatted results suitable for presentation on a display, on paper, in speech, or onto other media. The first aspect is called tree transformation and the second is called formatting.

Tree transformation allows the structure of the result tree to be significantly different from the structure of the source tree. For example, one could add a table-of-contents as a filtered selection of an original source document, or one could rearrange source data into a sorted tabular presentation. In constructing the result tree, the tree transformation process also adds the information necessary to format that result tree.

- Directly transforms your supplied XSL-FO files to AFP or PDF.

### 2.17.4 XML Extender execution mode

When using XML Extender to transform files, you can use one of the following modes:

- A standalone transform
  
  You can use a batch JCL procedure or a z/OS UNIX System Services command to transform XML data to AFP.

- A transform managed by Infoprint Server
  
  When XML Extender is a managed transform, Infoprint Server automatically transforms XML data to an AFP or a PDF data stream. You have the option of submitting the XML data to a printer definition that the print administrator has configured to use XML Extender.

  If you submit the XML data to a printer definition, the transformed AFP data is automatically sent to an AFP printer and the transformed PDF data is automatically sent to an e-mail destination. Otherwise, the transformed data is available for viewing, archiving, or further transformation.

### 2.17.5 Using xml2afp.dll filter options in Infoprint Server

When XML Extender is managed by Infoprint Server, the administrator can associate a transform filter program with the XML data format in the printer definition. This association causes Infoprint Server to automatically call the filter program to convert the XML data to AFP before the data is written to the JES spool. To associate the filter program, the administrator specifies the xml2afp.dll transform DLL in the Filter field of the printer definition in the Printer Inventory. In the printer definition, the administrator can also specify filter options, such as the style sheet to use; the length and width of the output images; and the destination type and transform class to use. The job submitter can specify the same filter options in the filter-options job attribute when submitting print requests.

To use the transform filter for the XML data format specification for an AFP printer or an e-mail destination:

1. In the printer definition Filter field in the Printer Inventory specify xml2afp.dll or xml2pdf.dll. Type the absolute path name (such as /usr/lpp/Printsrv/lib/xml2afp.dll) if the filter is not in a directory named in the LIBPATH environment variable.

2. Specify filter options in the Filter field:
   - Transform class to use
   - Style sheet to use
– Length and width of the output images
– Destination type

For more information, see Infoprint XML Extender for z/OS, S544-5855.

2.17.6 Infoprint XT Extender for z/OS

Infoprint XT Extender is a z/OS product that transforms Xerox Line-Conditioned Data Stream (LCDS) jobs and metacode jobs into jobs that you can print on IBM IPDS printers. Intelligent Printer Data Stream (IPDS) is IBM's Systems Application Architecture® host-to-printer data stream for Advanced Function Presentation (AFP) subsystems. It provides an attachment-independent interface for controlling and managing all-points-addressable (APA) printers that allows the presentation of pages containing an architecturally unlimited mixture of different data types, including text, image, graphics, bar code, and object container.

Line Conditioned Data Stream (LCDS) is a line data stream used to drive Xerox Corp.'s (Rochester, NY) production printers from host systems. Unlike page description languages, which create pages from high-level graphical constructs, print command languages such as LCDS contain printer commands interspersed with data and are processed and executed sequentially.

Infoprint XT Extender converts the jobs and the resources they require, such as fonts and forms, into the AFP format. You can send your Xerox jobs directly to Infoprint XT Extender, without altering the applications that generate the jobs. Infoprint XT Extender converts the jobs, which you can then print on a wide variety of IBM IPDS printers.

For more information, see IBM Infoprint XT Extender for z/OS Customization and Usage, S544-5879.

2.18 Customizing the Infoprint Manager for AIX on z/OS

When processing a print request, Print Interface, NetSpool, and IP PrintWay extended mode call Transform Interface to transform data from one format to another if the administrator specifies a transform filter in the printer definition. The administrator can specify a different transform filter for each data format.

The transform filter aoprform.dll, provided by Infoprint Server, sends data to Infoprint Manager for AIX or Windows for transform.

Transform filter options for the aoprform.dll filter are:

```
```

Where:

%filter-options Causes any options that a job submitter specified in the filter-options job attribute to be passed directly to the transform. You can type %filter-options in any position relative to other filter options. If you specify any options to the right of %filter-options, those options override the same options that were specified in the filter-options job attribute, with the exception of any options that are cumulative.
-a imagetype The type of AFP data stream image that the transform generates for each page in the PCL, PostScript, or PDF file.

-l length This option is passed directly to the Infoprint Manager transform daemon. For the values you can specify, see the ps2afp or pcl2afp command description in IBM Infoprint Manager: Reference, S544-5475.

-P The TCP/IP port number on the AIX or Windows system at which the transform daemon is receiving data. Consult the AIX or Windows administrator for the correct value to specify. If you omit this option, the default port for the type of input data is used.

-p pagerange This option is passed directly to the Infoprint Manager transform daemon. For the values you can specify, see the ps2afp or pcl2afp command description in IBM Infoprint Manager: Reference.

-q transformattributes You can specify attributes when you select the FS45 image output (-a fs45). For a detailed description of these attributes and values, see IBM Infoprint Manager: Reference.

-r resolution This option is passed directly to the Infoprint Manager transform daemon. For the values you can specify, see the ps2afp or pcl2afp command description in IBM Infoprint Manager: Reference.

-t outputtype Determines the type of output to generate. Valid values are:

* document - Printable document. This is the default value.
* overlay - Graphic image that can be printed on each page of a printable document.
* pagesegment - Graphic image that can be embedded in a printable document.

w width This option is passed directly to the Infoprint Manager transform daemon. For the values you can specify, see the ps2afp or pcl2afp command description in IBM Infoprint Manager: Reference.

x xoffset This option is passed directly to the Infoprint Manager transform daemon. For the values you can specify, see the ps2afp or pcl2afp command description in IBM Infoprint Manager: Reference.

y yoffset This option is passed directly to the Infoprint Manager transform daemon. For the values you can specify, see the ps2afp or pcl2afp command description in IBM Infoprint Manager: Reference.

ipaddress The host name or dotted decimal address of the AIX or Windows system on which the transform daemon is running. This is a required option. You can specify the IP address in the dotted-decimal format. If the AIX or Windows system has an IPv6 address, you must specify its host name instead of its colon-hexadecimal format address.

**Note:** Infoprint Manager transforms, by default, create 32K records. To limit the maximum AFP record length to 8K bytes, the Infoprint Manager administrator must edit the Infoprint Manager configuration file.

### 2.18.1 Security for AFP resource libraries

See “Security for AFP resource libraries” on page 51.
2.19 Adding paper sizes

Table 2-2 summarizes the paper names and sizes that all transforms support. You can specify these paper names in the AOP_PAPER environment variable in the aopxfd.conf transform configuration file. Most values in the table are expressed in dots, based on 300 dots per inch (dpi). To convert values to inches, divide the values by 300. The origin of the X (width) and Y (height) coordinates is the top, left corner of the page.

Table 2-2  Paper names and dimensions

<table>
<thead>
<tr>
<th>Paper name</th>
<th>Paper size (in or mm)</th>
<th>Paper size (300 dpi)</th>
<th>Printable area origin (300 dpi)</th>
<th>Printable area size (300 dpi)</th>
<th>Logical page origin (300 dpi)</th>
<th>Logical page size (300 dpi)</th>
<th>PCL paper type ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>a3</td>
<td>292.25 x 413.25 mm</td>
<td>3507,4960</td>
<td>50,50</td>
<td>3407,4880</td>
<td>71,0</td>
<td>3365,4960</td>
<td>27</td>
</tr>
<tr>
<td>a4</td>
<td>210 x 297 mm</td>
<td>2489,3507</td>
<td>50,50</td>
<td>2380,3407</td>
<td>71,0</td>
<td>2338,3507</td>
<td>26</td>
</tr>
<tr>
<td>a4 ee*</td>
<td>210 x 297 mm</td>
<td>2489,3507</td>
<td>0,0</td>
<td>2480,3507</td>
<td>0,0</td>
<td>2480,3507</td>
<td>26</td>
</tr>
<tr>
<td>a5</td>
<td>148.3 x 210 mm</td>
<td>1748,2480</td>
<td>50,50</td>
<td>1648,2380</td>
<td>71,0</td>
<td>1606,2480</td>
<td>25</td>
</tr>
<tr>
<td>b4</td>
<td>250 x 363 mm</td>
<td>3035,4299</td>
<td>50,50</td>
<td>2935,4199</td>
<td>71,0</td>
<td>2893,4299</td>
<td>46</td>
</tr>
<tr>
<td>b5</td>
<td>176 x 250 mm</td>
<td>2149,3035</td>
<td>50,50</td>
<td>2049,2935</td>
<td>71,0</td>
<td>2007,3035</td>
<td>45</td>
</tr>
<tr>
<td>c5</td>
<td>162 x 229 mm</td>
<td>1913,2704</td>
<td>50,50</td>
<td>1813,2604</td>
<td>71,0</td>
<td>1771,2704</td>
<td>91</td>
</tr>
<tr>
<td>10</td>
<td>10 x 9.5 in. (254 x 241.3 mm)</td>
<td>1237,2850</td>
<td>50,50</td>
<td>1137,2750</td>
<td>75,0</td>
<td>1087,2850</td>
<td>81</td>
</tr>
<tr>
<td>dl</td>
<td>110 x 220 mm (4.4 x 8.8 in.)</td>
<td>1299,2598</td>
<td>50,50</td>
<td>1199,2498</td>
<td>71,0</td>
<td>1157,2598</td>
<td>90</td>
</tr>
<tr>
<td>executive</td>
<td>7.25 x 10.5 in. (185 x 267 mm)</td>
<td>2175,3150</td>
<td>50,50</td>
<td>2075,3050</td>
<td>75,0</td>
<td>2025,3150</td>
<td>1</td>
</tr>
<tr>
<td>ledger</td>
<td>11 x 17 in. (279 x 422 mm)</td>
<td>3300,1500</td>
<td>50,50</td>
<td>3200,1400</td>
<td>75,0</td>
<td>3150,1500</td>
<td>6</td>
</tr>
<tr>
<td>legal</td>
<td>8.5 x 14.0 in. (216 x 356 mm)</td>
<td>2559,4200</td>
<td>50,50</td>
<td>2450,4100</td>
<td>75,0</td>
<td>2400,4200</td>
<td>3</td>
</tr>
<tr>
<td>letter</td>
<td>8.5 x 11.0 in. (216 x 279 mm)</td>
<td>2559,3300</td>
<td>50,50</td>
<td>2450,3200</td>
<td>75,0</td>
<td>2400,3300</td>
<td>2</td>
</tr>
<tr>
<td>letteree*</td>
<td>8.5 x 11.0 in. (216 x 279 mm)</td>
<td>2559,3300</td>
<td>0,0</td>
<td>2550,3300</td>
<td>0,0</td>
<td>2550,3300</td>
<td>2</td>
</tr>
<tr>
<td>monarch</td>
<td>3.87 x 7.5 in. (98.4 x 190.5 mm)</td>
<td>1162,2250</td>
<td>50,50</td>
<td>1062,2150</td>
<td>75,0</td>
<td>1012,2250</td>
<td>80</td>
</tr>
</tbody>
</table>

* Use the a4ee and letteree paper names when the printer is configured for edge-to-edge printing. However, if the printer does not support edge-to-edge printing, documents created for edge-to-edge printing have the outside 50 pels, approximately 4 millimeters, of output cut off.

**Paper names and dimensions**

In Table 2-2, the table columns have the following meanings:

**Paper name**  The paper name, which you specify in the AOP_PAPER environment variable.

**Paper size (in or mm)**  The size in inches or millimeters of the entire media. The transforms use this value as the logical page to determine the area of the page that can be toned or printed.

**Paper size (300 dpi)**  The size in 300 dots per inch (dpi) of the entire media. The transforms use this value as the logical page to determine the area of the page that can be toned or printed.
Printable area origin (dpi)  The X (width) and Y (height) coordinates of the top left corner of the printable area in relation to the physical medium.

Printable area size (dpi) The area of the page that can be toned or printed.

Logical page origin (dpi) The X (width) and Y (height) coordinates of the top left corner of the logical page in relation to the physical medium.

Logical page size (300 dpi) The area of the page that can be addressed (positioned by the printer). The AFP to PCL transform uses this value.

PCL paper type ID The paper type identifier assigned by PCL and used to select this paper name.

For example, the afp2pcl_eu transform entry in Figure 2-53 on page 80 defines the AOP_PAPER -> a4, which matches the a4 paper name, European letter paper size, in Table 2-2.

**Steps for customizing the AOPPAPER table**

To customize the AOPPAPER table, do the following:

1. Copy the source code for the AOPPAPER table from SYS1.SAMPLIB(AOXPAPER) to a different data set.

2. To add a new paper size, code the PAPERGEN macro in AOXPAPER. In the PAPERGEN macro, you must specify:
   - The name of the paper entry. The name can contain 1 to 8 letters or numbers.
   - The width and length of the physical page in 300 dpi.
   - The origin (x and y coordinates) of the printable area in 300 dpi.
   - The width and length of the printable area in 300 dpi. The printable area is typically smaller than the size of the physical medium because many printers cannot print edge-to-edge.
   - The origin (x and y coordinates) of the logical page. Only the AFP to PCL transform uses this value.
   - The width and length of the logical page. Typically, the width of the logical page is equal to the width of the physical page minus 71 dots (at 300 dpi) at each end for European page formats or minus 75 dots for US page formats, and the length of the logical page is equal to the length of the physical page. Only the AFP to PCL transform uses this value.
   - The PCL paper type ID. See your printer manufacturer's documentation. Only the AFP to PCL transform uses this value.

3. Assemble and link the modified AOXPAPER module, and install the AOPPAPER table using SMP/E. IBM provides these sample usermods to install the AOPPAPER table. For instructions about installing the usermod, see the comments in the sample job.

   **Transform**  **Sample usermod**
   - AFP to PCL  SYS1.SAMPLIB(AOXPCLPJ)
   - AFP to PDF  SYS1.SAMPLIB(AOXPDFPJ)
   - AFP to PostScript  SYS1.SAMPLIB(AOXSPJ)

4. Specify the new paper name in the AOP_PAPER environment variable in the transform configuration file.

5. Restart the Transform Manager. For example, use the AOPSTOP and AOPSTART procedures to stop and restart the Transform Manager:
2.20 Adding fonts for font-mapping

The transforms use an internal font-mapping table to do the font-mapping function between raster and outline fonts. This font-mapping table supports all single-byte character sets included in the Expanded Core Fonts feature of the IBM AFP Font Collection V2 (program number 5648–B33), which are provided in both raster and outline formats.

The default font-mapping table is suitable for most installations. However, you might need to modify the table for these conditions:

- **All transforms:** If you have custom fonts in both raster and outline format, and you want the transform to substitute one font for another, you can add an entry in the table for each custom font.
- **AFP to PDF and AFP to PostScript transforms:** If you do not have AFP outline fonts for some character sets, you can delete the entries for these fonts from the table to avoid error messages. However, it is not necessary to delete the font entries. If the transform cannot find an outline font in the font libraries, it writes a warning message (AOX1109W CZxxx FONTLIB MEMBER NOT FOUND) to the transform’s stderr log and uses the raster font. You can ignore this message.

### AFP to PDF and PostScript transform limitations

These limitations apply to font-mapping:

- **All transforms:** You can map fonts only if positions 3-6 of the character set name for the outline and raster font are identical.
- **AFP to PCL transform:**
  - The font-mapping table does not support mapping asymmetric AFP outline fonts to raster fonts. Asymmetric AFP outline fonts are fonts that are anamorphically scaled in the Map Coded Font (MCF) structured field.
  - When a request to map an AFP outline font to a raster font results in the selection of a non-decimal point size, the font is mapped to the equivalent raster within one point size. If the corresponding raster font cannot be selected, the document is not transformed.

### Sample font-mapping table

Figure 2-66 on page 119 shows a portion of the default font-mapping table provided in SYS1.SAMPLIB(AOXFONTS). The table is identical to the default font-mapping table that the transforms use.
Outline fonts are resolution-independent; that is, they can be used by any printer that supports outline fonts. When a printer needs to use a given character (and not until then), it creates a rasterized bitmap from an outline font, scaling the character larger or smaller as needed (Figure 2-67 on page 120). A raster font character, by contrast, is stored and used in its full, rasterized size.
2.20.1 Modifying the font-mapping table

To modify the font-mapping table, do the following:

1. Edit the font-mapping entries in the default font-mapping table provided in
   SYS1.SAMPLIB(AOXFONTS). Do these steps when you add or delete an entry:
   
   – To add a font entry, specify positions 3 through 6 of the outline and raster character-set
     name. For example, Figure 2-3 shows the character set names for the Latin1 Gothic
     Text font and the value to specify in the font-mapping table.

   Table 2-3  Character set names for the Latin1 Gothic Text font

<table>
<thead>
<tr>
<th>Character set name of raster font</th>
<th>Character set name of outline font</th>
<th>Value in font-mapping table</th>
</tr>
</thead>
<tbody>
<tr>
<td>C06200xx</td>
<td>CZ6200</td>
<td>C'6200'</td>
</tr>
</tbody>
</table>

   The transforms use only a portion of the font character-set name to map fonts (that is,
   the type family, typeface, code page category, and complement), which are positions 3
   through 6 of the character set name. You do not need to include a separate entry for
   each point size. The transforms select the appropriate point size based on the input
   font.

   – Make sure that all entries in the table are in ascending order, with the smallest
     hexadecimal EBCDIC values first. For example, you would specify these fonts in the
     order shown:
     
     C 420P
     C 4200

     The hexadecimal EBCDIC value of C 420P (X'F4F2F0D7') is smaller than C 4200
     (X'F4F2F0F0').

   – Do not change the CSECT name (PSSFONTS) of the font table.

   Unexpected output results can occur if the font metrics of the source input font do not
   reflect the target font metrics.

2. Replace the font-mapping table in each transform that you want to use this table. In most
   situations, all transforms can use the same font-mapping table. IBM provides these
   sample usermods to replace the font-mapping table. For instructions about installing
   the usermod, see the comments in the sample job.

   – Transform AFP to PCL - Sample usermod SYS1.SAMPLIB(AOX2PCLJ)
2. Start the Transform Interface. For example, use the AOPSTOP and AOPSTART procedures to stop and restart the Transform Interface:

```
START AOPSTOP, OPTIONS= -d xfd
START AOPSTART
```

3. Check for error messages in the transforms stderr file. If you find any error messages, fix the errors and restart the transforms.

For more information about how to find the transform message logs, see Finding the transform stderr file on “Infoprint Server transform error messages” on page 44.

### 2.21 AOXCF30 program

The AOXCF30 font-conversion program lets you scale (that is, convert) your single-byte and double-byte, bounded-box, 240-pel fonts to 300-pel fonts. This program places the scaled fonts into a partitioned data set, which you can then use with the transform. You must scale 240-pel fonts to 300-pel fonts if you do not already have 300-pel fonts and either of these conditions applies:

- You use the AFP to PCL transform. This transform requires 300-pel raster fonts.
- You use the AFP to PDF or AFP to PostScript transform, AFP documents reference raster fonts, and you do not have outline fonts.

If you have previously scaled your 240-pel fonts using the PSF-supplied APSRCF30 program, you do not need to use the AOXCF30 program to scale them again. Because IBM Core Interchange raster fonts are shipped in both 240-pel and 300-pel resolutions, you do not need to convert them. Outline fonts are resolution-independent and do not need to be converted. The AOXCF30 program converts a 240-pel raster font to a close approximation of the font at 300-pel resolution. However, IBM does not guarantee the quality of the resulting character pattern when using this program. Specific fixed pitch fonts require the replacement of box characters. These are supplied in the AOX.SAOXC30 library.

#### 2.21.1 Using the AOXCF30 program

To use the AOXCF30 program, do the following:

1. Define a new font library for 300-pel fonts if one does not already exist. SYS1.FONT300 is the recommended name for your 300-pel system font library. You can use the existing 240-pel font library as a model. However, the 300-pel font library requires approximately 30% more space than the 240-pel font library.

2. Create a list of 240-pel fonts that require conversion. Only the character set members, whose file names start with C0, are converted with AOXCF30. However, you can also specify coded font members, whose file names start with X0, and code page members, whose file names start with T0. The coded fonts and code pages are copied to the output file without conversion.

3. Modify the JCL supplied in SYS1.SAMPLIB(AOXCF30J). Specify an entry in the JCL for each font member to be converted. The sample JCL is shipped with two dummy entries:

```
//C0XXXXXX EXEC PROC=SCALE, NAME=C0xxxxxx
//C0YYYYYY EXEC PROC=SCALE, NAME=C0yyyyyy
```
Duplicate these statements for each font that is to be converted. For example, specify:

```
C0A055A0 EXEC PROC=SCALE,NAME=C0A055A0
C0A055B0 EXEC PROC=SCALE,NAME=C0A055B0
```

Figure 2-68 shows the SYS1.SAMPLIB(AOXCF30J) member.

4. Submit the JCL to run the AOXCF30 program. This program can use a large amount of CPU resources. To run the AOXCF30 program, you should have a minimum region size of 5 MB.

**Note:** Do not convert the Bar Code or Optical Character Recognition fonts (program number 5688-021). The conversion process usually distorts the font, making it unreadable by an optical scanning device.

```plaintext
//AOXCF30J   JOB <job parameters>
//********************************************************************
//*  LICENSED MATERIALS - PROPERTY OF IBM                           *
//*  5697-F51                                                        *
//*  (C) COPYRIGHT IBM CORP. 2000                                   *
//*  AFP TRANSFORM (C) COPYRIGHT I-DATA INTERNATIONAL 1990,2000      *
//*                                                                  *
//*  AOXCF30 PROGRAM                                                 *
//*                                                                  *
//*  CONVERTS SBCS AND DBCS BOUNDED BOX FONTS FROM 240 TO 300 DPI    *
//*  USING A PROC INVOCATION OF THE AOXCF30 FONT SCALING PROGRAM.    *
//*  THE SUPPLIED PROC STATEMENTS ARE PROVIDED AS A SAMPLE ONLY, AND *
//*  SHOULD BE TAILORED TO MATCH INSTALLATION REQUIREMENT FOR        *
//*  SCALING OF FONTS TO 300 DPI.                                    *
//*                                                                  *
//*  BOX DRAWING CHARACTERS LOCATED IN FONTS CAN BE REPLACED USING    *
//*  THE BOX PARAMETER, AND SUPPLIED BOX DRAW CHARACTERS (FONTB300)  *
//*                                                                  *
//*  PARM='BOX' USE SUPPLIED BOX CHARACTER SET (DEFAULT)             *
//*  PARM='NOBOX' DO NOT USE THE SUPPLIED BOX CHARACTER SET          *
//*                                                                  *
//*  S2      EXEC PROC=SCALE,NAME=C0XXXXXX,PARM='BOX'                 *
//*                            (FONT IMAGE MEMBER NAME)              *
//*                                                                  *
//********************************************************************
//*
//SCALE     PROC HLQ=AOX,NAME=,PARM=BOX
//S1        EXEC PGM=AOXCF30,REGION=5120K,PARM='&PARM1'
//SYSPRINT DD SYSOUT=*  
//*    SYSUT1 IS THE INPUT 240 DPI FONT LIBRARY
//SYSUT1   DD DISP=SHR,DSN=SYS1.FONTLIBB(&NAME.)
//*    SYSUT2 IS THE OUTPUT 300 DPI FONT LIBRARY
//SYSUT2   DD DISP=SHR,DSN=SYS1.FONT300(&NAME.)
//FONTB300  DD DISP=SHR,DSN=HLQ..SAOXCF30  
// Pend
//*
//*
//C0XXXXXX EXEC PROC=SCALE,NAME=C0xxxxxx
//C0YYYYYY EXEC PROC=SCALE,NAME=C0yyyyyy
```

Figure 2-68  Sample JCL for the AOXCF30 font-scaling program
DD statements used by the AOXCF30 program

Check all the JCL for this font before running the AOXCF30 program.

DD statements for the AOXCF30 program are as follows:

- **SYSPRINT**  
  Program log and error message output
- **SYSUT1**  
  Input font library, including data set and member name
- **SYSUT2**  
  Output font library, including data set and member name
- **FONTB300**  
  Box character source library, which is supplied with the AOXCF30 program

The AOXCF30 program parameters

You can specify these parameters to the AOXCF30 program in the PARM1 parameter. The parameter are:

- **BOX**  
  Indicates that substitution of box characters will be done (default)
- **NOBOX**  
  Indicates that substitution of box characters will not be done

Box-character substitution is done with fixed-pitch fonts. When the BOX parameter is specified, the AOXCF30 program replaces box characters in the font with characters from the supplied AOX.SAOXCF30 library. To disable this substitution, specify the NOBOX parameter. For example, specify NOBOX when you scale custom built fonts, such as logo fonts.

AOXCF30 program return codes

The AOXCF30 program detects each of the errors listed below and sets a return code in register 15. When an error occurs, the font-conversion program might create an incomplete output file, which you should discard.

The return codes have the following meanings:

- **4**  
  The AOXCF30 program could not locate the box character member. Make sure that the FONTB300 DD name correctly specifies the SAOXCF30 library.
- **20**  
  The AOXCF30 program could not open the output data set, or an I/O error occurred while writing to the output data set.
- **28**  
  The AOXCF30 program could not open the input data set.
- **32**  
  The font named in the input data set is not valid. Make sure that the input font member is a valid 240-pel bounded box.
- **44**  
  An unexpected end-of-file occurred on the input data set. Make sure that the input data is not corrupted.
- **88**  
  Insufficient storage was available for program execution.
- **100**  
  The AOXCF30 program could not write to the SYSPRINT data set.

2.21.2 ITSOCF30 ISPF edit macro to generate JCL for AOXCF30 program

If you have to scale a large number of font members, you could use the ITSOCF30 ISPF edit macro to generate the AOXCF30 program’s JCL. See Appendix A.2, “ITSOCF30 ISPF edit macro” on page 189 for the ITSOCF30 REXX program.

To use the ITSOCF30 ISPF edit macro:

1. In ISPF invoke edit or view for an empty JCL sequential data set or a partition data set member.
2. On the edit panel's Command line enter `ITSOCF30 'font_library_name'` and press Enter. The `font_library_name` must be the same as what you will use for the font library data set name on the SYSUT1 DD-statement in the JCL SYS1.SAMPLIB(AOXCF30J).

3. Modify the JCL copied from the SYS1.SAMPLIB(AOXCF30J) member.
   - For each generated job add a valid JOB statement.
   - Update all SYSUT1 and SYSUT2 DD statements to specify correct font library data set names.

   **Note:** Change the DISP=SHR specification to DISP=OLD on the SYSUT2 DD statements. The generated job stream may consist of several jobs which all will update the same font library and thus cannot execute simultaneously.

   If Infoprint Server transforms are active, you must stop them to allow the generated jobs to execute.

4. Select the `//C0XXXXXX EXEC PROC=SCALE,NAME=C0xxxxxx` statements for each font that is to be converted.

5. Submit the generated jobs.

6. Restart the Infoprint Server transforms.
Specifying Infoprint Server transform daemons

Documents can be made up of different kinds of data, such as text, graphics, image, and bar code. Object content architecture (OCA) describes the structure and content of each type of data format that can exist in a document or appear in a data stream. Objects can be either data objects or resource objects.

A data object contains a single type of presentation data, that is, presentation text, vector graphics, raster image, or bar codes, and all of the controls required to present the data.

A resource object is a collection of presentation instructions and data. These objects are referenced by name in the presentation data stream and can be stored in system libraries so that multiple applications and the print server can use them.

This chapter describes:
- How to use filters in printer definitions
- How to administer the transform daemons
- How to start and stop transform daemons
3.1 Using transform filters

A transform filter is a program that transforms the documents to another data format. You can specify a different filter for each input data format that Infoprint Server supports: line data, JPEG, MO:DCA-P, PCL, PDF, PostScript, SAP, TIFF, XML, text, and other. After Infoprint Server detects the input data format of a document, it calls the transform filters automatically that you associated with that data format.

To transform data automatically, you specify a transform filter in the printer definition. A transform filter is a program that transforms the input data to another data format.

You can specify a different filter for each input data format that Infoprint Server supports as follows and as shown the Processing panel in the ISPF application to define printers in the Printer Inventory in Figure 3-1 on page 126.

![Processing panel used to define printers in the Printer Inventory](image)

3.1.1 Transform filter names

After Infoprint Server detects the input data format of a document, it calls the transform filter that you associated with that data format. The following lists the Infoprint transform products and the name of the transform filters that you can specify in the printer definition.

- Infoprint Transform for AFP to HP PCL for z/OS
  - afp2pcl.dll filter for AFP to PCL transform
- Infoprint Transform for AFP to Adobe PDF for z/OS
  - afp2pdf.dll filter for AFP to PDF transform
- Infoprint Transform for AFP to Adobe PostScript for z/OS
  - afp2ps.dll filter for AFP to PS transform
- Infoprint Transforms to AFP for z/OS
  - pcl2afp.dll filter for PCL to AFP transform which is shown in Figure 3-1 on page 126
  - ps2afp.dll filter for PDF to AFP and PS to AFP transforms
  - sap2afp.dll filter for SAP to AFP transform
- Infoprint Transform Manager for Linux
**aoprxf.so** filter to send documents to Linux for all transforms

- **Infoprint XML Extender for z/OS**
  - **xml2afp.dll** filter for XML to AFP transform
  - **xml2pdf.dll** filter for XML to PDF transform

- **Infoprint XT Extender for z/OS**
  - **x2afp.dll** filter for Xerox to AFP transform

- **Infoprint Manager for AIX (5785–E42) and Infoprint Manager for Windows (5639–I27)**
  - **aoprform.dll** filter to send documents to Infoprint Manager for AIX or Windows.

### 3.1.2 Printer definitions for Infoprint transforms and filters

A filter program can accept options which control the transform processing. If you are using the Infoprint Server Printer Inventory Manager panels to define printers, the transform filters can be specified on the IP PrintWay Printer Definition Processing panel.

**Specify printer filters**

Infoprint Server provides the following filters:

**%filter-options**

Causes options specified in the filter-options job attribute (specified, for example, on the `lp` command) to be passed to the transform. You can type the %filter-options option in any position relative to the other filter options. If you specify filter options to the right of %filter-options, those options override the same options specified in the filter-options job attribute.

- **aopfiltr.so** This filter prepares text data for printing on ASCII printers. This filter converts ASCII line-feed controls that are not preceded by carriage-return controls to carriage-return and line-feed controls (`X'0D0A'`). The `X'0D0A'` control is suitable for most ASCII printers.

  **Important:** Specify filter aopfiltr.so for the text data format in all IP PrintWay printer definitions except when you select the VTAM protocol or the e-mail protocol. Do not specify filter aopfiltr.so in a PSF printer definition.

- **lpd_compat.so** This filter formats text data and line data in a similar way to the z/OS TCP/IP LPD and creates line data. This filter is suitable for use with the line data and text data formats for printers that accept line data. The LPD compatibility filter, lpd_compat.so, provides support for some LPD command codes and parameters that the Print Interface LPD does not otherwise support. (LPD command codes and parameters are specified in the LPD control file sent by the LPR with each document to be printed.) This filter also lets you specify the `-f`, `-l`, and `-w` filter options, which correspond to the FILTER, LINECOUNT, and WIDTH parameters of the TCP/IP `lpr` command.

  **Important:** If the only filter you specify in the printer definition is lpd_compat.so, do not select the Resubmit for filtering field.

**Printer definitions for printer SYSLAB45**

Figure 3-2 on page 129 shows Panel ID AOPIPDRR and how to specify the aop2pcl.dll filter for data formats line and MO:DCA-P.

---

Chapter 3. Specifying Infoprint Server transform daemons 127
For each type of data format that Print Interface supports, you can specify the name of an associated filter. A filter is a program that can inspect and modify data before Print Interface writes the data to an output data set on the JES spool. When you specify the name of an filter for a supported data format in a printer definition, Print Interface automatically calls that filter before writing data to the JES spool.

The Help panel for the Filter field is also shown in Figure 3-2 on page 129.

When specifying the Data format and Filter in Figure 3-2 on page 129, the allowed values are specified in one or more value pairs in the format:

```
dataformat -> filterpath [options]
```

*filterpath [options]* specifies the path name of the filter, followed by filter options.

**Processing panel fields**

Following is an explanation of the AOPIPDRR panel fields for Supported Data Formats and Associated Filters fields:

- **Filter**
  The selected data formats are suitable for an IP Printway controlled IBM Infoprint 1154 printer:
  - Line data and MO:DCA-P are selected for AFP to PCL transformation.
  - For text data aopfiltr.so is selected to prepare text data for printing on ASCII printers. This filter converts ASCII line-feed controls that are not preceded by carriage-return controls to carriage-return and line-feed controls (X'0D0A'). The X'0D0A' control is suitable for most ASCII printers.
  - PostScript and PCL are selected because these data formats can be printed on the target printer without transformation.

- **%filter-options**
  Causes the transforms to use any options that the job submitter specifies in the filter-options job attribute. Because %filter-options is specified to the left of the duplex=yes option, the job submitter cannot override this option.

  If % operator in filter options definition, you can pass filter options and operands to your DLL or UNIX filter. The Infoprint Server administrator can specify filter options and operands in the printer definition, and the user can specify filter options in the filter-options job attribute.

  The administrator can use the special % operator to pass the value of a job attribute to the filter. For example, if a user specifies a job attribute on the **ip** command, the value of the attribute can be passed to the filter:

  ```
  %attribute_name
  ```

  - % causes the entire %attribute-name substring to be replaced by the value of the attribute.
  - attribute-name is the name of any job attribute, as listed in “IP Printway data transform processing” on page 17.

**Note:** Resubmit for filtering is selected because it indicates that a filter in any of the Filter fields is to be used for data sets submitted as batch jobs to IP PrintWay basic mode, which is the mode the printer is running. Resubmit for filtering applies to IP PrintWay basic mode only.
3.1.3 Filter examples

Specify the full path name unless the filter is in a directory named either in the LIBPATH (for DLL filters) or PATH (for UNIX filters) environment variable specified in the aopstart EXEC and in the /etc/profile file.

For a UNIX filter, type spawn before the path name. If the path name contains special characters (such as /), or if you specify filter options, enclose the path name and options in single or double quotation marks.

```
filters = {text -> "spawn /usr/mylib/mfilter myoption1 myoption2"}
```

**Default value**: Infoprint Server does not use any filter.

The filters shown in Figure 3-3 on page 129 are related to using filter lpd_compat.so.

```
/ Line data  lpd_compat.so -w 72 %filter-options (extend)
_ MO:DCA-P  lpd_compat.so -w 72 %filter-options (extend)
_ PostScript  lpd_compat.so -w 72 %filter-options (extend)
/ Text  lpd_compat.so -w 72 %filter-options (extend)
_ PCL  lpd_compat.so -w 72 %filter-options (extend)
```

**Figure 3-3 Example showing filter lpd_compat.so**
3.1.4 Using the PIDU command

The Printer Inventory Definition Utility (PIDU) program is used to manage objects in the Printer Inventory. Inventory objects include printer definitions, printer pool definitions, components, FSS definitions, FSA definitions, and job selection rules.

The PIDU program is useful for creating and editing many objects at the same time. Also, it lets you do some functions that you cannot do with Infoprint Server ISPF panels. For example, you can export or dump objects in the Printer Inventory to a file, and you can do more powerful searches of the Printer Inventory.

Note: The required authorization to use this utility is: (1) Have a UID of 0 or be a member of the AOPADMIN group and (2) have access to the AOPADMINSTRATOR resource profile in the RACF PRINTSRV class:

Print Interface and NetSpool invoke the specified filter before writing data to the output data set. IP PrintWay extended mode invokes the specified filter before sending data to the printer. To filter batch jobs submitted directly to IP PrintWay basic mode, specify the resubmit-for-filtering attribute.

In the pidu display, the multi-valued filters attribute associates data formats with a filter programs.
3.2 Starting and stopping transform daemons

If the transform runs on z/OS, the Infoprint Server Transform Manager (part of Transform Interface) starts and stops the transform daemons, using configuration information specified by the administrator.

Note: The Infoprint Server Transform Manager does not start the SAP to AFP transform because this transform is not implemented as a daemon.

The `aopd.conf` file contains attributes that customize the Infoprint Server. The default location of this configuration file is `/etc/Printsrv/aopd.conf`.

The `aopxfd.conf` file is optional. If the configuration file does not exist or if an attribute in the configuration file is omitted, default values are used.

```bash
start-daemons = {lpd}
```

Figure 3-4  **pidu** display of the SYSLBD45 printer's definition

# display printer SYSLBD45
printer-codepage = ISO8859-1
filters = {
  line -> "afp2pcl.dll -c sl %filter-options duplex=yes"
  modca -> "afp2pcl.dll -c sl %filter-options duplex=yes"
  text -> aopfiltr.so
}
destination = SYSLBD45
output-class = Y
error-disposition = default
print-error-reporting = all
print-queue-name = PASS
printer-ip-address = 11.22.33.44
document-formats-supported = {
  line
  modca
  postscript
  text
  pcl
  other
}
printer-type = ip-printway
forms = STD
duplexes-supported = {
  yes
}
document-codepage = IBM-1047
dcf-routing = yes
printway-pagination = suppress
lpr-print-function = l
lpr-print-banner = no
lpr-restrict-ports = yes
input-tray-number = 4
translate-document-header = no
duplex = yes
resubmit-for-filtering = yes
Infoprint Server Transform Manager

**xdf** is the Transform Interface aopxfd daemon. This daemon manages data stream transforms. To start the Infoprint Server Transform Manager, you must add or edit this attribute in the Infoprint Server configuration file, aopd.conf:

```plaintext
start-daemons = { xfd }
```

If transforms have not been used previously, add xfd to the existing values in this attribute to start the Infoprint Server Transform Manager daemon, aopxfd. Enclose all values in braces.

**Important:** After you edit this attribute and create the transform configuration file, restart Infoprint Server. You do not need to stop Infoprint Server before restarting. Also, if you run Infoprint Central, restart the z/OS HTTP Server.

### 3.2.1 aopstart command

The **aopstart** command starts Infoprint Server daemons that are specified in the aopd.conf configuration file. If a daemon is already active, aopstart does not start that daemon again.

**Format:** `aopstart`

The **aopstart** command does not restart a daemon that is running. To restart a daemon, first use the **aopstop** command to stop the daemon, and then enter the **aopstart** command.

To use the **aopstart** command, you must either be a member of the RACF AOPOPER group or have a UID of 0. You can use the z/OS UNIX `su` command to switch to an effective UID of 0 if you have READ access to the BPX.SUPERUSER profile in the RACF FACILITY class.

**Note:** The **aopstart** command always starts daemons aopd, aopsdbd, aophinvd, and aoplogd. The **aopstart** command also starts the optional daemons specified in the start-daemons attribute in the Infoprint Server configuration file.

**Starting daemons**

In TSO, you can use the TSO **OSHELL** or **OMVS** commands to run the **aopstart** command. You can invoke TSO commands from the ISPF Command Shell panel (ISPF option 6). To run the **aopstart** command, enter:

- oshell aopstart from the TSO command line
- omvs from the TSO command line and then, on the OMVS command line in the z/OS UNIX shell session, type aopstart and press Enter
- You can also use the TSO **ISHELL** command to enter the **aopstart** command.

To run the command from the ISHELL, you must enter the **full path name of the aopstart** command because the z/OS ISPF shell does not use environment variables set in the `/etc/profile` profile.

An example of the **aopstart** command from an ISPF shell is in Figure 3-5 on page 133 and Figure 3-6 on page 133.

1. If the **aopstart** command is in the default directory, select **Tools** from the action bar, select option 2 **Run shell command (SH)**
2. Issue the command as follows:

   /usr/lpp/Printsrv/bin/aopstart

3.2.2 aopstop command

The aopstop command stops all Infoprint Server daemons or only selected daemons. Unless you specify the now or the force option, the daemons stop after current activity completes.

Format: aopstop [-d daemon]... [now] [force]

-d daemon This option specifies the daemon to stop. If you omit this option, all active daemons are stopped, including the Printer Inventory Manager daemons (aopd, aophinvd, aoplogd, and aopsdbd). -d xfd stops the Transform Interface aopxfd daemon and all transforms.

now This option stops the daemons immediately. If work is in progress, incorrect output or data loss might result. If this option is not specified, the daemons stop after current activity completes.

force This option stops the daemons immediately using a "destructive" shutdown. If work is in progress, incorrect output or data loss might result. For daemons that
manage a database, such as aopd, aophinvd, or aopsdbd, the database might be corrupted. If this option is not specified, the daemons stop after current activity completes.

An example of stopping only the transforms:

```
aopstop -d xfd
```

To use the `aopstop` command, you must either be a member of the RACF AOPOPER group or have a UID of 0. You can use the z/OS UNIX `su` command to switch to an effective UID of 0 if you have READ access to the BPX.SUPERUSER profile in the RACF FACILITY class.

To stop all Infoprint Server daemons, enter the `aopstop` command without specifying any daemon names.

### 3.2.3 AOPSTART JCL procedure

The AOPSTART JCL procedure invokes the AOPBATCH program to execute the aopstart command to start Infoprint Server daemons.

To start Infoprint Server daemons with the AOPSTART JCL procedure, enter the MVS START command:

```
START AOPSTART
```

After you run the AOPSTART procedure, you see one or more messages in the AOSTART started task's STDOUT or STDERR data set, or in the MVS hardcopy log.

**Note:** The `aopmsg.conf` message configuration file lets you customize the Infoprint Server message processing. In the configuration file, you can enable the Infoprint Server hardcopy log function. The hardcopy log function sends Infoprint Server messages to the z/OS system hardcopy log so that you can use a z/OS message automation facility to process these messages.

If you do not enable the hardcopy log function, only messages that Infoprint Server sends to the console are sent to the hardcopy log.

### 3.2.4 AOPSTOP JCL procedure

The AOPSTOP JCL procedure invokes the AOPBATCH program to execute the aopstop command. The procedure accepts the OPTIONS symbolic parameter on the MVS start command. The OPTIONS parameter lets you to specify the Infoprint Server daemon to stop.

After you run the AOPSTOP procedure, you see one or more messages in the STDOUT or STDERR data set, or in the MVS hardcopy log.

To stop Infoprint Server daemons with the AOPSTOP JCL procedure, enter the MVS START command:

```
START AOPSTOP,[OPTIONS='-d daemon... [now] [force]']

-d daemon  This option specifies the daemon to stop. If you omit this option, all active daemons are stopped, including the Printer Inventory Manager daemons (aopd, aophinvd, aoplogd, and aopsdbd). The values you can specify for the daemon includes `xfd` to stop transforms.
now  This option stops the daemons immediately. If work is in progress, incorrect output or data loss might result. If this option is not specified, the daemons stop after current activity completes.

force  This option stops the daemons immediately using a "destructive" shutdown. If work is in progress, incorrect output or data loss might result. For daemons that manage a database, such as aopd, aophinvd, or aopsdbd, the database might be corrupted. If this option is not specified, the daemons stop after current activity completes.

An example of the MVS START command to stop only the transforms:

```
S AOPSTOP,OPTIONS='-d xfd'
```
Using transforms

IBM provides several separately priced Infoprint transform products that convert data from one format to another. Some of these products run on the z/OS system, while others run on other systems. The documentation for each transform product describes the supported functions and limitations of the transforms.

Usually, you do not have to worry about transforming your data to another format. If an IBM Infoprint transform product is installed, Infoprint Server automatically calls the appropriate transform when you submit a print request to a printer definition (for a printer or for an e-mail destination) that your administrator has configured for transformation.

This chapter provides z/OS Infoprint Server UNIX transform command usage guidelines. The sections in this chapter are:

- pcl2afp - Transform PCL data to AFP data
- pdf2afp - Transform PDF data to AFP data
- ps2afp - Transform PostScript data to AFP data
- sap2afp - Transform SAP OTF or ABAP data to AFP data
- Transforming data with the AOPBATCH program
- Using the Print Interface subsystem
4.1 Using Infoprint Server transforms

Infoprint Server works with several separate IBM transform products. It can automatically transform data streams from one format to another before the data is printed or sent to an e-mail destination. For example, you can use the PDF to AFP transform to convert documents in PDF format to AFP format. Transforms are programs that convert a data stream from one format to another. The IBM-provided transforms are implemented as DLL filters. These data stream transforms give you the flexibility to print a variety of output on a wide range of printers. The transforms can run automatically when associated as a filter for an IP PrintWay printer definition supported data format.

Additional transforms

Additionally, there are AFP to PCL, AFP to PDF, and AFP to PostScript transforms that are separately priced features of Infoprint Server Transforms. These can transform files in AFP or S/390® line data format to any of the following formats for printing on PCL or PostScript printers, or posting on the Web:

- PCL 5, 5e, or 5c (color)
- PostScript 1.2 (monochrome or color)
- PDF Language Level 2 (monochrome or color)

Infoprint XML Extender for z/OS

The Infoprint XML Extender for z/OS transforms Extensible Markup Language (XML) files to AFP format for printing or viewing. To transform XML files to non-AFP data streams, you can use this process:

- First transform XML to AFP. Then, you can transform the AFP file to another data stream.
- Transform an XML file to a PDF file.

Infoprint XML Extender for z/OS (5655-J66) is an optional product.

z/OS UNIX shell commands

z/OS UNIX Services users can run the transforms with the following z/OS UNIX System Services shell commands:

- pcl2afp  Transforms a PCL file to an AFP file
- pdf2afp  Transforms a PDF file to an AFP file
- ps2afp   Transforms a PostScript file to an AFP file
- sap2afp  Transforms an SAP ABAP or SAP OTF Version 1 or Version 2 file to an AFP file
- afp2pdf  Transforms an AFP or line data file to a PDF file
- afp2pcl  Transforms an AFP or line data file to a PCL file
- afp2ps   Transforms an AFP or line data file to a PostScript file
- xml2afp  Transforms an XML file to an AFP file
- xml2pdf  Transforms an XML file to a PDF file

Infoprint Server transforms also provide support for printing Japanese DBCS data streams on IBM AFP printers and for printing line data on coaxially-attached VTAM-controlled printers.

Infoprint XT Extender for z/OS

Infoprint XT Extender for z/OS, hereafter referred to as Infoprint XT Extender, is a data stream transform for the z/OS operating system. XT represents the IBM Xerox Transform
technology. The transform converts legacy application output into an AFP data stream that lets you take advantage of many of the benefits of AFP without changing your existing applications.

**Xerox print resources**

Xerox print resources such as fonts, forms, and images are converted into their AFP equivalents, and Xerox metacode and Line Conditioned Data Stream (LCDS) application data streams into the AFP MO:DCA (Mixed Object Document Content Architecture) data stream for printing on AFP/IPDS printers, or for use with the IBM Content Manager OnDemand archival/retrieval system. Once converted and stored in libraries identified to PSF, the AFP print resources are automatically accessed for printing on all of the printers that use them.

Administrators can set up the transforms to automatically transform documents before they are printed. Users can also use the z/OS UNIX command line to transform documents, which can then be saved in the converted format and later printed or sent to other users.

Figure 4-1 shows an LPR print request of a client data set from the workstation to the Infoprint Server's LPD daemon (AOPLPD). The Printer Inventory entry for the target AFP printer is specified such that the client data set will be transformed to AFP format for printing on the AFP printer.

The Print Interface subsystem uses Infoprint Server Transforms to provide filters that transform files in any of the following formats to AFP files for printing on AFP printers:

- PCL 5e
- PDF 1.2
- PostScript Language Level 3
- SAP ABAP versions 1 and 2
- SAP OTF versions 1 and 2

![Figure 4-1 Infoprint Server Transforms component](image-url)
4.2 Printing using Infoprint transform commands

IBM Infoprint transform products can convert files from AFP format into PCL, PDF, and PostScript. They let you print files in AFP format on PCL and PostScript printers, and transform an AFP file to PDF format for viewing on a workstation. z/OS Infoprint Server provides z/OS UNIX transform commands that allow data streams to be transformed from a z/OS UNIX shell session.

The transform commands are:

- pcl2afp
- pdf2afp
- ps2afp
- sap2afp
- afp2pcl
- afp2pdf
- afp2ps
- xml2afp
- xml2pdf

Files needed for transform commands

The following files are needed to execute any of the transform commands.

- `$HOME/.aopconf` Contains the user-specific Infoprint Server configuration file. This file takes precedence over `/etc/Printsrv/aopd.conf`.
- `/etc/Printsrv/aopd.conf` Contains the system default Infoprint Server configuration file.

**Note:** For the `sap2afp` transform command, the following customization files for the SAP to AFP transform are required:

- barcode.tab
- defcp.tab
- fonts.tab
- image.tab
- pagedef.tab
- xxxx0000.tab

Return codes for commands

The following return codes are issued when any of the transform command completes:

- 0 The data was transformed successfully. However, the output document might contain error messages related to errors in the input data stream.
- >0 An error occurred. No output document was created.

Environment variables for commands

All the transform commands use these environment variables:

- `AOPCONF` Names the Infoprint Server configuration file. This variable takes precedence over the user-specific configuration file (`$HOME/.aopconf`) and the system default configuration file (`/etc/Printsrv/aopd.conf`). For more information about the configuration file, see z/OS Infoprint Server Customization, S544-5744.
- `NLSPATH` Names the directory paths that the `pdf2afp` command searches for message catalogs.

The `sap2afp` command uses this environment variable in addition to the others:

- `AOP_SAP2AFP_RESOURCES` Specifies the directory that contains all of the SAP to AFP transform resources. The default value is `/usr/lpp/Printsrv/sap2afp`.

**Note:** For information about setting and using environment variables, see z/OS Infoprint Server User's Guide, S544-5746.

Using transform commands

In general, for all the transform commands, to position data on the page:

- Use `-l` and `-w` to set the physical page dimensions.
Use -x and -y to set the amount of white space between the physical page dimensions and the image.

These options do not shift or scale the image on the page. If the image is defined to print in the unprintable areas, it is cropped.

For example, to create a 6.5 x 9 inch image that is centered on an 8.5 x 11 inch page, enter:

```
-l 11in -w 8.5in -x 1in -y 1in
```

Use a form definition that specifies zero vertical offset and zero horizontal offset, or specify X and Y offsets of 0 when you submit the print job.

### 4.2.1 Printing with Infoprint Server

Infoprint transforms can be used to print files and data sets using the following methods:

- **Infoprint Server z/OS UNIX commands**
  
  Using the `lp` print command from an OMVS shell session, you can do the following:
  - Print UNIX files and MVS data sets on any printer that your administrator has defined in a printer definition in the Infoprint Server Printer Inventory. The printers can be local printers that are attached directly to z/OS, or remote printers in a TCP/IP or SNA network. Send UNIX files and MVS data sets to any e-mail destination that your administrator has defined in a printer definition in the Infoprint Server Printer Inventory.
  - Show printer names and locations and status of print jobs.
  - Cancel print jobs.

  **Note:** Print Interface includes enhanced z/OS UNIX System Services printing commands, which provide more function than the printing commands available without Infoprint Server. These modified commands conform to UNIX standards to facilitate the porting of UNIX applications to z/OS UNIX System Services:

  - **lp** - The `lp` command sends files for printing to Print Interface running on the local z/OS system. The files can be UNIX files or traditional MVS data sets, such as sequential data sets and partitioned data sets. This command conforms to the Single UNIX Specification Version 3 standard.

  - **lpstat** - The `lpstat` command queries the status of print jobs. It also queries the names, locations, and descriptions of printers that the administrator has defined in the Printer Inventory. This command conforms to the X/Open Portability Guide Issue 4 Version 2 (XPG4.2) standard.

  - **cancel** - The `cancel` command cancels print requests, provided that the data set allocated on the JES spool has not yet been selected for printing. This command conforms to the X/Open Portability Guide Issue 4 Version 2 (XPG4.2) standard.

- **Submitting batch jobs**
  
  When submitting a batch job, you can code the OUTPUT and DD statements in the JCL to use Infoprint Server to process an output data set. In the JCL parameters, you can print output using IP PrintWay and the output can be transformed and you can specify job attributes.
AOPPRINT JCL procedure

The AOPPRINT JCL procedure, provided in SYS1.PROCLIB, lets you submit print requests from z/OS. This procedure lets you take advantage of all the features of Infoprint Server as follows:

- You can specify job attributes.
- If an IBM Infoprint transform product is installed, you can automatically transform jobs from one data format to another.
- Infoprint Server validates that data can print on the selected printer.

AOPBATCH program

The AOPBATCH program lets you submit a batch job to transform data to and from the Advanced Function Presentation data format using the IBM Infoprint transform products. Infoprint Server provides the AOPBATCH program in SYS1.LINKLIB.

4.2.2 z/OS UNIX users and printing

To print from a z/OS UNIX shell, use the printing commands (lp, lpstat, and cancel) that Infoprint Server provides, as shown in Figure 4-4 on page 144. Using these commands, you can print MVS data sets and UNIX files, query the status of a print job, and cancel a print job. You can run these commands from the z/OS UNIX shell command line or from a UNIX application.

lp command

The lp command prints one or more files, or sends the files to an e-mail destination. The e-mail addresses are specified in the printer definition or in job attributes. The address of the printer is specified in the printer definition in the Infoprint Server Printer Inventory.

```
lp [-csw] [-d destination] [-n copies] [-o option] ... [-t title] [filename ...]
```

The files specified as the filename can be:

- MVS data sets, such as partitioned data sets or sequential data sets
- UNIX files, such as files in a Hierarchical File System (HFS), a z/Series File System (zFS), a Network File System (NFS), or a temporary file system (TFS)
- Lists of printable files

lpstat command options

The lpstat command shows printer names and locations and status of print jobs, as follows:

```
lpstat [-dt] [-a [printername ...]] ... [-o [printername ...]] ... [-p [printername ...]] ... [-u [userid ...]] ... [jobid ...]
```

lpstat returns printer definition names, location information specified in the printer definitions, and the status of jobs to standard output.

For jobs that Infoprint Server has processed (including jobs submitted in any of these ways: from a VTAM application through NetSpool; from a remote system; with the lp command through Print Interface; from batch JCL printed by IP PrintWay extended mode; or using the Print Interface subsystem), the lpstat command returns status information. Figure 4-2 on page 143 is an example of an lpstat command.
Figure 4-2 *lpstat* command to display names and locations of defined printers

<table>
<thead>
<tr>
<th>Printer</th>
<th>Jobs</th>
<th>Location</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NS</td>
<td>0</td>
<td>2C-16</td>
<td>VTAM NetSpool SCS Printer</td>
</tr>
<tr>
<td>PASSTHRU</td>
<td>0</td>
<td>U.S.A</td>
<td>Back to host</td>
</tr>
<tr>
<td>POK122B</td>
<td>0</td>
<td>POK 2-A30</td>
<td>IBM Infoprint Color 122B</td>
</tr>
<tr>
<td>POK45AN</td>
<td>0</td>
<td>POK 2-A30</td>
<td>1145A Portrait (Extended Mode)</td>
</tr>
<tr>
<td>POK45ANE</td>
<td>0</td>
<td>POK 2-A30</td>
<td>1145A Portrait (Extended Mode)</td>
</tr>
<tr>
<td>POK45AW</td>
<td>0</td>
<td>POK 2-A30</td>
<td>1145A Landscape/Rotated</td>
</tr>
<tr>
<td>POK45AWE</td>
<td>0</td>
<td>POK 2-A30</td>
<td>1145A Landscape/Rotated/Duplex</td>
</tr>
<tr>
<td>PR3287</td>
<td>0</td>
<td>SYSLAB</td>
<td>Test for OSA-ICC</td>
</tr>
<tr>
<td>ROSS</td>
<td>0</td>
<td>POK 2-A30</td>
<td>1145A Landscape/Rotated</td>
</tr>
</tbody>
</table>

Figure 4-3 shows the *lpstat* command options.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-d</td>
<td>Query default printer</td>
</tr>
<tr>
<td>-o</td>
<td>Query specified printer and jobs</td>
</tr>
<tr>
<td>-p</td>
<td>Query specified printer</td>
</tr>
<tr>
<td>-t</td>
<td>Query all printers and jobs</td>
</tr>
<tr>
<td>-u</td>
<td>Query all printers and jobs by user ID</td>
</tr>
<tr>
<td>-a</td>
<td>Query names and locations of all printers</td>
</tr>
</tbody>
</table>

Figure 4-3 *lpstat* command options

**cancel command**

The *cancel* command cancels a print job.

    cancel jobid ...

The *cancel* command cancels one or more print jobs that you submitted, with these restrictions:

- You can only cancel your own jobs.
- You cannot cancel a job after it has started processing.
- In a JES3 environment, you might not be able to cancel a job that is held on the spool.
Attribute files
You can predefine attribute and value pairs in permanent files. You can use the permanent attribute files when you need those attribute values with the `lp`, `afp2pcl`, `afp2pdf`, and `afp2ps` commands. You can also use the permanent attribute files when you need to specify those values with the AOPPRINT JCL procedure and when you use the Print Interface subsystem.

For example, you could create an attributes file called `myatts` to request 5 copies of a job, simple duplex printing, and a specific output bin. Your file contains these lines:

```
# These are myatts
copies = 5
duplex = yes
output-bin = collator # Collate the job
```

Use the `-o` option to read an attributes file into the `lp` command. Use the `-j` option to read an attributes file into the `afp2pcl`, `afp2pdf`, and `afp2ps` commands.

For example, to print a file called `myfile`, using the attributes in the `myatts` file, enter:

```
lp -o "attributes=myatts" myfile
```

This command is equivalent to this command:

```
lp -o "copies=5 duplex=yes output-bin=collator" myfile
```

Default printer for z/OS UNIX users
The default Infoprint Server printer definition is used in these situations:

`lp` command
When the job submitter does not specify a printer definition on the `lp` command and the PRINTER and LPDEST environment variables are not set
Print Interface subsystem
When the printer definition is omitted in the SUBSYS parameter on the DD JCL statement and the FSSDATA parameter is not specified on the OUTPUT JCL statement.

Environment variables
These environment variables affect the behavior of these commands. You can set the environment variables in the /etc/profile file so that they apply for all users of the lp, lpstat, and cancel commands.

AOPOPTIONS
Specifies a string of job attributes and values that are to be in effect for each lp command. The lp command includes the attributes specified in this environment variable before the attributes specified, if any, on the -o option of the lp command. Because the lp command reads the value of the AOPOPTIONS environment variable before the options you specify on the command line, a user can override the values of this variable. This variable is optional. Use this environment variable to specify job attributes that are constant for all print jobs.

AOPATH
The directory that contains attribute files used by the lp command. This variable is optional. If you have not created attribute files for the lp command, you do not need to set this variable.

MANPATH
The path of directories that contain the man pages. Infoprint Server man pages are available only in English. Add /usr/lpp/Printsrv/man/En_US to the values in this variable. This variable is required.

PRINTER or LPDEST variables
The PRINTER or LPDEST variable specifies the default printer for the lp command. The value in LPDEST overrides the value in PRINTER. This variable is optional. You can also use Infoprint Server ISPF panels to define a default printer. The printer named in either LPDEST or PRINTER overrides the printer named on the ISPF configuration panel.

Note: You do not need to start any daemons in addition to the Printer Inventory Manager daemon to use the z/OS UNIX printing commands.

4.3 Transforming files to AFP format
Version 2 Release 1 (V2.1) of IBM Infoprint Transforms to AFP for z/OS (program number 5655-N60) provides data-stream transforms that let you transform documents to Advanced Function Presentation (AFP) format from other data stream formats. These transforms let you print non-AFP data on IBM AFP printers, also known as Intelligent Printer Data Stream (IPDS) printers. You can transform documents to AFP format from:

- HP Printer Control Language (PCL)
- Adobe Portable Document Format (PDF)
- Adobe PostScript
- SAP R/3 System Generic Output Format (SAPGOF)
You might want to transform documents without printing them if you intend to print a document many times. In this case, it is more efficient to transform the document once and print the output than to transform the document every time you print it.

z/OS UNIX transform commands - z/OS UNIX System Services users can use transform commands to convert documents to AFP format from another format without printing the documents. You can run the transform commands from the z/OS UNIX command line.

There are four ways to transform files to AFP format:

1. Printer definitions implemented by an administrator can set up the printer definitions in the Infoprint Server Printer Inventory so that Infoprint Server automatically transforms files to the appropriate format. For example, suppose that your administrator has configured the printer1 printer definition to use the PCL to AFP or PostScript to AFP transform.

   To submit a PCL file, test.pcl, to the printer named IPDSBOB and transform it automatically, enter from the OMVS command line:

   \texttt{lpr -d IPDSBOB test.pcl}

   In the Printer Inventory, the definition shown in Figure 4-5 exists for printer IPDSBOB. The filter shown to transform the PCL data to AFP is:

   \texttt{pcl2afp.dll -c letter 300 %filter-options}

2. Using the following transform commands:

   \texttt{pcl2afp} Transform PCL data to AFP data
   \texttt{pdf2afp} Transform PDF data to AFP data
   \texttt{ps2afp} Transform PostScript data to AFP data
   \texttt{sap2afp} Transform SAP OTF or ABAP data to AFP data

3. Using the AOPBATCH program.

4. Using the Print Interface subsystem.
### 4.3.1 pcl2afp - Transform PCL data to AFP data

The `pcl2afp` command converts a Printer Control Language (PCL) data file into an Advanced Function Presentation (AFP) data stream file.

If you specify the same option multiple times, the command uses only the last option.

You can specify one or more input files to be transformed. If you do not specify an input file name, or if you specify a dash (-) as the file name, `pcl2afp` uses standard input. The output file name is also optional. If you do not specify one, the `pcl2afp` command writes the results to standard output.

```
pcl2afp [-a imagetype] [-c transformclass] [-o outputfile] [-p pagerange] [-t outputtype] [inputfile ...]
```

**Note:** Descriptions of the keywords used with this transform command are shown in Table F-1 on page 258.

**pcl2afp command examples**

To transform the PCL file `myfile.pcl` into an AFP data stream, using the `a4_300` transform class, and write the result to a file called `myfile.afp`, enter:

```
pcl2afp -c a4_300 -o myfile.afp myfile.pcl
```

To transform the PCL file `myfile.pcl` into an AFP data stream as an IO1-MMR image, and send the result to the default printer with the `lp` command, enter:

```
pcl2afp -a io1-mmr myfile.pcl | lp
```

Figure 4-6 shows how to set the AOPCONF environment variable and invoke the `pcl2afp` command from a z/OS UNIX shell.

```
SYS1USR @ SC65:/>export AOPCONF=/u/syusr/aop.conf
SYS1USR @ SC65:/>echo $AOPCONF

SYS1USR @ SC65:/>pcl2afp -o "//tst.afp" "//tst.pcl"
SYS1USR @ SC65:/>
SYS1USR @ SC65:/>
```

**Figure 4-6** Set AOPCONF environment variable for pcl2afp command in z/OS shell

### 4.3.2 pdf2afp - Transform PDF data to AFP data

The `pdf2afp` command converts a Portable Document Format (PDF) data file into an Advanced Function Presentation (AFP) data stream file.

If you specify the same option multiple times, the command uses only the last option.

You can specify one or more input files to be transformed. If you do not specify an input file name, or if you specify a dash (-) as the file name, `pdf2afp` uses standard input. The output file name is also optional. If you do not specify one, the `pdf2afp` command writes the results to standard output.

```
```

**Note:** Descriptions of the keywords used with this transform command are shown in Table F-1 on page 258.
You can use the filter-options job attribute with, for example, the `lp` command to pass any of these options except `-o outputfile` to the transform.

**pdf2afp command examples**
To transform the PDF `myfile1.pdf` file into an AFP data stream as an IO1-MMR image, and send the result to the default printer with the `lp` command, enter:

```
pdf2afp -a io1-mmr myfile1.pdf | lp
```

To transform the PDF file `myfile1.pdf` into an AFP data stream, and then submit it to the 4019 printer called POKE, enter:

```
pdf2afp -r 300 myfile1.pdf | lp -d poke
```

You need to specify a resolution of 300 pels (`-r 300`) because the 4019 is a 300-pel resolution printer. The default resolution for the `pdf2afp` command is 600 pels.

To transform file `input.pdf` into an existing, cataloged MVS output data set called `hlq.output.afp(member)` enter:

```
pdf2afp -o "//'HLQ.OUTPUT.afp(member)" input.pdf
```

### 4.3.3 ps2afp - Transform PostScript data to AFP data

The `ps2afp` command converts a PostScript data file into an Advanced Function Presentation (AFP) data stream file. If you specify the same option multiple times, the command uses only the last option, with the exception of the `-i` option. You can specify the `-i` option multiple times.

You can specify one or more input files to be transformed. If you do not specify an input file name, or if you specify a dash (-) as the file name, `ps2afp` uses standard input. The output file name is also optional. If you do not specify one, the `ps2afp` command writes the results to standard output.

```
```

**Note:** The descriptions of the keywords used with this transform command are shown in Table F-1 on page 258.

You can use the filter-options job attribute with, for example, the `lp` command to pass any of these options except `-o outputfile` to the transform.

**ps2afp command examples**
To transform the PostScript file `myfile.ps` into an AFP data stream, using the bigjob transform class, and write the result to a file called `myfile.afp`, enter:

```
ps2afp -c bigjob -o myfile.afp myfile.ps
```

To transform the PostScript file `myfile2.ps` into an AFP data stream, with an image that is 8 inches high and 5 inches wide, and write the result to a file called `myfile2.afp`, enter:

```
ps2afp -l 8in -w 5in -o myfile2.afp myfile2.ps
```
To transform the PostScript files input.01.ps, input.02.ps, input.xx.ps into one AFP output file called output.afp, enter:

```
ps2afp -o output.afp input.01.ps input.02.ps input.xx.ps
```

To transform file input.ps into an existing, cataloged MVS output data set called HLQ.OUTPUT.AFP(MEMBER) enter:

```
ps2afp -o "//'hlq.output.afp(member)"" input.ps
```

### 4.3.4 sap2afp - Transform SAP OTF or ABAP data to AFP data

The **sap2afp** command converts System Generic Output Format (SAPGOF) Output Text Format (OTF) and Advanced Business Application Programming (ABAP) data files:

- SAP OTF data files are converted into Advanced Function Presentation (AFP) Presentation Text Object Content Architecture (PTOCA) data streams.
- SAP ABAP data files are converted into line data streams.

If you specify the same option multiple times, the command uses only the last option, with the exception of the page range (-p) option. You can specify the -p option up to 20 times.

You can specify one or more input files to be transformed. If you specify more than one input file, the command concatenates the files, and the job attributes are determined by the first file transformed. If you do not specify an input file name, or if you specify a dash (-) as the file name, **sap2afp** uses standard input. The output file name is also optional. If you do not specify one, the **sap2afp** command writes the results to standard output.

The size of the transformed image and the position of SAP data on the page depend on the values that the administrator has defined.

```
sap2afp [-st] [-o outputfile] [-p pagerange] [-r resolution] [inputfile ...]
```

**Note:** Descriptions of the keywords used with this transform command are shown in Table F-1 on page 258.

**sap2afp command examples**

To transform the SAP ABAP file myfile.abap for printing on a 600-pel AFP printer, and write the result to a file called myfile.afp, enter:

```
sap2afp -r 600 -o myfile.afp myfile.abap
```

To transform the SAP OTF file myfile.otf into an AFP data stream, and send the result to the default printer with the **lp** command, enter:

```
sap2afp myfile.otf | lp
```

To transform file input.sap into an existing, cataloged MVS output data set called hlq.output.afp(MEMBER) enter:

```
sap2afp -o "//'hlq.output.afp(MEMBER)"" input.sap
```

To transform the MVS data set HLQ.INPUT.SAP(MEMBER) into an output file called output.afp, enter:

```
sap2afp -o output.afp "//'HLQ.INPUT.SAP(MEMBER)"
```
4.4 Copying xxx2afp transform output files to MVS data sets

If you have specified for the Infoprint Transforms to AFP the output to be written into the hierarchical file systems (HFS) and you want to copy the data into an MVS data set, during such a copy, the structured fields of the AFP data are treated as a stream of data and the original records are lost, rendering the data set non-printable.

Important: The AFPFIX REXX program, shown in Appendix A.3, “AFPFIX REXX exec” on page 190, examines the input file for AFP structured field records that may have been reformatted during a transform to HFS or copy from the HFS to an MVS data set. It can be used to create a printable MVS AFP data set from the already transformed AFP file.

4.5 Infoprint Transforms from AFP for z/OS

z/OS UNIX transform commands z/OS UNIX System Services users can use transform commands to convert documents from AFP format to another format without printing the documents. You can run the transform commands from the z/OS UNIX command line. These transforms can be used to print a document many times. In this case, it is more efficient to transform the document once and print the output than to transform the document every time you print it. Also, another consideration is to present a document on the Web as a PDF document by transforming the AFP document to a PDF.

4.5.1 afp2pcl - Transform AFP data to PCL data

The afp2pcl command converts an Advanced Function Presentation (AFP) data file into a Printer Control Language (PCL) data stream file. Error messages related to errors in the input data stream are written at the end of the output document.

```
afp2pcl [-c transformclass] [-F tracefile] [-i inputcodepage]
         [-j jobattributes] [-o outputfile] [-T traceoptions] [inputfile ....]
```

Note: Descriptions of the keywords used with this transform command are shown in Table F-2 on page 261.

Using afp2pcl transforms

If you specify multiple values of the same option, except for -j, the transform uses the last value that you specified.

When transforming line data in UNIX files that contain ANSI or no carriage control characters, you must specify document-format=line. If the data has ANSI control characters, also specify carriage-control-type=ansi.

afp2pcl command examples

To transform the AFP file myfile.afp into a PCL file, using the us transform class, and write a file called myfile.pcl, enter:

```
afp2pcl -c us -o myfile.pcl myfile.afp
```

To transform the MVS data set USERX.AFP(MYFILE) into a PCL file, using the form definition F1CP0110, and write a file called myfile.pcl, enter:

```
afp2pcl -j "form-def=f1cp0110 " -o myfile.pcl "//USERX.AFP(MYFILE)"
```
To transform the AFP file myfile.afp into a PCL file, using the form definition F1CP0110 that contains references to user-supplied AFP resources, and write a file called myfile.pcl, enter this command on one line:

```bash
afp2pcl -j "form-def=f1cp0110 res-lib={lib1.pseglib lib3.private}"
-o myfile.pcl myfile.afp
```

To transform the AFP file input.afp into the PCL output file called output.pcl, enter:

```bash
afp2pcl < input.afp > output.pcl
```

**Note:** You can use redirection operators only with UNIX files.

To transform line data in file myfile.line that contains ANSI carriage control characters into a PCL file, using the form definition F1CP0110 and page definition P1P06362, and write a file called myfile.pcl, enter this command on one line:

```bash
afp2pcl -j "form-def=f1cp0110 page-def=p1p06362 c-c-t=a doc-format=line"
-o myfile.pcl myfile.line
```

Transform line data, specifying a form definition and fonts. To transform the line data file myfile.line containing machine carriage control characters and table reference characters into a PCL file, using the form definition F1CP0110, and write a file called myfile.pcl, enter this command on one line:

```bash
afp2pcl -j "form-def=f1cp0110 c-c-t=m t-r-c=yes chars={60D8 60D0}"
-o myfile.pcl myfile.line
```

### 4.5.2 afp2pdf - Transform AFP data to PDF data

The `afp2pdf` command converts an Advanced Function Presentation (AFP) data file into an Adobe Portable Document Format (PDF) data stream file for printing or e-mailing. Error messages related to errors in the input data stream are written at the end of the output document.

```bash
afp2pdf [-c transformclass] [-F tracefile] [-i inputcodepage]
[-j jobattributes [-o outputfile] [-T traceoptions] [inputfile ...]
```

**Note:** Descriptions of the keywords used with this transform command are shown in Table F-2 on page 261.

**Usage notes**

If you specify multiple values of the same option, except for `-j`, the transform uses the last value that you specified.

When transforming line data in UNIX files that contain ANSI or no carriage control characters, you must specify `document-format=line`. If the data has ANSI control characters, also specify `carriage-control-type=ansi`.

**afp2pdf command examples**

To transform the AFP file myfile.afp into a PDF file, using the us transform class, and write a file called myfile.pdf, enter:

```bash
afp2pdf -c us -o myfile.pdf myfile.afp
```

Note: You can use redirection operators only with UNIX files.

Note: Descriptions of the keywords used with this transform command are shown in Table F-2 on page 261.

**Usage notes**

If you specify multiple values of the same option, except for `-j`, the transform uses the last value that you specified.

When transforming line data in UNIX files that contain ANSI or no carriage control characters, you must specify `document-format=line`. If the data has ANSI control characters, also specify `carriage-control-type=ansi`.

**afp2pdf command examples**

To transform the AFP file myfile.afp into a PDF file, using the us transform class, and write a file called myfile.pdf, enter:

```bash
afp2pdf -c us -o myfile.pdf myfile.afp
```
To transform the MVS data set USERX.AFP(MYFILE) into a PDF file, using the form definition F1C10110, and write a file called myfile.pdf, enter:

```
afp2pdf -j "form-def=f1c10110" -o myfile.pdf "//'USERX.AFP(MYFILE)"
```

To transform the AFP file myfile.afp into a PDF file, using the form definition F1C10110 that contains references to user-supplied AFP resources, and write a file called myfile.pdf, enter this command on one line:

```
afp2pdf -j "form-def=f1c10110 res-lib={lib1.pseglib lib3.private}" -o myfile.pdf myfile.afp
```

To transform the line data file myfile.line containing machine carriage control characters into a PDF file, using the page definition P1P06362, positioning the output 24 millimeters (approximately one inch) from the left edge of the paper, and write a file called myfile.pdf, enter this command on one line:

```
afp2pdf -j "page-def=p1p06362 c-c-t=m x-image-shift-front=24" -o myfile.pdf myfile.line
```

To transform the AFP file myfile.afp into a PDF file, encrypting the PDF document and specifying that the user cannot print or change the encrypted PDF document, and write a file called myfile.pdf, enter this command on one line:

```
afp2pdf -j "pdf-owner-identifier='Nurse-Lee' pdf-user-identifier='Dr-Smith' pdf-protect={print modify}" -o myfile.pdf myfile.afp
```

### Job attributes for encrypting PDF documents

You can use the AFP to PDF transform to encrypt PDF documents by specifying proper job attributes. Encrypting a PDF document protects it from unauthorized access.

**Note:** For a description of other Infoprint Server job attributes, see *z/OS Infoprint Server User's Guide*, S544-5746.

**pdf-encryption-level**

This single-valued attribute specifies the level of encryption used to encrypt PDF documents. A high level of encryption provides enhanced security. However, some users might not be able to open PDF documents that use a high level of encryption.

For the allowed values, you can enter one of these fixed values:

- **bits40** A low level of encryption (a 40-bit encryption key) is used. Select this value if you e-mail PDF documents to countries that do not use 128-bit encryption, or for Adobe Acrobat Reader 3.0 - 4.x.
- **bits128** A high level of encryption (a 128-bit encryption key) is used. Select this value for sensitive PDF documents.

**Default value:** bits128

**pdf-owner-identifier**

This single-valued attribute specifies the identifier of the owner of an encrypted PDF document. The owner identifier is associated with a password that is stored in a separate database. An owner password is required to restrict actions with the pdf-protect attribute.

**Allowed values** You can enter a text string of 1 to 256 characters. You can enter any combination of letters, numbers, blanks, and special characters that the Password exit allows. Your administrator sets up the Password exit. This text string might be case-sensitive, depending on the
Password exit. If the text string you specify contains blanks or special characters (such as @ $ & ( ) > < ' " #), enclose the text string in single or double quotation marks. For example:

- j "pdf-owner-identifier='Nurse-Lee@hospital.com'"

If the string contains double quotation marks, enclose the string in single quotation marks.

**Default value**
The owner identifier that the administrator has specified in the printer definition. If none is specified, there is no default.

**Usage guidelines**
The transform encrypts a PDF document when a user identifier, an owner identifier, or both are specified. Encrypting a PDF document protects it from unauthorized access. In most cases, the user and owner identifiers should be different because the passwords must be different.

### pdf-protect
This multi-valued attribute specifies one or more actions that users cannot do on encrypted PDF documents.

**Allowed values**
You can enter one or more fixed values to restrict actions. If you specify more than one value, separate the values with spaces and enclose the list of values in braces {}. For example:

- j "pdf-protect=all"
- j "pdf-protect={copy update}"

**Value:** Actions users cannot do:

- **all**
  - All actions (copy, print, update)
- **copy**
  - Copy or extract content to another document
  - Extract content for accessibility
- **print**
  - Print at low resolution (150 dpi)
  - Print at high resolution
- **update**
  - Change the document
  - Assemble the document (insert, delete, rotate pages)
  - Add comments
  - Fill in form fields or sign
  - Create template pages

**Default value**
The value that the administrator has specified in the printer definition. If none is specified, no actions are restricted.

### pdf-user-identifier
This single-valued attribute specifies the identifier of the user of an encrypted PDF document. The user identifier is associated with a password that is stored in a separate database. The user enters the user password when opening the encrypted PDF document.

**Allowed values**
You can enter a text string of 1 to 256 characters. You can enter any combination of letters, numbers, blanks, and special characters that the Password exit allows. Your administrator sets up the Password exit. This text string might be case-sensitive, depending on the...
Password exit. If the text string you specify contains blanks or special characters (such as @ $ & ( ) > < ' " #), enclose the text string in single or double quotation marks. For example:

- \texttt{j "pdf-user-identifier='Dr-Smith@hospital.com'"}

If the string contains double quotation marks, enclose the string in single quotation marks.

**Default value**

The user identifier that the administrator has specified in the printer definition. If none is specified, any user can open the PDF document without a password.

**Usage guidelines**

The transform encrypts a PDF document when a user identifier, an owner identifier, or both is specified. Encrypting a PDF document protects it from unauthorized access.

In most cases, the user and owner identifiers should be different because the passwords must be different.

**afp2pdf transform example**

To transform the MVS data set U.CLIST(AA) into an encrypted PDF data file, using the transform class encrypt, enter:

\texttt{afp2pdf -c encrypt -o /tmp/encr.pdf -j "p-p=all p-o-i=IBM p-u-i=ITSO" "//u.clist(aa)"

The afp2pdf_encrypt transform class is shown in “Examples of AFP to PDF transforms” on page 95. The /etc/Printsrv/aoppdfexit.db file, used by the sample password exit /usr/lpp/Printsrv/lib/aoppdfexit.dll, contains identifiers and passwords for the IBM and ITSO users.

### 4.5.3 afp2ps - Transform AFP data to PostScript data

The **afp2ps** command converts an Advanced Function Presentation (AFP) data file into a PostScript data stream file.

Error messages related to errors in the input data stream are written at the end of the output document.

\texttt{afp2ps [-c transformclass] [-F tracefile] [-i inputcodepage] [-j jobattributes] [-o outputfile] [-T traceoptions] [inputfile ]}

**Note:** Descriptions of the keywords used with this transform command are shown in Table F-2 on page 261.

You can use the **filter-options** job attribute with, for example, the **lp** command to pass the \texttt{-c transformclass} and \texttt{-i inputcodepage} options to the transform.

**Usage notes**

If you specify multiple values of the same option, except for \texttt{-j}, the transform uses the last value that you specified.

When transforming line data in UNIX files that contain ANSI or no carriage control characters, you must specify \texttt{document-format=line}. If the data has ANSI control characters, also specify \texttt{carriage-control-type=ansi}.
afp2ps transform examples

To transform the AFP file myfile.afp into a PostScript file, using the us transform class, and write a file called myfile.ps, enter:

```
afp2ps -c us -o myfile.ps myfile.afp
```

To transform the MVS data set USERX.AFPO(MYFILE) into a PostScript file, using the form definition F1CP0110, and write a file called myfile.ps, enter:

```
afp2ps -j "form-def=f1cp0110" -o myfile.ps "//USERX.AFPO(MYFILE)"
```

To transform the AFP file myfile.afp into a PostScript file, using the form definition F1CP0110 that contains references to user-supplied AFP resources, and write a file called myfile.ps, enter this command on one line:

```
afp2ps -j "form-def=f1cp0110 res-lib={lib1.pseglib lib3.private}" -o myfile.ps myfile.afp
```

To transform the MVS data set PROD.AFPOUT(JOB1) into a PostScript file, using the form definition F1CP0110 that contains references to user-supplied AFP resources, and print the output, enter this command on one line:

```
afp2ps -j "form-def=f1cp0110 res-lib={lib1.pseglib lib3.private}" "//PROD.AFPOUT(JOB1)" | lp
```

To transform the AFP file input.afp into the PostScript output file called output.ps, enter:

```
afp2ps < input.afp > output.ps
```

Note that you can use redirection operators only with UNIX files.

To transform the line data file myfile.line containing machine carriage control characters into a PostScript file, using the page definition P1P06362, positioning the output approximately 1 inch from the left edge of the paper, and write a file called myfile.ps, enter this command on one line:

```
afp2ps -j "page-def=p1p06362 c-c-t=m x-image-shift-front=24" -o myfile.ps myfile.line
```

4.6 Transforms for XML data streams

Infoprint XML Extender for z/OS, 5655-J66 lets you transform Extensible Markup Language (XML) files to AFP or PDF format for printing or e-mailing. To transform XML files to another format, you can transform XML to AFP, and then transform AFP to PCL or PostScript.

The transforms can transform XML documents encoded in EBCDIC (single-byte only), ASCII (single-byte only), UTF-8, or UTF-16. A page definition is required to provide the data placement and presentation information.

To transform and print XML documents, the administrator must associate the transform with both the line data and XML data formats in the printer definition.

4.6.1 xml2afp - Transform XML data to AFP data

The xml2afp command converts an Extensible Markup Language (XML) data file into an Advanced Function Presentation (AFP) data file.

Error messages related to errors in the input data stream are written to stderr.
xml2afp [-T] [-c transformclass] [-d destinationtype] [-l pagelength[in | mm]]
[-o outputfile] [-s { stylesheet | none}] [-w pagewidth[in | mm]]
[inputfile...]

**Note:** Descriptions of the keywords used with this transform command are shown in Table F-3 on page 263.

**Note:** If you specify multiple values for the same option, xml2afp uses the last value. If the transform is managed by Infoprint Server, default options are specified in the aopxfd.conf transform configuration file.

**Messages**
Messages are written to stderr.

**xml2afp transform examples**
To transform, in the OMVS shell, the XML file ivp.xml into a AFP file, using the default transform class, and write a file called t.afp, enter:

```shell
xml2afp -o t.afp ivp.xml
```

The t.afp file is empty and the stderr file displays the following messages:

```
---------------------------
org.xml.sax.SAXException: Text before the root.
org.xml.sax.SAXException: Text before the root.
  at com.ibm.xsl.composer.framework.FOHandler.characters(FOHandler.java:142)
  at com.ibm.xsl.composer.framework.XFCHandler.characters(XFCHandler.java:113)
  at org.apache.xerces.parsers.AbstractSAXParser.characters(Unknown Source)
  at org.apache.xerces.impl.XMLDocumentFragmentScannerImpl.scanContent(Unknown Source)
  at org.apache.xerces.impl.XMLDocumentFragmentScannerImpl$FragmentContentDispatcher.dispatch(Unknown Source)
  at org.apache.xerces.impl.XMLDocumentFragmentScannerImpl.scanDocument(Unknown Source)
  at org.apache.xerces.parsers.XML11Configuration.parse(Unknown Source)
  at org.apache.xerces.parsers.XML11Configuration.parse(Unknown Source)
  at org.apache.xerces.parsers.XMLParser.parse(Unknown Source)
  at org.apache.xerces.parsers.XMLParser.parse(Unknown Source)
  at org.apache.xerces.parsers.AbstractSAXParser.parse(Unknown Source)
  at javax.xml.parsers.SAXParser.parse(Unknown Source)
  at com.ibm.xml2afp.AFPComposer.compose(AFPComposer.java:340)
  at com.ibm.xml2afp.AFPComposer.runManaged(AFPComposer.java:228)
  at com.ibm.xml2afp.AFPComposer.run(AFPComposer.java:120)
  at com.ibm.xml2afp.AFPComposer.main(AFPComposer.java:87)
```

A style sheet specification is added to the transform request:

```shell
xml2afp -o t.afp -s ivp.xsl ivp.xml
```

No error messages are issued and the t.afp file contains a printable AFP data.
4.6.2 xml2pdf - Transform XML data to PDF data

The xml2pdf command converts an Extensible Markup Language (XML) data file into an Portable Document Format (PDF) data file.

Error messages related to errors in the input data stream are written to stderr.

```
xml2pdf [-T] [-c transformclass] [-o outputfile] [-s { stylesheet | none}] [inputfile...]
```

**Note:** Descriptions of the keywords used with this transform command are shown in Table F-3 on page 263.

**Note:** If you specify multiple values for the same option, xml2pdf uses the last value. If the transform is managed by Infoprint Server, default options are specified in the aopxfd.conf transform configuration file.

**Messages**

Messages are written to stderr.

4.7 The AOPBATCH program and the transforms

The AOPBATCH program lets you submit a batch job to transform data to and from the Advanced Function Presentation data format using the IBM Infoprint transform products. Infoprint Server provides the AOPBATCH program in SYS1.LINKLIB.

**Note:** The AOPBATCH program is assigned an authorized program facility (APF) authorization code 1 (one) that allows the use of restricted system services and resources. SYS1.LINKLIB in the LNKLST concatenation is APF-authorized by default. APF authorization for LNKLST data sets in the LNKLST concatenation is APF-authorized by default, if you accept the default for the LNKAUTH system parameter (LNKAUTH=LNKLST) or specify this value through an IEASYSxx member.

**Note:** IBM recommends that you use AOPBATCH instead of BPXBATCH to run programs that Infoprint Server provides because AOPBATCH sets default values for the PATH, LIBPATH, and NLSPATH environment variables that are suitable for installations that installed Infoprint Server files in default locations. Also, AOPBATCH lets stdin be read from a DD statement and lets stdout and stderr be written to a DD statement.

4.7.1 EXEC JCL statement PARM parameters

The AOPBATCH program accepts parameters that specify the transform command, followed by transform options and arguments:

```
//stepname EXEC PGM=AOPBATCH,PARM="/transform_name transform_options"
```

The slash in the beginning of the PARM data indicates that the PARM data that follows is input to AOPBATCH.
PARM data

If you omit the initial slash, your PARM data might be interpreted as C++ run-time options. You must include the initial slash if any of the PARM data itself includes a slash. For example, if the transform name is /mylib/ps2afp, specify:

```
PARM='/transform_name transform_options'
PARM='//mylib/ps2afp ...
```

Where:

- **transform_name**
  - The name of an executable transform program that resides in an HFS file. The name of the transform program is case-sensitive. You can specify one of these command names for to AFP transforms: pcl2afp, pdf2afp, ps2afp, sap2afp.
  - If the transform program does not reside in one of the directories specified in the PATH environment variable, also specify the pathname. You can use the STDENV DD statement to set the PATH environment variable if the default value set by AOPBATCH is not suitable. For information about the defaults set for environment variables, see 4.7.2, “AOPBATCH DD statements” on page 159.

- **transform_options**
  - Options and arguments accepted by the transform. For a description of each, see 4.2, “Printing using Infoprint transform commands” on page 140.
    - You must specify the transform input data set or file as a transform argument, and you must specify the -o transform option to identify where you want the transform to write its output. (This is because the transform cannot write its output to standard output and cannot read input from standard input.) To identify the transform input and output data set or file, you can specify either a DD statement name or a data set or file name. You must specify a DD statement name if you want to write the transform output to an MVS data set that does not already exist.
    - Specify the names of DD statements to the transform in this format:
      ```
      //DD:DDname
      ```
    - Where the name of the DD statement is DDname. When you specify an MVS data set name in the -o option, code two slashes before the data set name and enclose the data set name in two sets of single quotation marks if you specify a fully qualified data set name. If you do not enclose the data set name in quotation marks, a high-level qualifier is added to the name you specify:
      - If you are running under TSO (batch or interactive), the TSO user prefix is appended.
      - If you are running under MVS batch or IMS (batch or online), the RACF user ID is appended.
      - If your system does not use RACF, a high-level qualifier is not added.
For examples of different ways to specify transform input and output data sets and files, see 4.7.3, “AOPBATCH examples” on page 160.

### 4.7.2 AOPBATCH DD statements

The AOPBATCH JCL procedure accepts the standard DD statements shown in Figure 4-7.

```plaintext
//AOPBATCH JOB ...
//TRANSFRM EXEC PGM=AOPBATCH,
// PARM='/afp2pdf -o /tmp/output.pdf "//HLQ.INPUT.AFP"'
//STDOUT   DD   SYSOUT=*  
//STDERR   DD   SYSOUT=* 
//STDENV   DD   *
    PATH=/usr/lpp/Printsrv/bin:/bin:/usr/bin
    LIBPATH=/usr/lpp/Printsrv/lib:/lib:/usr/lib
    NLSPATH=/usr/lpp/Printsrv/En_US/%N:/usr/lib/nls/msg/En_US/%N
    AOPCONF=/etc/Printsrv/aopd.conf
/*
```

Figure 4-7   Sample AOPBATCH JCL

Where:

**STDENV**  Specifies environment variables for use by the transform. You can specify the environment variables in-stream in the JCL, in an MVS data set, or in a UNIX file. Specify the environment variables in the format variable=value, with one environment variable per line or record. Sequence numbers in columns 73 to 80 in data specified with the STDENV DD statement are ignored and not treated as part of the data.

If you omit the STDENV DD statement or do not specify one of the environment variables, AOPBATCH sets these default values, which are suitable for running Infoprint Server programs if your installation installed Infoprint Server files in the default directories, as follows:

- `PATH=/usr/lpp/Printsrv/bin:/bin:/usr/bin`
- `LIBPATH=/usr/lpp/Printsrv/lib:/lib:/usr/lib`
- `NLSPATH=/usr/lpp/Printsrv/En_US/%N:/usr/lib/nls/msg/En_US/%N`
- `AOPCONF=/etc/Printsrv/aopd.conf`

**Note:** Do not specify the `_BPX_SHAREAS` environment variable. AOPBATCH will set it appropriately.

**STDERR**  Specifies the system output data set where error messages are to be written. The data set can be an MVS data set or a UNIX file. The transforms do not write messages related to errors in the input data stream in this data set. Instead, the transforms write these messages at the end of the output document.

**STDOUT**  Specifies the system output data set where informational messages are to be written. The data set can be an MVS data set or a UNIX file. You can also include DD statements to specify MVS data sets that contain input data to be transformed, the transformed output, or job attributes that are input to the transform. Do not use DD names STDIN, STDOUT, or STDERR to specify the
transform input and output data sets. Instead, use other DD names, such as INPUT and OUTPUT, which are used in the examples.

Important:

- If the output data set is an MVS data set, these requirements apply:
  - You must either allocate and catalog the data set before you run AOPBATCH, or include a DD statement in the AOPBATCH job to allocate the data set.
  - The MVS output data set must be large enough to hold the output data stream. The size of the output data stream depends on the complexity of the document. If you run the PCL to AFP, PDF to AFP, or PostScript to AFP transform, the type of image compression you select in the -a option also affects the size of the output data stream. Typically, an output AFP data stream is several times as large as the input data stream.
  - Allocate a data set with the format and characteristics:
    - Record format: VBM
    - Record length: 8192 (8K) or larger
  - Specify a disposition of SHR or OLD if you want the transform to overwrite any existing data. Otherwise, specify a disposition of MOD to append the output to any existing data. If you do not specify any disposition, the transform overwrites any existing data.
- If you have not added the Language Environment run-time library (CEE.SCEERUN) or the C++ run-time library (CBC.SCLBDLL) to the system LNKLST, specify these data sets in a STEPLIB DD statement.
- You can concatenate input data sets that have the same data format; for example, PostScript data or AFP data. However, you cannot concatenate data sets that contain PDF data.

4.7.3 AOPBATCH examples

This section provides examples of using the z/OS UNIX transform commands.

Example 1

This example shows how to transform data when the transform input is in an MVS data set and transform output is written to an MVS data set. The PostScript to AFP transform reads PostScript input from data set USER.INPUT.PS and writes AFP output to data set USER.OUTPUT.AFP. USER represents the high-level qualifier; for example, your TSO or RACF user ID.

This example also shows how to specify environment variables in-stream in the STDENV DD statement. If you installed Infoprint Server and created Infoprint Server configuration files in default directories, you do not need to specify the environment variables and you can also omit the STDENV DD statement.
Example 2
The second example shows how to transform data when the transform input is in a UNIX file and transform output is written to an MVS data set. The PDF to AFP transform reads the PDF input from file /tmp/input.pdf and writes AFP output to USER.OUTPUT.AFP.

Example 3
The third example shows how to transform data and print the output from the transform. In the first and only step, the AFP to PCL transform reads EBCDIC line data input from the data set specified with the INPUT DD statement and writes PCL output to JES spool SYSOUT class Y destination SYSLBD45.

Here is an extract of the SYSLBD45 printer’s pidu display output:

```bash
# display printer SYSLBD45
printer-codepage = ISO8859-1
destination = SYSLBD45
output-class = Y
printer-ip-address = syslab1145.osti.mbi.com
document-formats-supported = { line modca postscript text pcl other }
```

Since the spooled AFP data set’s attributes match the SYSLBD45 printer’s destination = SYSLBD45 and output-class = Y, the spooled data set will be printed on that printer. The extended mode IP Printway Job Selection Rule specifies as the only selection criteria CLASS=Y.
Example 4

This example shows the batch job to support the Infoprint XML Extender installation verification process. It transforms XML data and prints the output from the transform.

In the first step (X2AIVP), the XML to PDF transform reads the IVP input from the file specified with the XMLIN DD statement and writes AFP output (DD:AFPOUT) to the JES spool sysout class Y destination SYSLBD45. The filter definition for the MO:DCA-P (AFP) data format on the SYSLBD45 printer definition transforms the AFP data to PCL and sends it to the IBM Infoprint 1145 printer.
In the second step (X2PIVP), the XML to AFP transform reads the IVP input from the file specified with the XMLIN DD statement and writes PDF output (DD:PDFOUT) to JES spool sysout class Y destination PDF2AFP. (The PDF2AFP printer writes the print data back to spool sysout class Y destination SYSLBD45.)

Steps X2AIVP and X2PIVP invoke inline procedures XML2AFP and XML2PDF respectively.
pidu display
Following is a pidu display of the PDF2AFP printer definition in the Printer Inventory. The printer uses the LPR protocol type and the loopback IP address (127.0.0.1) is used for the “remote” printer.

```plaintext
# display printer PDF2AFP
filters = { pdf -> "ps2afp.dll %filter-options" }
destination = PDF2AFP
output-class = F
print-queue-name = AFP2PCL
printer-ip-address = 127.0.0.1
document-formats-supported = { pdf }
printer-type = ip-printway
forms = STD
duplexes-supported = { }
printway-sosi-mode = none
dcf-routing = yes
lpr-print-banner = no
printer-codepage = IBM-1047
print-error-reporting-supported = { none }
print-page-header = no
translate-document-trailer = no
translate-document-header = no
```

AOPBATCH exit values
AOPBATCH returns the exit code of the spawned process. If AOPBATCH cannot execute the program, it returns RC=4. If a transform command fails, it returns RC=1.

4.7.4 Using AOPBATCH in REXX programs in TSO/E environment

TSO/E provides the LINKMVS host command environment that lets you link to unauthorized programs. Note that the AOPBATCH program does not use restricted system services for all programs that reside in z/OS UNIX files. All programs that reside in z/OS UNIX files do not use restricted system services.

**Note:** If the AOPBATCH program fails when trying to use restricted system services, you could use the z/OS UNIX bpxwunix() function. The bpxwunix() function runs a shell command and optionally provides its stdin, traps its stdout output, traps its stderr data, and exports a set of environment variables.

When TSO/E REXX is used for z/OS UNIX processing, all of the z/OS UNIX functions, except bpxwunix() and syscalls(), must be run in a z/OS UNIX environment.

AOPBATCH REXX example
The following REXX exec prints a file or data set from a TSO/E session.

```rexx
/* rexx - Sample print a file or data set TSOPR EXEC for TSO/E sessions */
/* Syntax: TSOPR printer_name input_data options */
/* Values entered for options must follow lp command syntax rules*/
parse arg printer input options
parm = "Alloc STDOUT"
"alloc dd(STDOUT) ds(*) reu" /* STDOUT for AOPBATCH*/
if rc = 0 then do
parm = "Alloc STDERR"
```

164 Infoprint Server for z/OS Infoprint Transforms
"alloc dd(STDERR) ds(*) reu" /* STDERR for AOPBATCH */
if rc = 0 then do
  /* STDENV may point to a dataset containing environment variables. */
  if rc = 0 then do
    parm = "/lp -d "printer" "options" "input /* Command to execute */
    address "LINKMVS" "AOPBATCH parm" /* Invoke command processing */
  end
end
end
if rc <> 0 then say parm "rc" rc "*** errors ***" return

bpxwunix() example
The bpxwunix() function runs a shell command and optionally:

- Provides its stdin
- Traps its stdout
- Traps its stderr
- Exports a set of environment variables

Note: The bpxwunix() function runs the shell by passing a single command similar to the `sh -c` command. It does not run a login shell. bpxwunix() can be used outside of the z/OS UNIX REXX environment (for example, in TSO/E). In this case, stdin, stdout, stderr, and environment variables are not inherited from the current process environment.

Infoprint Server provides, mainly for testing filters, the `filter` command. The syntax is as follows:

filter dll_name optional_arguments

The input data to the filter command is read from stdin; output is directed to stdout. For a description of the filter command, see /usr/lpp/Printsrv/samples/cfilter.h.

In the following example, the filter command is invoked in TSO/E foreground to execute a pipe which first transforms a PDF file to AFP, then the AFP to PCL, and finally prints the PCL file.

/* rexx - Sample TSOFILTR EXEC to transform and print a PDF as PCL */
/* using filter command. */
/* Syntax: TSOFILTR input_PDF_file printer_name */
/* Limitation: Does not support PDS in an MVSDataset */
parse arg file printer /* take file and printer as argument */
if file = "" then do; say '*** Error! Please specify a UNIX file'
return; end
if printer = "" then printer='IAZFSS' /* establish default printer */
ADDRESS TSO "alloc dd(OUT) sysout reu"
ADDRESS TSO "alloc dd(ERR) ds(*) reu"
if syscalls('ON')>3 then do
  say 'Unable to establish the SYSCALL environment'; return; end
/* setup for bpxwunix() function to invoke filter command */
uss_cmd="filter -i "file" ps2afp.dll | " /* pipe pdf -> afp */
"filter afp2pcl.dll | " /* afp -> pcl */
"lp -d "printer" /* pcl -> lp */
env.0=4 /* establish the environment */
env.1='NLSPATH=/usr/lpp/Printsrv/En_US/%N'
env.2='LIBPATH=/usr/lpp/Printsrv/lib'
env.3='PATH=/usr/lpp/Printsrv/bin'
env.4='_BPX_SHAREAS=SHARE'
ADDRESS TSO "unalloc dd(OUT ERR)"
return

4.8 Using the Print Interface subsystem

The Print Interface subsystem can transform and print output from z/OS applications with minimal changes to an application’s job control language (JCL). The subsystem processes the application’s output, transforms it to the format required by the printer or e-mail destination, and writes it to the JES spool. From the JES spool, the transformed data can be printed on any printer, including IBM AFP printers controlled by PSF and PostScript or PCL printers controlled by the IP PrintWay component of Infoprint Server. Also, the transformed data can be sent to e-mail destinations using the e-mail support that IP PrintWay provides.

Using Print Interface example

This example shows how to print a line data document created by a batch application on an IBM Infoprint 1145 printer. It assumes that:

1. The Print Interface subsystem named AOP1 has been started. AOP1 is the name of both the Printer Inventory and the Print Interface subsystem. The name must contain exactly four letters or numbers, and it is defined in the aopd.conf file.
2. Printer definition SYSLBD45 exists in the Infoprint Server Printer Inventory with the following characteristics:
   - The AFP to PCL transform is specified.
   - The class and destination name for the IP PrintWay printer are specified. (CLASS and DEST are JES work-selection parameters that direct output from the JES spool to an IP PrintWay-controlled printer.)

   ```
   //AFP2PCL  JOB #3,TRANSFORMER,MSGLEVEL=1,MSGCLASS=A,NOTIFY=&SYSUID
   //PRINT EXEC PGM=IEBGENER
   //SYSUT2 DD SUBSYS=(AOP1,'SYSLBD45')
   //SYSPRINT DD DUMMY
   //SYSIN DD DUMMY
   //SYSUT1 DD *
   Two lines line ..
   .. data format data;-
   ```

The output produced by this example is the same as in the third example described previously. The transform filter definitions for the SYSLBD45 printer are shown in Figure 4-12.
4.8.1 OUTPUT JCL statements and transforms from AFP

The OUTPUT JCL statements are used to specify processing options for system output (SYSOUT) data sets. These processing options are used only when the OUTPUT JCL statement is explicitly or implicitly referenced by a SYSOUT DD statement. JES combines the options from this OUTPUT JCL statement with the options from the referencing DD statement. All parameters are optional. For more information about JCL parameters, see z/OS MVS JCL Reference, SA22-7597.

This section describes the parameters on the DD and OUTPUT JCL statements that the transforms use. Many of these JCL parameters have corresponding job attributes. For example, the PAGEDEF parameter of the OUTPUT JCL statement is equivalent to the page-definition job attribute.

If you need to specify job attributes that do not have corresponding JCL parameters, you can submit the output data set to the Print Interface subsystem and specify job attributes on the SUBSYS parameter of the OUTPUT JCL statement.

Using the Print Interface subsystem

An example of using the Print Interface subsystem is to transform an AFP document to PDF format and mail it to an e-mail address. The Print Interface subsystem lets you specify job attributes that you cannot specify in JCL parameters, such as the job attributes used to encrypt PDF documents. The example shows how to specify:

- The SUBSYS JCL parameter, with job attributes, can be used to encrypt the PDF document (pdf-owner-identifier, pdf-user-identifier, pdf-protect).
- SUBSYS=(subsystem_name[,,'printer_definition_name'][,'attribute=value ...'])

- JCL parameters used to e-mail the PDF document are:
  MAILTO, MAILFROM, REPLYTO, MAILFILE, TITLE
The example also assumes that:

- Printer definition EMAIL10 exists in the Infoprint Server Printer Inventory, with these characteristics:
  - The E-mail protocol is selected.
  - The AFP to PDF transform is specified for the data format.
  - The name of the Print Interface subsystem is AOP1.
  - The environment variable AOP_ENCRYPT is set to Yes in the transform class.

AOP_ENCRYPT indicates whether you want the transform to encrypt documents. Before setting AOP_ENCRYPT to yes, you must customize the Open Cryptographic Services Facility (OCSF). If OCSF is not customized, transform requests fail.

```java
//VAINIM JOB #23,EMAIL,MSGLEVEL=1,NOTIFY=&SYSUID
//STEPS EXEC PGM=IEBGENER
//OUTDS1 OUTPUT FORMDEF=MYDEF,MAILTO='CEO@xyz.com',
//   MAILFROM='John Sender',REPLYTO='secretary@xyz.com',
//   MAILFILE='July report',TITLE='Monthly Report'
//SYSUT2 DD SUBSYS=(AOP1,'EMAIL10','pdf-owner-identifier=john,
//   pdf-user-identifier=ceo pdf-protect={copy update}'),
//   OUTPUT=*.*.OUTDS1
//SYSIN DD DUMMY
//SYSPRINT DD SYSOUT=* 
//SYSUT1 DD DISP=SHR,DSN=USERX.DATA(JULY)
```

![Figure 4-13 E-mail protocol panel for printer EMAIL10](image)

For more information about the subsystem, see z/OS Infoprint Server User’s Guide, S544-5746.

**DD and OUTPUT JCL parameters**

Figure 4-14 on page 169 summarizes the parameters on the DD and OUTPUT JCL statements that the transforms use.
Figure 4-14  Summary of OUTPUT JCL statement parameters used by transforms

Figure 4-15 on page 170 summarizes all job attributes Infoprint Server uses and the corresponding OUTPUT JCL statement parameters. The (DD) and (JOB) values indicate the JCL statement where the JCL parameter can be specified.
OUTPUT JCL parameters transform differences

In most cases, transforms interpret the parameters in the same way as PSF does, so that you can use the same JCL that you use when the output is printed on IBM AFP printers. Some parameters, however, have different characteristics when used with these transforms:

**CHARS**

If the page definition used to print the job is the system default page definition and the page definition specifies a font, the transforms do not use the font specified in the CHARS parameter. Instead, the transforms use the font in the page definition. PSF, on the other hand, uses the font specified in the CHARS parameter.

**DCB=RECFM**

When transforming XML data, if the RECFM subparameter indicates that the data set contains carriage control characters, the transforms do not transform the first character of each record. PSF, on the other hand, ignores the RECFM subparameter and always prints the first character of each XML record. Typically, XML data sets do not contain carriage control characters. Therefore, when printing XML data, do not specify a record format that indicates carriage controls.

**INTRAY**

The transforms use default input tray 1. PSF, on the other hand, uses the printer’s default source.
**OUTPUT or DD JCL parameters**

The parameters that you can specify on an OUTPUT or DD JCL statements are:

- **CHARS=(font_name1,[,font_name2][[,font_name3]][,[font_name4]])**
  Specifies the 4-character member name of the coded font that you want to use to print a data set that contains line data. You can specify up to four fonts. `font_name` specifies the name of a coded font (in a font library) containing 4 or fewer characters, not including the prefix.

- **DCB**
  - RECFM - Specifies the record format of the data set, including whether the data set contains carriage control characters.
  - OPTCD=J - Specifies whether the print data set contains table reference characters (TRCs). This is the same as TRC=YES.

- **DUPLEX={NO | NORMAL | TUMBLE}**
  - NO- The job is printed only on the front side of each sheet.
  - NORMAL - The job is printed on both sides of the sheet so that the top of side 1 is the top of side 2 (for side binding).
  - TUMBLE - The job is printed on both sides of the sheet so that the top of side 1 is the bottom of side 2 (for top binding).

- **FCB=fcb_name**
  Specifies the 1 to 4 character name of the page definition. The transforms add the prefix P1 to the FCB name you specify. Therefore, do not specify the prefix P1.
  You can also specify the name of the page definition in the PAGEDEF parameter. If you specify both the FCB and the PAGEDEF parameters, the transforms ignore the FCB parameter. For more information, see the PAGEDEF parameter.

- **FORMDEF=form_definition_name**
  Specifies the member name (from 1 to 6 alphanumeric or national characters) of the form definition you want to use. Omit the system prefix, F1, from the name. These transforms add F1 to the member name you specify.
  You can store the form definition that you use in any of these places:
  - In a system library assigned to these transforms
  - In a user library referred to in the printer definition
  - In a user library referred to in your JCL
  - Inline in the print data set

- **INTRAY=nnn**
  Specifies the 1 to 3 decimal digit number from 1 to 255 that identifies the tray from which paper is to be selected. These transforms map this tray number to the tray number of the PCL or PostScript printer, using tray-mapping values specified by the administrator in the transform configuration file.

- **OFFSETXB= nnnn[,mmm]unit**
  Specifies the X offset of the logical page origin to the right of the physical page origin on the back side of a double-sided sheet.

- **OFFSETXF=nnnn[,mmm]unit**
  Specifies the X offset of the logical page origin to the right of the physical page origin on the front of the sheet.
OFFSETYB=nnnn[,mmm]unit
Specifies the Y offset of the logical page origin below the physical page origin on the back side of a double-sided sheet.

OFFSETYF=nnnn[,mmm]unit
Specifies the offset in the Y direction of the logical page origin below the physical page origin on the front of the sheet.

OUTBIN=1-65 535
Specifies the 1 to 5 decimal digit identifier of the output bin into which Infoprint Server will place a print job. If the printer does not support the selection of an output bin, the job is stacked in the default output bin for the printer.

When you print on a PCL or PostScript printer, instead of on an AFP printer, specify the output bin number of the PCL or PostScript printer. The value you specify for this parameter overrides any output bin that the form definition specifies.

OVERLAYB=overlay_name
Specifies the member name (from 1 to 8 alphanumeric or national characters) of a medium overlay to be placed on the back side of each sheet in a two-sided job, in addition to overlays from other sources. Specify the complete name of the overlay member because these transforms do not add an O1 prefix.

PAGEDEF=page_definition_name
Specifies the member name (from 1 to 6 alphanumeric or national characters) of the page definition you want to use. When you specify the name in the JCL, omit the system prefix, P1. These transforms add it automatically.

If a PAGEDEF parameter is not coded in your JCL, these transforms use the page definition specified in the printer definition. If no form definition is specified in the printer definition, these transforms use the form definition in the Infoprint Server transform configuration file.

You can store the page definition that you use in any of these places:
- In a system library assigned to these transforms
- In a user library referred to in the printer definition
- In a user library referred to in your JCL
- Inline in the print data set

PRMODE={SOSI1 | SOSI2 | SOSI3}
Specifies the type of data in the print data set and whether these transforms must do optional processing of the data.
- SOSI1 - Specifies that each shift-out, shift-in code is converted to a blank and a Set Coded Font Local text control.
- SOSI2 - Specifies that each shift-out, shift-in code is converted to a Set Coded Font Local text control.
- SOSI3 - Specifies that the shift-in code is converted to a Set Coded Font Local text control and two blanks. A shift-out code is converted to a Set Coded Font Local text control.

JES uses values in the PRMODE parameter for job routing. The transforms ignore all values except SOSI1, SOSI2, and SOSI3, which they use to format data sets that contain both single-byte and double-byte fonts.
When you use the SOSI process, remember that:

- For the process to work correctly, you must specify two fonts in the CHARS parameter or in a page definition font list. The first font must be the single-byte font, and the second font must be the double-byte font.
- IBM recommends that you do not mix SOSI codes and TRCs in the same job.

▶ TRC={YES|NO}

Specifies whether the print data set contains table reference characters (TRCs). This is the same as the DCB=OPTCD=J parameter.

The table reference character is a numeric character that corresponds to the order in which you specified the character arrangement table names with the CHARS keyword. The system uses the table reference character for selection of a character arrangement table during printing.

Examples:

//OUTDS OUTPUT CHARS=(GT10,GT12),TRC=YES
//DD2 DD CHARS=(GT10,GT12),DCB=OPTCD=J

When you use table reference characters, remember that:

- If the TRC=YES and the page definition does not identify fonts, you must specify fonts with the CHARS parameter.
- The order in which the fonts are specified in the CHARS parameter establishes which number is assigned to each associated TRC. For example, the TRCs for the fonts in the preceding example are zero for font_name1 and one for font_name2.
- If you do not specify TRC=YES, but your line data contains a TRC as the first character of each line (or the second character if carriage control characters are used), the TRC is not used as a font identifier, but is printed as a text character.
- IBM recommends that you do not mix SOSI codes and TRCs in the same job.

▶ UCS=font_name

Serves as another way to select a font. When a CHARS parameter is not specified, you can specify the universal character set (UCS) parameter to select one font. If the page definition specifies a font, the UCS parameter is ignored.

▶ USERLIB=(library_name[,....])

Specifies the name of 1 to 8 cataloged MVS data sets (user libraries) containing AFP resources for processing the data set. The transforms dynamically allocate these data sets and search for resources in them in the order specified on the USERLIB statement. If the transforms find no resources, they search the system libraries defined in the Infoprint Server transform configuration file. The libraries you specify can contain any AFP resources: fonts, page segments, overlays, page definitions, form definitions, or object container resources.
Using OUTPUT JCL to transform AFP documents examples

The first example shows how to transform an AFP document to PCL format and print the transformed output on a PCL printer. It assumes that:

- Printer definition pcl1 exists in the Infoprint Server Printer Inventory with these characteristics:
  - The IP PrintWay LPR, direct-sockets, or IPP protocol is selected.
  - The AFP to PCL transform is specified.
- IP PrintWay prints output data sets in JES output class P.

```plaintext
//PCLJOB JOB #5,TRANSFORMER,MSGLEVEL=1,MSGCLASS=A,NOTIFY=&SYSUID
//PRINT EXEC PGM=IEBGENER
//OUTDS1 OUTPUT CLASS=P,FSSDATA='printer=pcl1',
// FORMDEF=MYDEF,USERLIB=USERX.MYLIB
//DD1 DD SYSOUT=(),OUTPUT=*.OUTDS1,DSNAME=&&MYFILE
```

4.9 Printing to e-mail addresses

With Infoprint Server, you can send print output to one or more e-mail addresses. This support lets you easily distribute documents over the Internet and lets e-mail recipients view the output and print it only when necessary. Some features of e-mail support are:

- IP PrintWay sends each document as an e-mail attachment.
- The data in the attachment can be in any format, including PDF, PostScript, AFP, and text format.
- Data can be transformed to another format that is suitable for viewing on a workstation. For example, you can transform:
  - Traditional line data and VTAM application data (for example, CICS and IMS data) to text format.
  - Traditional line data, VTAM application data, and AFP data to PDF or PostScript format. An Infoprint transform product is required.
  - PCL data and PostScript data to AFP format. An Infoprint transform product is required.
4.9.1 Using IP PrintWay for e-mail destinations

The Infoprint Server transforms can be invoked by the following Infoprint Server components and features:

**IP PrintWay extended mode**

IP PrintWay extended mode transforms data in a spool data set before the data is sent to a printer or to an e-mail destination.

**IP PrintWay basic mode**

IP PrintWay basic mode sends data in a spool data set to Print Interface. Print Interface transforms the data and writes the transformed data to a new output data set on the JES spool. IP PrintWay then prints the transformed data or sends it to an e-mail destination. The Resubmit for filtering indicator in a printer definition, shown in Figure 4-19 on page 179, controls the IP PrintWay basic mode transformation (filter) processing.

**OUTPUT JCL statement**

IP PrintWay can use the IP address or the e-mail address in the named printer definition to find the printer or e-mail destination. However, if you specify an IP address in the DEST=IP: parameter of the OUTPUT JCL statement, or an e-mail address in the MAILTO parameter of the OUTPUT JCL statement, IP PrintWay uses that address to find the printer or e-mail destination instead of the address in the printer definition.

**Tip:** You might want to specify the DEST=IP: parameter or MAILTO parameter when your administrator has not created a printer definition for your printer or e-mail destination.

### Specifying the printer definition name

You can specify the name of the printer definition that your administrator created for the printer or e-mail destination in the Printer Inventory. Use this definition as follows:

- Specify the printer definition name in the FSSDATA='printer' parameter on the OUTPUT JCL statement and you can use the FSSDATA='printer' parameter with both IP PrintWay basic mode and IP PrintWay extended mode, as shown in Figure 4-16.
- However, if you use the Print Interface subsystem, you instead specify the printer definition name in the SUBSYS parameter on the DD JCL statement. The printer definition name is case-sensitive and must exactly match the name of the printer definition created by your administrator in the Infoprint Server Printer Inventory.
Specifying the printer’s IP address

To print a data set on a printer for which your administrator has not created a printer definition, specify the DEST=IP: parameter on the OUTPUT JCL statement, as shown in Figure 4-16. Specify the IP address or host name of the printer or the system to which the printer is attached. When you specify DEST=IP:, you must also specify either the PRTQUEUE parameter or the PORTNO parameter on the OUTPUT JCL statement.

If you also specify the name of a printer definition in either the FSSDATA or SUBSYS parameter, IP PrintWay uses printing options specified in that printer definition, but uses the printer’s IP address specified in the DEST=IP: parameter. If you do not specify the name of a printer definition, IP PrintWay uses printing options specified in the IP PrintWay default printer definition.

If the resubmit for filtering function is used in IP PrintWay basic mode, do not code the DEST=IP:, PRTQUEUE, and PORTNO parameters on the OUTPUT JCL statement because IP PrintWay ignores these parameters and instead uses the IP address, print queue name, and port number in the printer definition.

4.9.2 Infoprint Server sendmail support

Before you begin using e-mail, you need to customize sendmail. When the e-mail protocol is selected in a printer definition, IP PrintWay uses z/OS UNIX sendmail to prepare and send e-mails to the recipients listed in the printer definition. Sendmail is a mail transfer agent provided with z/OS Communications Server that provides enhanced SMTP support.

Sending output to an e-mail destination

When you submit a print request, you can send the file to an e-mail destination instead of to a printer. You can specify the e-mail addresses of the primary and secondary recipients in one of these ways:

- Your administrator can specify the e-mail address list of the recipients in the printer definition in the Printer Inventory.
- You can specify the e-mail addresses using optional JCL parameters.
- You can specify an address list in an alias file that your administrator defines to z/OS UNIX sendmail.
The e-mail has these characteristics:

- Each output data set is a separate e-mail attachment. The name of the attachment is the value specified in one of these:
  - The MAILFILE parameter on the OUTPUT statement for the output data set
  - The DSNNAME parameter on the DD statement for the output data set
  - The job name

You can specify these optional OUTPUT JCL parameters to customize your e-mails:

**MAILBCC**  The e-mail addresses of the “blind copy (bcc)” recipients of an e-mail. A bcc means that other recipients of the e-mail do not see the bcc recipient listed.

**MAILCC**  The e-mail addresses of the “copy (cc)” recipients of an e-mail. A cc means that other recipients of the e-mail can see the cc recipient listed.

**MAILFILE**  The file name of the attachment to an e-mail.

**MAILFROM**  The descriptive name or other identifier of the sender of an e-mail.

**MAILTO**  The e-mail addresses of the recipients of an e-mail.

**REPLYTO**  The e-mail address that recipients of an e-mail can reply to.

**TITLE**  The subject of the e-mail.

### JCL example to transform AFP data to a PDF

The JCL example in Figure 4-17 shows how to transform an AFP document to PDF format and send it to an e-mail address. This example shows how to specify JCL parameters for e-mailing the document (MAILTO, MAILFROM, REPLYTO, and MAILFILE).

The example assumes that:

- Printer definition mail exists in the Infoprint Server Printer Inventory with these characteristics:
  - The e-mail protocol is selected.
  - The AFP to PDF transform is specified.
- IP PrintWay prints output data sets in JES output class P.

```
//MAILJOB JOB #11,TRANSFORMER,MSGLEVEL=1,MSGCLASS=Z
//PRINT EXEC PGM=IEBGENER
//OUTDS1 OUTPUT CLASS=P,FSSDATA='printer@mail',
// FORMDEF=MYDEF,MAILTO='user@xyz.com',
// MAILFROM='John Sender',REPLYTO='secretary@xyz.com',
// MAILFILE='July report',TITLE='Monthly Report'
//DD1 DD SYSOUT=(),OUTPUT=*.OUTDS1
```

*Figure 4-17  JCL to send the output to an e-mail address*

### 4.9.3 IP PrintWay and basic mode

In Figure 4-18 on page 178, a batch job is created to print AFP data to e-mail address. With IP PrintWay basic mode, a transform in printer definition is required.
Batch job to send e-mail output

Figure 4-18 shows a batch job that when submitted does the following:

- During execution using the IEBGENER program, it takes the AFP data set, ROGERS.SAPI.LIST3820 and creates an output data set on the JES spool.
- When the job ends, the output data set created on the spool is selected by IP PrintWay because the destination, DEST=EMAIL10, and the SYSOUT class J are specified.
- The processing done by IP PrintWay is described in Figure 4-21 on page 181.

Printer Inventory

The IP PrintWay printer to process the output to the e-mail destination is defined by the administrator in the Printer Inventory.

```plaintext
//ROGERSZ   JOB   (POK,999),MSGCLASS=A,NOTIFY=ROGERS
//PRINT     EXEC PGM=IEBGENER
//SYSPRINT  DD  SYSOUT=A
//SYSUT2    DD  SYSOUT=J,DEST=EMAIL10
//SYSUT1    DD  DISP=SHR,DSN=ROGERS.SAPI.LIST3820
//SYSIN     DD  DUMMY
```

Defining Processing panel options for EMAIL10

The ISPF panel in Figure 4-19 on page 179 shows how to specify the valid data formats for a printer that can print AFP and text data. Notice that Line data is selected also as AFP data type. This EMAIL10 printer in the Filter field converts AFP data to a PDF using the transforms.

Resubmit for filtering option

This single-valued attribute indicates whether a filter in the filters attribute is to be used for data sets submitted as batch jobs to IP PrintWay basic mode. When resubmit-for-filtering=yes, IP PrintWay resubmits batch data sets to Print Interface. Print Interface calls the filter (if any) associated with the input data format and then writes the data to a new output data set on the JES spool for subsequent processing by IP PrintWay.

This process is illustrated in Figure 4-21 on page 181.
Figure 4-19  Processing panel to define a printer in the Printer Inventory

### e-mail protocol panel for EMAIL10

In an IP PrintWay printer definition, you can select the transmission protocol that IP PrintWay uses to transmit output data sets from the JES spool to the target system. The target system can be a printer, a print server, or an e-mail destination.

Select the IP PrintWay e-mail protocol if you want IP PrintWay to transmit data sets to one or more e-mail addresses over the Internet using the z/OS UNIX sendmail function that z/OS Communications Server provides.

IP PrintWay transmits e-mails to the e-mail addresses that you specify in the printer definition (shown in Figure 4-20 on page 180). Some job submitters can override the e-mail addresses during job submission. You can set up just one printer definition for the e-mail protocol because the job submitter can specify the e-mail addresses. You must specify a default e-mail address in this printer definition.

#### Protocol panel e-mail addresses

On the protocol panel shown in Figure 4-20, you can specify addresses of the e-mail recipients, as follows:

- Specify real e-mail addresses or alias names that are defined to sendmail.
  - An alias name represents one or more e-mail addresses.
  - Define this field in a printer definition with e-mail protocol.
  - To extend the length of this field, place your cursor on the word "extend" and press Enter.

- The e-mail addresses or alias names of the "copy (cc)" recipients of the e-mail. Other recipients can see the cc listed. An alias name represents one or more e-mail addresses and is a name defined to z/OS UNIX sendmail. This field is not required. An address that the job submitter specifies overrides this value.

- The e-mail addresses or alias names of the "blind copy (bcc)" e-mail recipients. Other recipients do not see the bcc listed. An alias name represents one or more e-mail
addresses and is a name defined to z/OS UNIX sendmail. This field is not required. An address that the job submitter specifies overrides this value.

- IP PrintWay always includes "userid@domainname" to identify the sender.
- The e-mail address or alias name that recipients of the e-mail can reply to.

```
Edit Command ==> ____________________________ E-mail Protocol

Printer definition name . EMAIL10________

To addresses
   . . paulroge@us.ibm.com_____________________ (more)
CC addresses
   . . _____________________________ (more)
BCC addresses
   . . _____________________________ (more)
From name . . ____________________________
Reply address . . __________________________
```

Figure 4-20  IP PrintWay protocol panel for an e-mail printer definition

**Batch job output flow**

Figure 4-21 on page 181 illustrates the process flow when the batch job is submitted and the final destination is an e-mail address, as follows:

1. The batch job is submitted and during its execution it creates an AFP output data set on the JES spool and the job completes execution.
2. The AFP data set on the spool is selected by an IP PrintWay writer based on its output class and output destination.
3. IP PrintWay determines from the Printer Inventory that the resubmit for filtering flag is on and for AFP output that a data stream transform should be used to convert the output to a PDF.
4. IP PrintWay passes the transform request to the Print Interface for processing.
5. Print Interface passes the request to the Transform Manager and the proper transform is selected to transform the data from AFP to PDF.
6. The PDF created is then placed back on the JES spool.
7. IP PrintWay is notified to select the PDF output from the JES spool.
8. IP PrintWay transmits the PDF output to the IP addresses specified in the protocol definition for the EMAIL10 printer.
4.10 National languages and document-codepage attribute

A code page associates a code point and a graphic character identifier for each graphic character (hexadecimal code) supported by the code page and specifies how code points that are not valid are to be processed. Within a given code page, a code point can have only one specific meaning.

In Infoprint Server, the document-codepage single-valued attribute is used to specify the code pages for documents to be printed. Infoprint Server uses the code page for the source when converting documents from ASCII to EBCDIC, from EBCDIC to ASCII, or between EBCDIC code pages. IP PrintWay extended mode always uses the z/OS iconv utility to convert between code pages. It does not use standard or customized TCP/IP translate tables, as IP PrintWay basic mode can.

The way the default document-codepage is determined by the Print Interface depends on how the print request was submitted. If the print request was submitted from:

- If the print request was submitted from the local z/OS system, Print Interface uses the code page for the z/OS locale (usually an EBCDIC code page).
- If the print request was submitted from a remote system, Print Interface uses the ASCII code page specified in the Infoprint Server aopd.conf configuration file or, if not specified, code page ISO8859-1.

Using NetSpool

NetSpool uses the EBCDIC code page specified in the Infoprint Server aopd.conf configuration file. If not specified, code page IBM-1047 will be used.
Using IP PrintWay

Basic mode uses the code page in the default-document-codepage attribute in the printway-fss object class or, if not specified, code page IBM-1047 is used.

Extended mode uses the EBCDIC code page specified in the Infoprint Server aopd.conf configuration file, or, if not specified, code page IBM-1047 is used.

In most cases, the default value is suitable. One exception is when you need to print ASCII documents sent with the `lp` command. In this case, specify an ASCII code page either in this attribute or on the `lp` command.

For code page names, see z/OS XL C/C++ Programming Guide, SC09-4765.

National language example

The example shows how to transform and print a document created using the IBM-278 (Finnish-Swedish) code page. The document is stored in the MVS data set USER.FINSWED.DOCU.

The document was printed from the z/OS shell in a system operating in the USA locale En_US.IBM-037 (code page IBM-037) with an `lp` command on an IPDS printer:

```
lp -d IPDPOK "//'user.finswed.docu"
```

The Swedish part of the printed output is unreadable and thus, the printed document is unusable:

```
Swedish: ¦, ¦, h¦ ¦ -- English translation: island, island, hay island (IBM-278)
```

The hexadecimal code for the IBM-278 code page character ö is 6A. The hexadecimal code 6A maps to IBM-1047 code page character !.

In order to make the IPDS printer output readable, the codepage and chars attributes are used on the `lp` commands:

1. `lp -d IPDPOK -o "cha = 60D8 do-co = IBM-278" "//user.finswed.docu"
   
2. `lp -d IPDPOK -o "cha = 6060 do-co = IBM-1047" "//user.finswed.docu"
   
3. `lp -d IPDPOK -o "cha = 6060" "//user.finswed.docu"

All of the `lp` commands print readable output on the IPDS printer:

```
Swedish: ö, ö, hö ö -- English translation: island, island, hay island (IBM-278)
```

1. On the first `lp` command, the `-o` options’s attribute chars specify coded font 60d8 (code page INTERNATIONAL #5 – character set GOTHIC TEXT LATIN1) and document-codepage IBM-278. Infoprint Server Print Interface translates the line data document from IBM-278 to IBM-1047 before writing the data to the JES spool. PSF can write the IBM-1047 code page document correctly using the 60D8 character set.

   Since the document is translated to code page IBM-1047 on spool, coded font 6060 (code page FINLAND/SWEDEN- CECP – character set GOTHIC TEXT LATIN1) cannot be used to print the document when the document-codepage IBM-278 is also specified on the `lp` command.

2. On the second `lp` command, the `-o` options’s attribute chars specify coded font 6060 (code page FINLAND/SWEDEN- CECP – character set GOTHIC TEXT LATIN1) and document-codepage IBM-1047 (the Infoprint Server default). Infoprint Server Print Interface does not translate the line data document from IBM-278 to IBM-1047 before
writing the data to the JES spool. PSF, however, can write the IBM-1047 code page
document correctly using the 6060 character set.

3. The third `lp` command works the same as the second `lp` command.

To transform and print the IBM-278 code page document on printers controlled by IP
PrintWay in a system operating in the USA locale En_US.IBM-037 (code page IBM-037),
enter in a z/OS shell command:

```
lp -d POK45AN -o "cha = 6060 do-co = IBM-278" "//user.finswed.docu"
```

On the `lp` command the `-o` options's attribute chars specify coded font 6060 (code page
FINLAND/SWEDEN- CECP - character set GOTHIC TEXT LATIN1) and the
document-codepage IBM-278 code page. Infoprint Server AFP2xxx transforms needs to
know both the code font and document-codepage to create correct output.
Appendix A. Sample REXX programs

This appendix contains sample REXX programs for Infoprint Server transforms. These programs can be downloaded from the ITSO Redbooks site and used as sample programs to assist you in implementing the transforms. The following REXX execs are provided:

- XFDERRE REXX exec to view stderr files
  XFDERR REXX TSO/E command invokes the UNIX System Services ISPF shell(ISHELL) for the transform stderr files that are not empty.

- ITSOCF30 ISPF edit macro
  Generates JCL and EXEC statements for the AOXCF30 font-conversion program.

- AFPFIX REXX exec
  This routine examines the input file for AFP structured field for records that may have been reformatted during a transform to the file system and formats into records with each record consisting of a structured field.

Instructions for downloading these programs are also provided.
A.1 XFDERRRE REXX exec

This REXX exec invokes the z/OS UNIX ISHELL to transform stderr files.

```rexx
/*
   *---------------------------------------------------------------------------*/
/* rexx XFDERR */
/* The XFDERR REXX TSO/E command invokes the UNIX System Services ISPF shell
   (ISHELL) for the transform stderr files that are not empty. The format of a
   transform error file is "transform_name.#.stderr". The XFDERR REXX accepts as
   an argument a character string that is used as a filter for the transform
   stderr
   file selection. The filter can be a whole transform_name or any part of it.
   The stderr files with data in are presented the last changed first.
   Examples: XFDERRRE /* select all not-empty xfd stderr files */
              XFDERRRE 2a /* select pcl2afp, pdf2afp, and ps2afp files */
   *---------------------------------------------------------------------------*/

Parse arg arg
if sysvar(SYSISPF) <> "ACTIVE" then do /* ISPF is required */
   parse source src
   parse var src . . c .
   say c "must be invoked from an active ISPF environment"
   exit
end
numeric digits 16
brg = translate(arg)
z = wordpos("DB",brg)
if z ^= 0 then do
   x = MSG("ON")
   Trace "I"
   arg = delword(arg,z,1)
end
else Trace "O"
   address "ISPEXEC" "VGET (ZSCREEN) SHARED"
   dsn = "xfdplib.$tmp$" || ZSCREEN /* temporary ISPF ISPPLIB */
   z = LISTDSI(dsn) /* check tmp ISPPLIB */
   if z < 16 then do /* already in use - split */
      opmsg = "Temporary ISPPLIB creation failed. Exiting.."
      call sayit opmsg
      exit 16
   end
   z = crpan() /* create tmp ISPPLIB */
   if z <= 0 then do /* no-go -- say it & exit */
      pmsg = "Temporary ISPPLIB creation failed. Exiting.."
      call sayit pmsg
      exit 16
   end
   k = 0; opmsg = ""
   z = value("se."k,k)
   parse upper var arg xfdtyp /* take xforn type if any */
   dir = "/var/Printsrv/xfd/" /* xform stderr directory */
/* */
call syscalls "ON"
```

Infoprint Server for z/OS Infoprint Transforms
address syscall
call bpxwunix 'ls -to /var/Printsrv/xfd/ ',,,d.
k = 0                                /* count of selected stderrs */
/* */
address "ISPEXEC" "CONTROL ERRORS RETURN"
address "ISPEXEC" "TBCREATE XFDERRT NAMES(FIL MODA TI VI) REPLACE NOWRITE"

sel = ""
vi = ""
do i=1 to d.0
    parse var d.i . . . bc mo da ti fil .
    if pos("stderr",fil) <> 0 then do /* select xfd stderr files */
        file = d.j
        if bc > 0 then do              /* select only files with data*/
            z = translate(fil)            /* upper                      */
            parse var z y ".".
            if pos(xfdtyp,y) <> 0 | xfdtyp = "" then do
                k = k + 1
                z = dir || fil
                z = value("se."k,z)         /* save path and file name */
                moda = mo da
                fil = se.k
                address "ISPEXEC" "TBADD XFDERRT"
                se.0 = k
            end
        end
    end
    end
/* */
if se.0 > 0 then do                  /* show transform srterr files */
    address "ISPEXEC" "TBTOP XFDERRT"
    tdr = 0
    address "ISPEXEC" "TBDISPL XFDERRT PANEL(XFDERTP) AUTOSEL(NO) ROWID(RID)"
    tdr = rc
    do while tdr < 8
        top = ZTDTOP
        if sel <> "" then do
            sel = ""
            vi = "Browsed"
            address "ISPEXEC" "TBPUT XFDERRT"
            address "ISPEXEC" "CONTROL DISPLAY SAVE"
            cmd = "OBROWSE" fil         /* invoke ISHELL for the file */
            address "ISPEXEC" "SELECT CMD("cmd")"
            z = rc
            address "ISPEXEC" "CONTROL DISPLAY RESTORE"
        end
    address "ISPEXEC" "TBTOP XFDERRT"
    address "ISPEXEC" "TBSKIP XFDERRT NUMBER("top")"
    address "ISPEXEC" "TBDISPL XFDERRT PANEL(XFDERTP) AUTOSEL(NO) ,
    "ROWID(RID) CSRROW("rid")"
    tdr = rc
end
end
/* */
address "ISPEXEC" "TBCLOSE "
address "ISPEXEC" "LIBDEF ISPPLIB"
z = MSG("OFF")
address "TSO" "del" dsn
z = MSG(z)
if k = 0 then do
  opmsg = opmsg k "not_empty" xfdtyp "transform strerr files found."
call sayit opmsg
end
return
sayit:
  parse arg xyz
  if xyz <> "" then zedlmsg = xyz
  else zedlmsg = ""
  If sysvar(SYSISPF) = "ACTIVE" then do
    zedmsg = ""
    address "ISPEXEC" "SETMSG MSG(ISRZ000)"
  end
  else Say zedlmsg
return 0
crpan:
/* ispplib XFRERRTP panel data */
  p.1 = ")ATTR "
  p.2 = " @ TYPE(OUTPUT) INTENS(LOW)"
  p.3 = " # TYPE(OUTPUT) INTENS(HIGH)"
  p.4 = ")BODY "
  p.5 = "%----------------- Infoprint Server - Transform Error Files",
        "-------------------"
  p.6 = "%COMMAND ===>_ZCMD                           "
        " %SCROLL ===>_AMT +"
  p.7= "+ "
  p.8 = "%A File                                  Date     Time      Message"
  p.9 = ")MODEL "
  p.10= "_ZFIL                                  @MODA    @TI       #VI ">
  p.11= ")INIT "
  p.12= " .ZVARS = '(SEL)' "
  p.13= " &AMT= HALF "
  p.14= ")REINIT "
  p.15= ")PROC "
  p.16= ")END "
  p.17= " "
  p.0 = 16
  address "TSO"
  z = MSG("OFF")
  "del" dsn
  z = MSG(z)
  frc = 0
/* create XFRERRTP panel       */
  "alloc ds("dsn") XFDFERTP) uni(sysallda) spa(1 1)",
  "tra dir(1) recff(f) lrecf(80) dd(xfdpanel) reu new"
  frc = max(frc,rc)
  "EXECIO * DISKW XFDPANEL (STEM P. FINIS "
  frc = max(frc,rc)
  "free dd(xfdpanel)"
  address "ISPEXEC" "LIBDEF ISPPLIB DATASET ID("dsn") STACK"
  frc = max(frc,rc)
return frc
A.2 ITSO CF30 ISPF edit macro

This REXX exec generates JCL for the AOXCF30 program.

```rexx
/* rexx ITSO CF30 */
ITSO CF30 ISPF edit macro to generate JCL and EXEC statements for the AOXCF30 font-conversion program. SYS1.SAMPLIB(AOXCF30J) JCL is used as the JCL skeleton. The //C0XXXXXX EXEC PROC=SCALE,NAME=C0xxxxxx staments are generated from the member names of the font library to be converted.

Usage: In ISPF invoke edit for an empty JCL data set and then enter on the edit panel's Command line ITSO CF30 'font_library_name'

The 'font_library_name' is the same as the font library data set name on the SYSUT1 DD-statement in the JCL SYS1.SAMPLIB(AOXCF30J).

*/

trace "O"
cn = 0; jc = 0; mc = 0
address "ISPEXEC" "CONTROL ERRORS RETURN"
em = "MACRO"
address "ISREDIT" "MACRO (FLIB)"
if flib = "" then em = "No fontlib argument"
else do
  address "ISREDIT" "EXCLUDE P=' ' ALL"
  address "ISREDIT" "DELETE ALL X "
  address "ISREDIT" "RESET"
  if rc = 0 then do
    z = LISTDSI(flib)
    em = "LISTDSI" flib "RC =" z "RSN =" sysreason
    if z = 0 then do
      if rc = 0 then do
        em = "ALLOC"
        address "TSO" "alloc dd(flib) ds("flib") sh reu"
        if rc = 0 then do
          em = "LMINIT"
          address "ISPEXEC" "LMINIT DATAID(DI) DDNAME(FLIB) ENQ(SHR)"
          if rc = 0 then do
            em = "LMOPEN"
            address "ISPEXEC" "LMOPEN DATAID(DI) OPTION(INPUT)"
            if rc = 0 then do
              lmrc = 0
              do while lmrc = 0
                em = "LMMLIST"
                address "ISPEXEC" "LMMLIST DATAID(DI) MEMBER(MEM)" , "STATS(NO) PATTERN(C0*)"
                lmrc = rc
                if cn = 0 then do
                  em = "COPY"
                  address "ISREDIT" "COPY 'SYS1.SAMPLIB(AOXCF30J)' AFTER .ZL"
                  lmrc = rc
                  jc = jc + 1
              end
              if lmrc = 0 then do
```

/* rexx ITSOCF30 */
ITSO CF30 ISPF edit macro to generate JCL and EXEC statements for the AOXCF30 font-conversion program. SYS1.SAMPLIB(AOXCF30J) JCL is used as the JCL skeleton. The //C0XXXXXX EXEC PROC=SCALE,NAME=C0xxxxxx staments are generated from the member names of the font library to be converted.

Usage: In ISPF invoke edit for an empty JCL data set and then enter on the edit panel's Command line ITSO CF30 'font_library_name'

The 'font_library_name' is the same as the font library data set name on the SYSUT1 DD-statement in the JCL SYS1.SAMPLIB(AOXCF30J).

*/

trace "O"
cn = 0; jc = 0; mc = 0
address "ISPEXEC" "CONTROL ERRORS RETURN"
em = "MACRO"
address "ISREDIT" "MACRO (FLIB)"
if flib = "" then em = "No fontlib argument"
else do
  address "ISREDIT" "EXCLUDE P=' ' ALL"
  address "ISREDIT" "DELETE ALL X "
  address "ISREDIT" "RESET"
  if rc = 0 then do
    z = LISTDSI(flib)
    em = "LISTDSI" flib "RC =" z "RSN =" sysreason
    if z = 0 then do
      if rc = 0 then do
        em = "ALLOC"
        address "TSO" "alloc dd(flib) ds("flib") sh reu"
        if rc = 0 then do
          em = "LMINIT"
          address "ISPEXEC" "LMINIT DATAID(DI) DDNAME(FLIB) ENQ(SHR)"
          if rc = 0 then do
            em = "LMOPEN"
            address "ISPEXEC" "LMOPEN DATAID(DI) OPTION(INPUT)"
            if rc = 0 then do
              lmrc = 0
              do while lmrc = 0
                em = "LMMLIST"
                address "ISPEXEC" "LMMLIST DATAID(DI) MEMBER(MEM)" , "STATS(NO) PATTERN(C0*)"
                lmrc = rc
                if cn = 0 then do
                  em = "COPY"
                  address "ISREDIT" "COPY 'SYS1.SAMPLIB(AOXCF30J)' AFTER .ZL"
                  lmrc = rc
                  jc = jc + 1
              end
              if lmrc = 0 then do
```
A.3 AFPFIX REXX exec

This REXX exec formats AFP structured field records.

/* AFPFIX rexx */
/* This routine examines the input file for AFP structured field */
/* records that may have been reformatted during a transform to HFS */
/* or copy from the HFS to a MVS data set. */
parse arg arg
parse upper var arg z
z = wordpos("DB",z) /* traceing requested? */
if z ^= 0 then do /* yes */
x = MSG("ON")
Trace "I"
arg = delword(arg,z,1)
end
else Trace "O" /* no */
parse var arg ids ods rest
say arg
if ids = '?' | ids = '' | ods = '?' | rest ^= '' then do
say ' ',
say 'AFPFIX takes an unformatted (HFS) AFP file and formats it
say ' into records, each record consisting of a structured field.'
say ' ',
say 'Format:'

say '                                                            '
say ' AFPJOIN  input_ds output_ds                                '
say '          /HFS_file                                         '
say '                                                            '
say '    input_ds      The name of the input data set.           '
say '    /HFS_file     The full PATH name of the input HFS file. '
say '                                                            '
say '    output_ds     The name of the output data set. Optional.'
say '                                                            '
exit
end

infile = "AFPIN"                           /* DD-name for input        */
fm = substr(strip(ids,"B","'"),1,1)
if fm <> "/" then do                       /* MVS data set             */
    rest = LISTDSI(ids)                      /* Check input validity     */
    if rest <> 0 then do
        say ids": LISTDSI RC =" rest " RSN =" SYSREASON " SYSMSGLVL1
        say SYSMSGLVL2
        exit
    end
/* Compute output space*/
if pos("BLOCK",SYSUNITS) <> 0 then rest ="("SYSBLKSIZE")"
else rest = ""
spa = SYSUNITS||rest "SPACE("SYSPRIMARY SYSSECONDS") RELEASE"
end
else do                                    /* UNIX file                */
    ids = strip(ids,"B","'")
    y = outtrap("OF.")
    "alloc dd("infile") path(""ids"') pathdis(keep keep) pathmod(sirusr) reu" ,
    "lrec(80) "
    y = outtrap("OFF")                       /* Check input availability */
    if rc <> 0 then do; say ids "ALLOC PATH RC =" rc." of.1; exit; end
    spa = "spa(10 10) cyl RELEASE"
end
if ods = "" then do                        /* Name the output data set */
    if fm = "/" then do                   /* Input is UNIX file and     */
        z = lastpos("/",ids)            /* no output name given.     */
        if z <> 0 then do
            wods = substr(ids,z+1)
            yz. = ''
            yz.0 = 1
            do 20
                y = lastpos(".",wods)
                if y <> 0 then do
                    z = pos(".",wods)
                    w = z
                    if z > 8 then w = 9
                    v = value("yz."yz.0,substr(wods,1,w-1))
                    yz.0 = yz.0 + 1
                    wods = substr(wods,z+1)
                end
            else do
                if length(wods) > 8 then w = 8
                v = value("yz."yz.0,substr(wods,1))
            end
        end
        end
    end
leave
end
end
wods = ""
do i = 1 to yz.0
wods = wods || yz || "."
end
if length(wods) > 26 then wods = substr(wods,1,26)
wods = strip(wods,"T","."")
end
else wods = translate(ids,",","."") /* Input is a MVS data set */
x = words(wods) /* and no output name */
select
when x = 1 & substr(ids,1,1) <> "" then ,
wods = strip(wods) "LIST3820"
when x > 1 & substr(ids,1,1) <> "" then ,
wods = strip(delword(wods,x,1)) "LIST3820"
when x = 2 & substr(ids,1,1) = "" then ,
wods = strip(delword(wods,1,1)) "LIST3820"
when x > 2 & substr(ids,1,1) = "" then do
wods = delword(wods,1,1)
wods = strip(delword(wods,x,1)) "LIST3820"
end
otherwise wods = "LIST3820"
end
ods = translate(wods,".",""")
end
rest = SYSDSN(ods) /* Check output data set */
select /* status */
when rest = "OK" then do
rest = ""
do while rest <> "Y" & rest <> "N"
say ods": Data set exists - Do you want to recreate it? (Y/N)"
parse upper pull rest
if substr(rest,1,1) = "N" then do; say "Exiting..."; exit; end
else do
x = outtrap("OF.")
"DELETE" ods
x = outtrap("OFF")
if rc <> 0 then do
do i= 1 to OF.0
say OF.i
end
say "*** Error deleting" ods
Exit 16
end
end
end
when rest = "DATASET NOT FOUND" then say ods": Data set assigned to output"
otherwise do
say ods": SYSDSN =" rest
exit
end
outfile = "AFPOUT"                         /* Allocate input data set */
if fm <> "/" then do                       /* (Input file is allocated)*/
    "ALLOC DD("infile") DS("ids") REU SH"
if RC > 0 then do
    say "*** Error allocating" ids
    Exit 16
end
/* Allocate output data set */
attr = "DSORG(PS) RECFM(V B M) LRECL(8205)"
"ALLOC DD("outfile") DS("ods") REU NEW" attr spa
if RC > 0 then do
    say "*** Error allocating" ods
    Exit 16
end /* */
call AFPDS_fix infile, outfile             /* Invoke reformat */
x = unalloc()
Exit

AFPDS_Fix:
/* The routine examines the input file for AFP structured field */
/* records that may have been reformatted during a transform to HFS */
/* or copy from the HFS to a MVS data set. During such a copy, the */
/* structured fields are treated as a stream off data and the */
/* original records are lost. The transmitted file may have many */
/* structured fields in a single record, or an individual */
/* structured field may span multiple records. */
/* */
/* */
/* When properly formed AFP structured fields are detected they */
/* are written to the output as 1 structured field per output */
/* record. Data not occuring within a structured field is written */
/* out according to the following rules: */
/* */
/* 1. If no structured field introducer is present in the */
/*    record, the entire record is written unaltered. */
/* */
/* 2. If a structured field begins within the record, the data */
/*    preceding that structured field is written as a record. */
/* */
/* 3. Data falling between two valid structured fields is */
/*    written as a record. */
/* */
/* 4. Data following a valid structured field is written as a */
/*    record, with one exception. If the structured field is */
/*    IPO, IPS, IMM, or IDM and the remainder of the record in */
/*    which it is found is blank, it is assumed to be a */
valid structured field control record imbedded in a fixed
length record data file. The trailing blanks are stripped
and ignored.

This logic has the following effects:

1. Files with no structured field content are transcribed verbatim.

2. Files containing only structured fields are written out with 1 structured field per record.

3. Files with a mixture of structured field records and other data may or may not be reconstructed accurately. Since information about the original record lengths has been lost, only AFP records can be accurately reconstructed. If non-AFP data are isolated to their own records, the reconstruction should be accurate.

---------------------------------------------
/* arg infile, outfile */
/* Prime nextrec buffer */ nextrec = readrec()
/* Clear currec */ currec = ""
/* Main execution loop */
/* */
/* Get an output rec */ outrec = get_outrec()
/* If end of file, quit */ if outrec = "*EOF*" then leave
/* */
/* Write output rec */ "EXECIO 1 DISKW" outfile "(STEM OUTREC"
/* End main loop */ end
/* */
/* Close output file */ "EXECIO 0 DISKW" outfile "(FINIS"
/* Close input file */ "EXECIO 0 DISKR" infile "(FINIS"
/* Scram */ return 0
/* */
---------------------------------------------
/* Get Next Output Record */
/* This routine isolates the next record to be written to the output file. CURREC contains the current data record, NEXTREC contains the next record from the input file. When CURREC is fully processed, NEXTREC is moved to CURREC and a new record is read into NEXTREC from the input file. The logic isolates the next output record using the rules described earlier. The isolated record is returned to the caller and stripped from CURREC. */
/* */
/* Get_Outrec: procedure expose currec nextrec infile */
/* If currec empty */ if length(currec) < 1 then do
/* move in nextrec and */ currec = nextrec
/* read next record */ nextrec = readrec()
/* */ end
/* Hit eof, quit */ if currec = "*EOF*" then return currec
/* Start at position 0 */ candidate_pos = 0
/*Isolation loop */    Do forever
/*Look for "!"*/       candidate_pos = pos("!",currec,candidate_pos+1)
/*If none*/       if candidate_pos = 0 then do
/* return whole record*/          outrec = currec
/* to caller*/          currec = ""
/* */          return outrec
/* */       end
/*Found a "!"*/       else do
/*Check for strfld rec*/          strlen = ver_strfld(currec||nextrec)
/*If not valid,*/          if strlen = -99 then do
/* bump position and */             candidate_pos = candidate_pos + 1
/* look for next "!"*/             iterate
/* */          end
/*Found good strfld */          end
/*Must loop to read */          do while strlen > length(currec)
/* oops*/             if nextrec = "*EOF*" then do
/* bad str fld at end */                say "Found incomplete structured field",
/* save what we have */                "record at end of input file"
/* blank currec */                currec = ""
/* return incompl rec */                return outrec
/* */             end
/* append nextrec */             currec = currec||nextrec
/* get another rec */             nextrec = readrec()
/*end loop */          end
/*Isolate strfld rec */          strfld = substr(currec,1,strlen)
/*Update currec */          currec = substr(currec,strlen+1)
/*Str=IPO IPS IMM IDM */        chkstrng = x2c('d3afd840d3af5f40d3abcc40d3abca')
/*If rec one of these*/          if 0 < wordpos(mneumonic,chkstrng),
/* and remainder blank*/           & strip(currec) = ""
/* assume fixed len */          then currec = ""
/* str fld rec */         end
/*Return to caller */         return strfld
/* */       end
/* */       end
/*Read a Physical Record */
/* This routine reads the next physical record from the input file */
/* and returns it to the caller. */
/* */
Readrec: procedure expose infile
/* */       "EXECIO 1 DISKR" infile "(STEM DISKREC"
/* */       DISKREC = DISKREC1
/* */       if rc > 0 then return "*EOF*"
/* */       return diskrec
/* */
/* Verify a Structured Field */
/* This routine verifies that the data string passed as an argument */
/* conforms to the rules for a valid AFP structured field. If */
A.4 Instructions for downloading the sample programs

In the download instructions, do the following:

- \textit{myuser} in these instructions is your User ID, so replace your User ID where you see myuser; \textit{my.email@xx.com} is your email address.
- Retrieve the files sample.programs.unload.bin in binary to your workstation or directly into your z/OS system from the Redbooks web site.
  
  This can be done using your favorite browser (not shown here) or an ftp session.
- Following are samples of how to do this with ftp sessions.

\begin{verbatim}
Scenario 1: Getting the data directly to your z/OS system

ftp www.redbooks.ibm.com
User: anonymous
Password: my.email@xx.com
cd redbooks/SG247498/
lcd 'myuser'
locsite blk=3120 lrecl=80 recfm=fb
binary
get sg247498.clist.unload.bin sg247498.clist.unload
quit

Note: You may also decide to pre-allocate the unloaded files as XMIT-ed sequential files instead of using the LOCSITE setting.
\end{verbatim}

Figure A-1 Scenario to download directly to your z/OS system
Scenario 2: Getting the files to your workstation first

ftp www.redbooks.ibm.com
User: anonymous
Password: my.email@xx.com
cd redbooks/SG247498/
binary
get sg247498.clist.unload.bin
quit

The following example shows how to transfer the files to z/OS from your workstation afterwards.

ftp my.zos.system
User: myuser
Password: mypasswd
cd 'myuser'
binary
quote site blk=3120 lrecl=80 recfm=fb
put sg247498.clist.unload.bin sg247498.clist.unload
quit

Final installation steps

Afterwards run the following command on your z/OS system:

tso receive indsn('myuser.sg247498.clist.unload')

On display of the following messages you may request to restore the original PDS data set with a desired new data set name (You may also decide to rename the data set after completion of the receive command.):

INMR901I Dataset ROGERS.SG247498.CLIST from ROGERS on WTSCPLX4
INMR906A Enter restore parameters or 'DELETE' or 'END' +

Here is a sample to set the data set name.

dsn('myuser.sg247498.clist')
tso receive indsn(sg247498.clist.unload)
Dataset ROGERS.SG247498.CLIST from ROGERS on WTSCPLX4
Enter restore parameters or 'DELETE' or 'END' +
[enter]

Figure A-2  Scenario to download to your workstation

Figure A-3  Final install instructions
This appendix contains the list of fonts available when using data stream transforms.

- The PDF to AFP transform provides a set of fonts, which it uses to create a rasterized image of the data.
  
  If a PDF document specifies a font that is not embedded in the PDF document and is not provided with the transform, the transform substitutes another font that is the best match for the specified font. You cannot customize the way the transform performs font substitution.

- The PostScript to AFP transform provides a set of fonts which it uses to create a rasterized image of the data.
  
  If a PostScript document specifies a font that is not embedded in the PostScript document and is not provided with the transform, the transform substitutes another font that is the best match for the specified font. You cannot customize the way the transform performs font substitution.

The transforms use the default font in the transform configuration file. If no font is specified, the transforms use font X060D9.
# B.1 PDF to AFP and PostScript to AFP fonts

The following table lists the Ghostscript fonts that the PDF to AFP and PostScript to AFP transforms use. The Ghostscript fonts are functional equivalents of the corresponding PostScript fonts in that the typeface styles are similar and the font metrics are identical to provide the same pagination and line endings.

<table>
<thead>
<tr>
<th>Font name</th>
<th>Font alias</th>
<th>Font file name</th>
</tr>
</thead>
<tbody>
<tr>
<td>A028-Ext</td>
<td>Albertus-ExtraBold</td>
<td>a028017t.pfb</td>
</tr>
<tr>
<td>A028-Med</td>
<td>Albertus-Medium</td>
<td>a028014t.pfb</td>
</tr>
<tr>
<td>A030-BolIta</td>
<td>Arial-Bolditalic</td>
<td>n019024t.pfb</td>
</tr>
<tr>
<td>A030-Bol</td>
<td>Arial-Bold</td>
<td>n019004t.pfb</td>
</tr>
<tr>
<td>A030-Ita</td>
<td>Arial-Italic</td>
<td>n019023t.pfb</td>
</tr>
<tr>
<td>A030-Reg</td>
<td>Arial-Regular</td>
<td>n019003t.pfb</td>
</tr>
<tr>
<td>AntiqueOlive-Bol</td>
<td>AntiqueOlive-Bold</td>
<td>a008006t.pfb</td>
</tr>
<tr>
<td>AntiqueOlive-Ita</td>
<td>AntiqueOlive-Italic</td>
<td>a008023t.pfb</td>
</tr>
<tr>
<td>AntiqueOlive-Reg</td>
<td>AntiqueOlive</td>
<td>a008003t.pfb</td>
</tr>
<tr>
<td>CenturySchL-BoldItal</td>
<td>NewCenturySchlbk-BoldItalic</td>
<td>c059036l.pfb</td>
</tr>
<tr>
<td>CenturySchL-Bold</td>
<td>NewCenturySchlbk-Bold</td>
<td>c059016l.pfb</td>
</tr>
<tr>
<td>CenturySchL-Ita</td>
<td>NewCenturySchlbk-Italic</td>
<td>c059033l.pfb</td>
</tr>
<tr>
<td>CenturySchL-Roma</td>
<td>NewCenturySchlbk-Roman</td>
<td>c059013l.pfb</td>
</tr>
<tr>
<td>ClarendonURW-BolCon</td>
<td>Clarendon-Condensed-Bold</td>
<td>c011046t.pfb</td>
</tr>
<tr>
<td>Coronet</td>
<td>Coronet</td>
<td>c093000t.pfb</td>
</tr>
<tr>
<td>Dingbats</td>
<td>ZapfDingbats</td>
<td>d050000t.pfb</td>
</tr>
<tr>
<td>GaramondNo8-Ita</td>
<td>Garamond-Kursiv</td>
<td>g043023t.pfb</td>
</tr>
<tr>
<td>GaramondNo8-MedIta</td>
<td>Garamond-KursivHalbfett</td>
<td>g043024t.pfb</td>
</tr>
<tr>
<td>GaramondNo8-Med</td>
<td>Garamond-Halbfett</td>
<td>g043004t.pfb</td>
</tr>
<tr>
<td>GaramondNo8-Reg</td>
<td>Garamond-Antiqua</td>
<td>g043003t.pfb</td>
</tr>
<tr>
<td>LetterGothic-BolIta</td>
<td>LetterGothic-BoldItalic</td>
<td>l021024t.pfb</td>
</tr>
<tr>
<td>LetterGothic-Bol</td>
<td>LetterGothic-Bold</td>
<td>l021004t.pfb</td>
</tr>
<tr>
<td>LetterGothic-Ita</td>
<td>LetterGothic-Italic</td>
<td>l021023t.pfb</td>
</tr>
<tr>
<td>LetterGothic-Reg</td>
<td>LetterGothic</td>
<td>l021003t.pfb</td>
</tr>
<tr>
<td>Mauritius-Reg</td>
<td>Marigold</td>
<td>m048000t.pfb</td>
</tr>
<tr>
<td>NimbusMonL-BoldObli</td>
<td>Courier-BoldOblique</td>
<td>n022024l.pfb</td>
</tr>
<tr>
<td>NimbusMonL-Bold</td>
<td>Courier-Bold</td>
<td>n022004l.pfb</td>
</tr>
<tr>
<td>NimbusMonL-RegObli</td>
<td>Courier-Oblique</td>
<td>n022023l.pfb</td>
</tr>
<tr>
<td>NimbusMonL-Reg</td>
<td>Courier</td>
<td>n022003l.pfb</td>
</tr>
<tr>
<td>NimbusMono-BolIta</td>
<td>CourierHP-BoldOblique</td>
<td>n022024t.pfb</td>
</tr>
<tr>
<td>NimbusMono-Bol</td>
<td>CourierHP-Bold</td>
<td>n022004t.pfb</td>
</tr>
<tr>
<td>NimbusMono-Ita</td>
<td>CourierHP-Oblique</td>
<td>n022023t.pfb</td>
</tr>
<tr>
<td>NimbusMono-Reg</td>
<td>CourierHP</td>
<td>n022003t.pfb</td>
</tr>
<tr>
<td>NimbusRomanNo4-BolIta</td>
<td>CGTimes-BoldItalic</td>
<td>n033034t.pfb</td>
</tr>
<tr>
<td>NimbusRomanNo4-Bol</td>
<td>CGTimes-Bold</td>
<td>n033014t.pfb</td>
</tr>
<tr>
<td>NimbusRomanNo4-LigIta</td>
<td>CGTimes-Italic</td>
<td>n033032t.pfb</td>
</tr>
<tr>
<td>NimbusRomanNo4-Lig</td>
<td>CGTimes</td>
<td>n033012t.pfb</td>
</tr>
<tr>
<td>NimbusRomanNo9-Ita</td>
<td>TimesNewRoman-Italic</td>
<td>n021023t.pfb</td>
</tr>
<tr>
<td>NimbusRomanNo9-MedIta</td>
<td>TimesNewRoman-BoldItalic</td>
<td>n021024t.pfb</td>
</tr>
<tr>
<td>NimbusRomanNo9-Med</td>
<td>TimesNewRoman-Bold</td>
<td>n021004t.pfb</td>
</tr>
<tr>
<td>Font Name</td>
<td>Font Name</td>
<td>Font Name</td>
</tr>
<tr>
<td>---------------------------</td>
<td>---------------------------</td>
<td>---------------------------</td>
</tr>
<tr>
<td>NimbusRomanNo9-Reg</td>
<td>TimesNewRoman</td>
<td>n021003t.pfb</td>
</tr>
<tr>
<td>NimbusRomNo9L-MedIttal</td>
<td>Times-BoldItalic</td>
<td>n021024l.pfb</td>
</tr>
<tr>
<td>NimbusRomNo9L-Medi</td>
<td>Times-Bold</td>
<td>n021004l.pfb</td>
</tr>
<tr>
<td>NimbusRomNo9L-RegIttal</td>
<td>Times-Italic</td>
<td>n021023l.pfb</td>
</tr>
<tr>
<td>NimbusRomNo9L-Regu</td>
<td>Times-Roman</td>
<td>n021003l.pfb</td>
</tr>
<tr>
<td>NimbusSanL-BoldCondIttal</td>
<td>Helvetica-Narrow-BoldOblique</td>
<td>n019064l.pfb</td>
</tr>
<tr>
<td>NimbusSanL-BoldCond</td>
<td>Helvetica-Narrow-Bold</td>
<td>n019044l.pfb</td>
</tr>
<tr>
<td>NimbusSanL-BoldIttal</td>
<td>Helvetica-BoldOblique</td>
<td>n019024l.pfb</td>
</tr>
<tr>
<td>NimbusSanL-Bold</td>
<td>Helvetica-Bold</td>
<td>n019004l.pfb</td>
</tr>
<tr>
<td>NimbusSanL-ReguCondIttal</td>
<td>Helvetica-Narrow-Oblique</td>
<td>n019063l.pfb</td>
</tr>
<tr>
<td>NimbusSanL-ReguCond</td>
<td>Helvetica-Narrow</td>
<td>n019043l.pfb</td>
</tr>
<tr>
<td>NimbusSanL-Regulital</td>
<td>Helvetica-Oblique</td>
<td>n019023l.pfb</td>
</tr>
<tr>
<td>NimbusSanL-Regu</td>
<td>Helvetica</td>
<td>n019003l.pfb</td>
</tr>
<tr>
<td>StandardSymL</td>
<td>Symbol</td>
<td>s050000l.pfb</td>
</tr>
<tr>
<td>U001-BolIta</td>
<td>Univers-BoldItalic</td>
<td>u001024t.pfb</td>
</tr>
<tr>
<td>U001-Bol</td>
<td>Univers-Bold</td>
<td>u001004t.pfb</td>
</tr>
<tr>
<td>U001Con-BolIta</td>
<td>Univers-Condensed-BoldItalic</td>
<td>u001064t.pfb</td>
</tr>
<tr>
<td>U001Con-Bol</td>
<td>Univers-Condensed-Bold</td>
<td>u001044t.pfb</td>
</tr>
<tr>
<td>U001Con-Ita</td>
<td>Univers-Condensed-MediumItalic</td>
<td>u001063t.pfb</td>
</tr>
<tr>
<td>U001Con-Reg</td>
<td>Univers-Condensed-Medium</td>
<td>u001043t.pfb</td>
</tr>
<tr>
<td>U001-Ita</td>
<td>Univers-MediumItalic</td>
<td>u001023t.pfb</td>
</tr>
<tr>
<td>U001-Reg</td>
<td>Univers-Medium</td>
<td>u001003t.pfb</td>
</tr>
<tr>
<td>URWBookmanL-DemiBoldItal</td>
<td>Bookman-DemiItalic</td>
<td>b018035l.pfb</td>
</tr>
<tr>
<td>URWBookmanL-DemiBold</td>
<td>Bookman-Demi</td>
<td>b018015l.pfb</td>
</tr>
<tr>
<td>URWBookmanL-LightItalic</td>
<td>Bookman-LightItalic</td>
<td>b018032l.pfb</td>
</tr>
<tr>
<td>URWBookmanL-Ligh</td>
<td>Bookman-Light</td>
<td>b018012l.pfb</td>
</tr>
<tr>
<td>URWChanceryL-MedIttal</td>
<td>ZapfChancery-MediumItalic</td>
<td>z003034l.pfb</td>
</tr>
<tr>
<td>URWClassico-BolIta</td>
<td>CGOmega-BoldItalic</td>
<td>o004024t.pfb</td>
</tr>
<tr>
<td>URWClassico-Bol</td>
<td>CGOmega-Bold</td>
<td>o004004t.pfb</td>
</tr>
<tr>
<td>URWClassico-Ita</td>
<td>CGOmega-Italic</td>
<td>o004023t.pfb</td>
</tr>
<tr>
<td>URWClassico-Reg</td>
<td>CGOmega</td>
<td>o004003t.pfb</td>
</tr>
<tr>
<td>URW GothicL-BookObli</td>
<td>AvantGarde-BookOblique</td>
<td>a010033l.pfb</td>
</tr>
<tr>
<td>URW GothicL-Book</td>
<td>AvantGarde-Book</td>
<td>a010013l.pfb</td>
</tr>
<tr>
<td>URW GothicL-DemiObli</td>
<td>AvantGarde-DemiOblique</td>
<td>a010035l.pfb</td>
</tr>
<tr>
<td>URW GothicL-Demi</td>
<td>AvantGarde-Demi</td>
<td>a010015l.pfb</td>
</tr>
<tr>
<td>URWPalladioL-BoldItal</td>
<td>Palatino-BoldItalic</td>
<td>p052024l.pfb</td>
</tr>
<tr>
<td>URWPalladioL-Bold</td>
<td>Palatino-Bold</td>
<td>p052004l.pfb</td>
</tr>
<tr>
<td>URWPalladioL-Ital</td>
<td>Palatino-Italic</td>
<td>p052023l.pfb</td>
</tr>
<tr>
<td>URWPalladioL-Roma</td>
<td>Palatino-Roman</td>
<td>p052003l.pfb</td>
</tr>
</tbody>
</table>
Transform environment variables

This appendix identifies the environment variables for each transform that can be specified when using the data stream transforms.

Environment variables let you specify:

- Height and width of the page
- Margins of the page
- Resolution of the printer
- Record length of AFP images
- Tracing options

You can specify environment variables when using the transforms, but all environment variables are optional.

Environment variables affect the behavior of the Transform Interface. Set environment variables only if you do not create the transform configuration files in the default locations.

If you set these environment variables, set them in both of these locations:

- `aopstart EXEC` Transform Interface uses environment variables specified in this file. For information about how to edit this file.
- `/etc/profile file` z/OS UNIX transform commands, such as ps2afp and remotexf, use environment variables specified in this file.
## C.1 Environment variables for PCL to AFP

<table>
<thead>
<tr>
<th>Environment Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>_BPX_JOBNAME</td>
<td>The job name for the transform. When you assign a different job name to each class of transform, the operator can manage the transform daemons more effectively. Specify a job name of 1 to 8 alphanumeric characters. Incorrect job names are ignored.</td>
</tr>
<tr>
<td></td>
<td><strong>Default:</strong> The job name is AOPXFD.</td>
</tr>
<tr>
<td></td>
<td><strong>Example:</strong> environment = (_BPX_JOBNAME-&gt; PCL2AFPD)</td>
</tr>
<tr>
<td>AOP_HORIZONTAL_MARGINS</td>
<td>The left and right margins of the page. Specify a number followed by one of the units: in Inches, mm Millimeters, or pel Pels (the default unit). Inch values and millimeter values can contain a decimal point. Pel values cannot.</td>
</tr>
<tr>
<td></td>
<td><strong>Default:</strong> AOP_HORIZONTAL_MARGINS-&gt; 0.167in</td>
</tr>
<tr>
<td>AOP_PAGE_HEIGHT</td>
<td>The height of the page. Specify a number followed by one of the units: in Inches, mm Millimeters, or pel Pels (the default unit). Inch values and millimeter values can contain a decimal point. Pel values cannot.</td>
</tr>
<tr>
<td></td>
<td><strong>Default:</strong> AOP_PAGE_HEIGHT-&gt; 11</td>
</tr>
<tr>
<td>AOP_PAGE_WIDTH</td>
<td>The width of the page. Specify a number followed by one of these units: in Inches, mm Millimeters, or pel Pels (the default unit). Inch values and millimeter values can contain a decimal point. Pel values cannot.</td>
</tr>
<tr>
<td></td>
<td><strong>Default:</strong> AOP_PAGE_WIDTH-&gt; 8.5in</td>
</tr>
<tr>
<td>AOP_RECLEN</td>
<td>The length of the AFP images the transform creates. Each output record contains an AFP image and an additional 17 bytes of information (for example, header information). For example, if you specify a value of 8 (8,000 bytes) for the AFP image in this environment variable, the length of each output record is 8,017 bytes.</td>
</tr>
<tr>
<td></td>
<td>Valid values are from 8 to 32, in units of 1,000 bytes. For example, 8 = 8,000. The default record length is suitable for most installations. However, a larger record length might improve performance.</td>
</tr>
<tr>
<td></td>
<td><strong>Default:</strong> AOP_RECLEN -&gt; 8</td>
</tr>
<tr>
<td></td>
<td><strong>Example:</strong> environment = (AOP_RECLEN -&gt; 16)</td>
</tr>
<tr>
<td>AOP_RESOLUTION</td>
<td>The resolution in pels per inch of the printer. The transform creates the AFP image with this resolution.</td>
</tr>
<tr>
<td></td>
<td><strong>Default:</strong> AOP_RESOLUTION-&gt; 240</td>
</tr>
<tr>
<td></td>
<td><strong>Example:</strong> environment = (AOP_RESOLUTION-&gt; 600)</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> If you specify a resolution that the printer does not support, PSF prints the image under most conditions, but sometimes with degraded results. A resolution of 300 pels typically produces good quality output on 300 or 600 pel printers. 300 pel output requires one-fourth the space and transmission time.</td>
</tr>
<tr>
<td>AOP_RESOURCE_PATH</td>
<td>The directory that contains the fonts that are provided with the transform. If you installed the fonts in /usr/lpp/Printsrv/pcl2afpv2/fonts directory, the default, you do not need to specify this environment variable. The transform always looks for fonts in the default font directory last.</td>
</tr>
<tr>
<td></td>
<td><strong>Default:</strong> AOP_RESOURCE_PATH-&gt; /usr/lpp/Printsrv/pcl2afpv2/fonts</td>
</tr>
<tr>
<td></td>
<td><strong>Example:</strong> environment = (AOP_RESOURCE_PATH-&gt; /etc/Printsrv/pcl2afpv2/fonts)</td>
</tr>
</tbody>
</table>

Table C-1  Environment variables for PCL to AFP transforms
### Appendix C. Transform environment variables

#### Table C-2   Environment variables added in z/OS V1R8

<table>
<thead>
<tr>
<th>Environment Variable</th>
<th>Description</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>AOP_VERTICAL_MARGINS</strong></td>
<td>The top and bottom margins of the page. Specify a number followed by one of these units: in Inches, mm Millimeters, or pel Pels (the default unit). Inch values and millimeter values can contain a decimal point. Pel values cannot.</td>
<td>AOP_VERTICAL_MARGINS -&gt; 0.167in</td>
</tr>
<tr>
<td><strong>AOPTRACEDIR</strong></td>
<td>The full path name of the directory where the transform writes trace information. You can specify the same directory for different transform classes. The name of the trace file identifies the transform and transform class, and contains a timestamp.</td>
<td>AOPTRACEDIR -&gt; /var/Printsrv/trace</td>
</tr>
<tr>
<td><strong>AOPTRACEON</strong></td>
<td>Turns tracing on. The transform traces all transform requests that use this transform class. Any value turns tracing on. To turn tracing off, do not specify this environment variable. Specify this environment variable only if IBM directs you to do so. Tracing can adversely affect performance.</td>
<td>Tracing is turned off.</td>
</tr>
</tbody>
</table>

#### Example environments:

- environment = {AOPTRACEDIR -> /var/Printsrv/xfd}
- environment = {AOPTRACEDIR -> .}

<table>
<thead>
<tr>
<th>Environment Variable</th>
<th>Description</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>AOP_FONT_SUBSTITUTION_MESSAGES</strong></td>
<td>Indicates whether the transform writes a message (AOP2500W) in the transform’s stderr file when it substitutes fonts in a document if no other errors occurred. The transform substitutes a similar font when it cannot find a font that the document requests. If a document is transformed with no font substitution, the transform does not write a message. Font-substitution messages let you determine if the transform has used substitute fonts.</td>
<td>AOP_FONT_SUBSTITUTION_MESSAGES -&gt; yes</td>
</tr>
<tr>
<td></td>
<td><strong>Valid values are:</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>yes The transform writes messages about font substitution.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>no The transform does not write messages about font substitution.</td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>Default:</strong> AOP_FONT_SUBSTITUTION_MESSAGES -&gt; yes</td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>Example:</strong> environment = (AOP_FONT_SUBSTITUTION_MESSAGES -&gt; no)</td>
<td></td>
</tr>
</tbody>
</table>
### C.2 Environment variables for PS and PDF to AFP

#### Table C-3  Environment variables for PDF and PS to AFP

<table>
<thead>
<tr>
<th>Environment Variables</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>_BPX_JOBNAME</code></td>
<td>The job name for this transform. When you assign a different job name to each class of transform, the operator can manage the transform daemons more effectively. Specify a job name of 1 to 8 alphanumeric characters. Incorrect job names are ignored.</td>
</tr>
<tr>
<td></td>
<td><strong>Default</strong>: The job name is AOPXF.</td>
</tr>
<tr>
<td></td>
<td><strong>Example</strong>: <code>environment = { _BPX_JOBNAME -&gt; PS2AFPD}</code></td>
</tr>
<tr>
<td><strong>AOP_RECLEN</strong></td>
<td>The length of the AFP images the transform creates. Each output record contains an AFP image and an additional 17 bytes of information (for example, header information). For example, if you specify a value of 8 (8,000 bytes) for the AFP image in this environment variable, the length of each output record is 8,017 bytes.</td>
</tr>
<tr>
<td></td>
<td>Valid values are from 8 to 32, in units of 1,000 bytes. For example, 8 = 8,000.</td>
</tr>
<tr>
<td></td>
<td><strong>Tip</strong>: The default value is suitable for most installations. However, a larger record length might improve performance.</td>
</tr>
<tr>
<td></td>
<td><strong>Default</strong>: AOP_RECLEN -&gt; 8</td>
</tr>
<tr>
<td></td>
<td><strong>Example</strong>: <code>environment = { AOP_RECLEN -&gt; 16}</code></td>
</tr>
</tbody>
</table>
| **AOP_RESOURCE_PATH**               | The directories that contains fonts and other transform resources. If the fonts and other resources are in more than one directory, list all directories separated by a colon. The directories are searched in the order that they are listed. Do not list the default resource directories because the transform always searches these directories last. |}

**Default**: AOP_RESOURCE_PATH ->
/usr/lpp/Printsrv/ps2afpv2/lib:/usr/lpp/Printsrv/ps2afpv2/Resource:
/usr/lpp/Printsrv/ps2afpv2/fonts

| **AOPTRACEDIR**                     | The full path name of the directory where the transform writes trace information. You can specify the same directory for different transform classes. The name of the trace file identifies the transform and transform class, and contains a timestamp. |
|                                     | **Default**: AOPTRACEDIR -> /var/Printsrv/trace                                                                                                                             |
|                                     | **Examples**:                                                                                                                                                       |
|                                     |    `environment = {AOPTRACEDIR -> /var/Printsrv/xfd}`                                                                                                               |
|                                     |    `environment = {AOPTRACEDIR -> .}`                                                                                                                                    |

| **AOPTRACEON**                      | Turns tracing on. The transform traces all transform requests that use this transform class. Any value turns tracing on. To turn tracing off, do not specify this environment variable. Specify this environment variable only if IBM directs you to do so. Tracing can adversely affect performance. |
|                                     | **Default**: Tracing is turned off.                                                                                                                                       |
|                                     | **Example**: `environment = {AOPTRACEON -> 1}`                                                                                                                               |
|                                     | **Tip**: In addition to the environment variables listed in this section, you can specify other environment variables. For example, you can specify the `_CEE_RUNOPTS` environment variable. |
Table C-4  Environment variables added in z/OS V1R8

<table>
<thead>
<tr>
<th>Environment Variables - z/OS V1R8</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>_CEE_DMPTARG</td>
<td>The directory where Language Environment (LE) writes a CEEDUMP. You can specify a period (.) to write the CEEDUMP to the transform's current working directory. IBM customer service expects to find CEEDUMPS in the transform's current working directory. The transform's current working directory is: base-directory/xfd/ps2afp.#.d. base-directory is the base directory specified in the Infoprint Server configuration file, aopd.conf. An example of the transform's current working directory is: /var/Printsrv/xfd/ps2afp.0.d If you specify a directory other than the transform's current working directory, make sure that the user ID that owns the ps2afpd file has write access to the directory. For example, if user NOBODY owns transform file ps2afpd, user NOBODY must have write access to the directory. <strong>Default:</strong> LE writes the dump to the directory specified in the TMPDIR environment variable. If the TMPDIR environment variable is not specified, LE writes the dump to the /tmp directory. <strong>Example:</strong> environment = (_CEE_DMPTARG -&gt; .) <strong>Note:</strong> For other transforms, such as the PCL to AFP transform, LE writes the CEEDUMP to the transform's current working directory by default. This is because the owners of the other transforms have write access to the Infoprint Server base directory. Therefore, you do not need to specify the _CEE_DMPTARG environment variable for other transforms. For more information about how LE determines where to write the CEEDUMP, see z/OS Language Environment Debugging Guide, GA22-7560.</td>
</tr>
</tbody>
</table>
| AOP_FONT_SUBSTITUTION_MESSAGES     | Indicates whether the transform writes a message (AOP2500W) in the transform's stderr file when it substitutes fonts in a document if no other errors occurred. The transform substitutes a similar font when it cannot find a font that the document requests. If a document is transformed with no font substitution, the transform does not write a message. Font-substitution messages let you determine if the transform has used substitute fonts. If a substitute font is not acceptable, you might be able to add the original font to the transform. **Valid values are:**  
  - yes  The transform writes messages about font substitution.  
  - no  The transform does not write messages about font substitution.  
  **Default:** AOP_FONT_SUBSTITUTION_MESSAGES -> yes  
  **Example:** AOP_FONT_SUBSTITUTION_MESSAGES -> no |
C.3 Environment variables for AFP to PCL

Table C-5  Transform environment variables for AFP to PCL

<table>
<thead>
<tr>
<th>Environment Variables</th>
<th>Description</th>
</tr>
</thead>
</table>
| AOP_FAIL_ON_ERROR     | Specifies whether the transform stops processing when an error occurs during the transform.  
Valid values are:  
yes  The transform stops processing when any error occurs and does not create an output document. The return code from the transform is >0. The transform stderr file contains message AOP2501E and the transform error messages.  
no  The transform continues processing when certain types of errors, such as data stream errors, occur. If possible, the transform creates an output document that contains the transform error messages. The return code from the transform is 0.  
Default: AOP_FAIL_ON_ERROR -> no  
Example: environment = {AOP_FAIL_ON_ERROR -> yes} |
| _BPX_JOBNAME          | The job name for this transform. When you assign a different job name to each class of transform, the operator can manage the transform daemons more effectively. Specify a job name of 1 to 8 alphanumeric characters. Incorrect job names are ignored.  
Default: The job name is AOPXFD.  
Example: environment =[_BPX_JOBNAME-> AFP2PCLD] |

<table>
<thead>
<tr>
<th>Environment Variables</th>
<th>Description</th>
</tr>
</thead>
</table>
| _BPX_JOBNAME          | The job name for this transform. When you assign a different job name to each class of transform, the operator can manage the transform daemons more effectively. Specify a job name of 1 to 8 alphanumeric characters. Incorrect job names are ignored.  
Default: The job name is AOPXFD.  
Example: environment =[_BPX_JOBNAME-> AFP2PCLD] |
### AOP_CHARS

The default coded font. The transform uses this font to format error messages unless the page definition specified in the AOP_MSGPAGEDEDEF variable names a font. The transforms also use this font for (1) line data when no other font is specified in the page definition used to print the document and (2) AFP data when no other font is specified in the AFP data stream.

Specify the 1 to 4 character coded font name. You can specify the X0 (raster) or XZ (outline) prefix of the coded font name. If you do not specify a prefix, the transform adds an X0 prefix. Some coded fonts have 6-character names, not counting the X0 or XZ prefix. For these fonts, use the 4-character alternate coded font name. For font names and alternate font names, see IBM AFP Fonts: Font Summary for AFP Font Collection.

**Attention:**
- Specify only one default coded font in this environment variable.
- Specify a raster font because the AFP to PCL transform requires raster fonts.
- The font you specify, or default font X060D9, must exist in one of the AFP font libraries so that the transform can write error messages in the output.
- The code page associated with this coded font must be an EBCDIC code page.

**Note:** You might want to specify the same font as the resident font in the AFP printer to which output is usually printed.

**Default:** AOP_CHARS-> 60d9 (This is font X060D9 because the transform adds prefix X0.)

### AOP_COLOR

Indicates whether the transform is to produce color output. Specify yes if the printer supports color. Valid values are:

- **yes** The transform produces color output.
- **no** The transform does color simulation.

**Default:** AOP_COLOR-> no

### AOP_CUTSHEET

Indicates whether the transform is to prepare the output for printing on a cut-sheet printer. Valid values are:

- **yes** The output is to be printed on a cut-sheet printer. Therefore, the transform uses the cut-sheet specification in the form definition to determine whether to send medium orientation information to the printer. For more information about the cut-sheet specification in the form definition, see the description of the CUTSHEET command in IBM Page Printer Formatting Aid: User's Guide. Select this option if your output is incorrectly printing in the down direction on a cut-sheet printer.
- **no** The output is not to be printed on a cut-sheet printer. Therefore, the transform always sends medium orientation information to the printer.

**Default:** AOP_CUTSHEET-> no
### Environment Variables Description

<table>
<thead>
<tr>
<th>Environment Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AOP_FONTLIB</td>
<td>The AFP system resource libraries that contain fonts. Specify 1 to 8 data set names. Separate each name with a space. Libraries are searched in the order listed. Specify the AFP 300-pel raster and outline font libraries used by your installation:</td>
</tr>
</tbody>
</table>
|                          | - Specify AFP 300-pel raster font libraries because the AFP to PCL transform requires raster fonts.  
|                          | - Specify AFP outline font libraries if data to be transformed refers to outline fonts.                                                                                                                                 |
|                          | **Default**: AOP_FONTLIB-> "sys1.font300 sys1.fontoln"                                                                                                                                                      |
|                          | **Examples**: environment={AOP_FONTLIB-> "sys1.font300 \nsys1.fontoln inst.font300"}                                                                                                                     |
|                          | **Note**: The back slash (\) in this example indicates that the text within the quotation marks continues on the next line.  
|                          | environment={AOP_FONTLIB-> "sys1.font300"}                                                                                                                                                               |
| AOP_FONTMAP              | Indicates whether the transform maps outline fonts to AFP raster fonts. Because the AFP to PCL transform requires raster fonts, you should specify AOP_FONTMAP-> yes (default). Valid values are: |
|                          | - yes  The transform maps fonts.  
|                          | - no   The transform does not map fonts.                                                                                                                                                                  |
|                          | **Default**: AOP_FONTMAP-> yes                                                                                                                                                                             |
| AOP_FORMDEF              | The default form definition used to format the input data stream. Specify the 1 to 8 character form definition name, with or without the F1 prefix. If you omit the F1 prefix, the transform adds it. The transform uses this form definition only if no other form definition is specified. |
|                          | **Default**: AOP_FORMDEF-> F1CP0110                                                                                                                                                                          |
| AOP_FORMDEFLIB           | The AFP system resource libraries that contain form definitions. Specify from 1 to 8 data set names. Separate each name with a space. Libraries are searched in the order listed.                                       |
|                          | **Default**: AOP_FORMDEFLIB-> "sys1.fdeflib"                                                                                                                                                               |
|                          | **Example**: environment = {AOP_FORMDEFLIB-> "sys1.fdeflib inst.fdeflib"}                                                                                                                                   |
| AOP_MSGFORMDEF           | The form definition used to format transform error messages. Specify the 1 to 8 character form definition name, with or without the F1 prefix. If you omit the F1 prefix, the transform adds it. This form definition must be located in one of the libraries specified in the AOP_FORMDEFLIB variable. |
|                          | **Default**: AOP_MSGFORMDEF-> F1CP0110                                                                                                                                                                        |
| AOP_MSGPAGEDEF           | The page definition used to format transform error messages. Specify the 1 to 8 character page definition name, with or without the P1 prefix. If you omit the P1 prefix, the transform adds it. This page definition must be located in one of the libraries specified in the AOP_PAGEDEFLIB variable. The transform formats messages for the first paper size defined in the AOP_PAPER variable.  
<p>|                          | <strong>Recommendation</strong>: Use page definition P1P08682 for letter size paper, and page definition P1Q09182 for A4 paper.                                                                                      |
|                          | <strong>Default</strong>: AOP_MSGPAGEDEF -&gt; P1P08682                                                                                                                                                                     |</p>
<table>
<thead>
<tr>
<th>Environment Variables</th>
<th>Description</th>
</tr>
</thead>
</table>
| AOP_OVERLAYLIB        | The AFP system resource libraries that contain overlays. Specify from 1 to 8 data set names. Separate each name with a space. Libraries are searched in the order listed.  
**Default**: AOP_OVERLAYLIB->"sys1.overlib" |
| AOP_PAGEDEF           | The default page definition used to format line data and create PCL output. Specify the 1 to 8 character page definition name, with or without the P1 prefix. If you omit the P1 prefix, the transform adds it. The transform uses this page definition only if no other page definition is specified.  
**Recommendation**: Use page definition P1P08682 for letter size paper, and page definition P1Q09182 for A4 paper.  
**Default**: AOP_PAGEDEF-> P1P08682 |
**Environment Variables**

<table>
<thead>
<tr>
<th>Environment Variables</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>AOP_PAPER</strong></td>
<td>The name of the paper that is typically installed in each AFP input tray. The transform formats the PCL output for the paper in the AFP input tray ID the document selects. The position (1 through 9) of each paper name represents the number of the AFP input tray. The 10th position represents any AFP input tray number greater than 9. You can specify from 1 to 10 paper names. Separate each name with a space. If you specify fewer than 10 paper names, the transform sends an error message and uses the default tray number. To avoid an error message, specify 10 tray IDs. For more information about the paper names you can specify and how to add custom paper names, see &quot;Adding paper sizes&quot; on page 116. Valid paper names and their sizes (width x height) are:</td>
</tr>
<tr>
<td>Paper Name</td>
<td>Size</td>
</tr>
<tr>
<td>a3</td>
<td>292.25 x 413.25 mm</td>
</tr>
<tr>
<td>a4</td>
<td>210 x 297 mm</td>
</tr>
<tr>
<td>a4ee</td>
<td>210 x 297 mm</td>
</tr>
<tr>
<td>a5</td>
<td>148.3 x 210 mm</td>
</tr>
<tr>
<td>b4</td>
<td>250 x 353 mm</td>
</tr>
<tr>
<td>b5</td>
<td>176 x 250 mm</td>
</tr>
<tr>
<td>c5</td>
<td>162 x 229 mm (6.48 x 9.16 in.)</td>
</tr>
<tr>
<td>com10</td>
<td>4.125 x 9.5 in. (104.8 x 241.3 mm)</td>
</tr>
<tr>
<td>dl</td>
<td>110 x 220 mm (4.4 x 8.8 in)</td>
</tr>
<tr>
<td>executive</td>
<td>7.25 x 10.5 in (185 x 267 mm)</td>
</tr>
<tr>
<td>ledger</td>
<td>11 x 17 in (279 x 432 mm)</td>
</tr>
<tr>
<td>legal</td>
<td>8.5 x 14.0 in (216 x 356 mm)</td>
</tr>
<tr>
<td>letter</td>
<td>8.5 x 11.0 in (216 x 279 mm)</td>
</tr>
<tr>
<td>letteree</td>
<td>8.5 x 11.0 in (216 x 279 mm)</td>
</tr>
<tr>
<td>monarch</td>
<td>3.875 x 7.5 in. (98.4 x 190.5 mm)</td>
</tr>
<tr>
<td><strong>Note:</strong> Use the a4ee and letteree paper names when the printer is configured to print edge-to-edge. However, if the printer does not support edge-to-edge printing, documents created for edge-to-edge printing have the outside 50 pels, approximately 4 millimeters, of output cut off. <strong>Default:</strong> AOP_PAPER-&gt;&quot;letter letter letter letter letter letter letter letter letter letter&quot; <strong>Example:</strong> environment = {AOP_PAPER-&gt; &quot;letter legal letteree letter letter letter letter letter&quot;} <strong>Note:</strong> The back slash () in this example indicates that the text within the quotation marks continues on the next line.</td>
<td></td>
</tr>
<tr>
<td><strong>AOP_PAGEDEFLIB</strong></td>
<td>The AFP system resource libraries that contain page definitions. Specify from 1 to 8 data set names. Separate each name with a space. Libraries are searched in the order listed. <strong>Default:</strong> AOP_PAGEDEFLIB-&gt;&quot;sys1.pdeflib&quot;</td>
</tr>
<tr>
<td><strong>AOP_PAGESEGLIB</strong></td>
<td>The AFP system resource libraries that contain page segments. Specify from 1 to 8 data set names. Separate each name with a space. <strong>Default:</strong> AOP_PAGESEGLIB-&gt;&quot;sys1.pseglib&quot;</td>
</tr>
</tbody>
</table>
## AOP_PJL

Indicates whether the printer accepts all PCL 5 commands, including PJL commands. All printers that support PCL 5 support PJL commands. Some printers that support only PCL 4 do not support PJL commands. Sometimes, a printer that does not support PJL commands prints a smiley face where a PJL command occurs in the data stream or prints the PJL command on the first page. Valid values are:

- **yes**: The printer accepts PJL commands.
- **no**: The printer does not accept PJL commands. Therefore, the AFP to PCL transform does not create PJL commands.

**Default**: AOP_PJL-> yes

**Note:** Printer Job Language (PJL) was provided as a method for switching printer languages at the job level, and for status readback between the printer and the host computer. PJL adds job level controls, such as printer language switching, job separation, environment, status readback, device attendance and file system commands.

Many printer vendors have extended PJL to include commands proprietary to their products. Not all PJL commands documented by HP are implemented in all HP or other vendor products.

<table>
<thead>
<tr>
<th>Environment Variables</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AOP_PJL</td>
<td>Indicates whether the printer accepts all PCL 5 commands, including PJL commands. All printers that support PCL 5 support PJL commands. Some printers that support only PCL 4 do not support PJL commands. Sometimes, a printer that does not support PJL commands prints a smiley face where a PJL command occurs in the data stream or prints the PJL command on the first page. Valid values are:</td>
</tr>
<tr>
<td></td>
<td><strong>yes</strong>: The printer accepts PJL commands.</td>
</tr>
<tr>
<td></td>
<td><strong>no</strong>: The printer does not accept PJL commands. Therefore, the AFP to PCL transform does not create PJL commands.</td>
</tr>
<tr>
<td></td>
<td><strong>Default</strong>: AOP_PJL-&gt; yes</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> Printer Job Language (PJL) was provided as a method for switching printer languages at the job level, and for status readback between the printer and the host computer. PJL adds job level controls, such as printer language switching, job separation, environment, status readback, device attendance and file system commands.</td>
</tr>
<tr>
<td></td>
<td>Many printer vendors have extended PJL to include commands proprietary to their products. Not all PJL commands documented by HP are implemented in all HP or other vendor products.</td>
</tr>
</tbody>
</table>

## AOP_POSITIONING_METHOD

Indicates how the transform is to position GOCA characters. GOCA is an object architecture used to represent pictures generated by computer. For information about GOCA character positioning, see Data Stream and Objects Architectures: Graphics Object Content Architecture for Advanced Function Presentation Reference, S544-5498.

Valid values are:

- **cell**: The transform scales the GOCA characters using the cell size in the GOCA data stream, with a default cell size of (1,1) graphic units. Characters are scaled using the maximum baseline extent in the y direction and the maximum character increment in the x direction. The printed output from the transform is similar to the output on older IBM AFP printers, such as the IBM 3812 printer.

  **Note:** This positioning method can cause undesirable output if the GOCA data stream does not specify a cell size and the transform uses the default cell size. In this case, specify either the SCALE or FONT method.

- **font**: Positions the characters like normal text using the font size in the GOCA data stream. Characters are not scaled, and the cell size in the GOCA data stream is ignored. The printed output will be similar to the output on most newer IBM AFP printers. This positioning method is likely to produce the most readable output.

- **scale**: Scales the GOCA characters using the cell size in the GOCA data stream, with a default cell size of (140,230) graphic units. Characters are scaled using 0.6 of the point size in both the x and y directions. The scaled characters are proportionately spaced and positioned 10% over the baseline. The printed output from the transform is similar to the output on IBM AFP printers with “GCS=CHAR SCALE” selected, such as the IBM Infoprint 40 printer.

**Default**: AOP_POSITIONING_METHOD-> cell
### Environment Variables Description

<table>
<thead>
<tr>
<th>Environment Variables</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AOP_TRAYID</td>
<td>A mapping of AFP input tray numbers to PCL tray IDs. The position (1-9) of each PCL tray ID corresponds to the AFP tray number. The 10th position corresponds to all AFP input tray numbers greater than 9. Specify 1 to 10 PCL tray IDs, separating each number with a space. Number 0 (zero) indicates that an input tray is not installed in the printer. If the input document requests an input tray that is not installed, the transform writes an error message in the output file and uses printer tray 1. PCL tray IDs usually mean: 0 - Print current page (paper source remains unchanged) 1 - Feed paper from main paper source 2 - Feed paper from manual input 3 - Feed envelope from manual input 4 - Feed paper from alternate paper source 5 - Feed from optional large paper source 6 - Feed envelope from envelope feeder * 7 - Autoselect 8 - Feed paper from Tray 1 (right side tray) 20 - 39 - High Capacity Input (HCI) Trays 2-21 * Must be used in conjunction with Paper Size. The AFP to PCL transform codes the PCL tray ID specified in AOP_TRAYID in the PCL Paper Source command. For more information about PCL tray IDs, see the description of the Paper Source command in Hewlett Packard's PCL documentation. Because the implementation of paper tray IDs can vary from printer to printer, also see the documentation for your printer. Notes: ▶ PCL tray IDs do not match the tray numbers embossed on the actual printer trays. ▶ If the usual PCL tray IDs do not work, specify different tray IDs in the AOP_TRAYID variable until the printer selects paper from the desired paper tray. Default: AOP_TRAYID=&quot;1 4 0 0 0 0 0 0 2&quot; Example: environment = {AOP_TRAYID-&gt; &quot;1 4 1 1 1 1 1 2&quot;}</td>
</tr>
</tbody>
</table>

### C.4 Environment variables for AFP to PDF

#### Transform environment variables for AFP to PDF

<table>
<thead>
<tr>
<th>Environment Variables</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>_BPX_JOBNAME</td>
<td>The job name for this transform. When you assign a different job name to each class of transform, the operator can manage the transform daemons more effectively. Specify a job name of 1 to 8 alphanumeric characters. Incorrect job names are ignored. Default: AOPXFDF. Example: environment = (_BPX_JOBNAME -&gt; AFP2PDF)</td>
</tr>
<tr>
<td>Environment Variables</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>AOP_ANNOTATIONS</strong></td>
<td>Indicates whether the transform is to produce annotations, explanatory notes, in the PDF document. The transform can create an annotation on each page. The annotation contains printing instructions such as “Normal duplex Front side”. For more information about annotations, see the online help provided with Adobe Acrobat. Valid values are: &lt;br&gt; <strong>yes</strong> The transform produces annotations. &lt;br&gt; <strong>no</strong> The transform does not produce annotations. &lt;br&gt; <strong>Default</strong>: AOP_ANNOTATIONS -&gt; yes</td>
</tr>
<tr>
<td><strong>AOP_CHARS</strong></td>
<td>The default coded font. The transform uses this font to format error messages unless the page definition specified in the AOP_MSGPAGEDEF variable names a font. The transforms also use this font for (1) line data when no other font is specified in the page definition used to format the document and (2) AFP data when no other font is specified in the AFP data stream. &lt;br&gt; Specify the 1 to 4 character coded font name. You can specify the X0 (raster) or XZ (outline) prefix of the coded font name. If you do not specify a prefix, the transform adds an X0 prefix. Some coded fonts have 6-character names, not counting the X0 or XZ prefix. For these fonts, use the 4-character alternate coded font name. For font names and alternate font names, see IBM AFP Fonts: Font Summary for AFP Font Collection. If the default font is a raster font (indicated by the X0 prefix), the transform maps it to an outline font if AOP_FONTMAP -&gt; yes is set. &lt;br&gt; <strong>Attention</strong>: Specify only one default coded font in this environment variable. The font you specify, or default font X060D9, must exist in one of the AFP font libraries so that the transform can write error messages in the output. The coded font member for a raster font, for example X060D9, must exist in an AFP font library specified to the transform, even if the transform maps raster to outline fonts. The code page associated with this coded font must be an EBCDIC code page. &lt;br&gt; <strong>Notes</strong>: You might want to specify the same font as the resident font in the AFP printer to which output is usually printed. Specify an outline font because outline fonts provide higher quality output for printing and viewing. &lt;br&gt; <strong>Default</strong>: AOP_CHARS -&gt; 60d9 (This is font X060D9 because the transform adds prefix X0.)</td>
</tr>
<tr>
<td><strong>AOP_CUTSHEET</strong></td>
<td>Indicates whether the transform is to prepare the output for printing on a cut-sheet printer. Valid values are: &lt;br&gt; <strong>yes</strong> The output is to be printed on a cut-sheet printer. Therefore, the transform uses the cut-sheet specification in the form definition to determine whether to send medium orientation information to the printer. For more information about the cut-sheet specification in the form definition, see the description of the CUTSHEET command in IBM Page Printer Formatting Aid: User's Guide. Select this option if your output is incorrectly printing in the down direction on a cut-sheet printer. &lt;br&gt; <strong>no</strong> The output is not to be printed on a cut-sheet printer. Therefore, the transform always sends medium orientation information to the printer. &lt;br&gt; <strong>Default</strong>: AOP_CUTSHEET -&gt; no &lt;br&gt; <strong>Example</strong>: environment = {AOP_CUTSHEET -&gt; no}</td>
</tr>
<tr>
<td>Environment Variables</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------</td>
<td>-------------</td>
</tr>
</tbody>
</table>
| AOP_ENCRYPT          | Indicates whether you want the transform to encrypt documents.  
|                       | **yes** The transform encrypts documents if (1) the transform request specifies a user or owner identifier or (2) the AOP_PROTECT environment variable is specified.  
|                       | **no** The transform does not encrypt documents. Transform requests that specify user or owner identifiers fail.  
|                       | **Attention:** Before setting AOP_ENCRYPT -> yes, you must customize the Open Cryptographic Services Facility (OCSF). If OCSF is not customized, transform requests fail. For more information, see “Customizing OCSF” on page 98.  
|                       | **Default:** AOP_ENCRYPT -> no |
| AOP_FLATE             | Indicates whether the transform is to use the Adobe Flate compression algorithm to compress the output.  
|                       | **Tip:** Flate (also called ZIP) is a compression method that works well on images with large areas of single colors or repeating patterns. Acrobat provides 4-bit and 8-bit Flate compression options. If you use 4-bit Flate compression with 4-bit images, or 8-bit Flate with 4-bit or 8-bit images, the Flate method is lossless, which means it does not remove data to reduce file size and so does not affect an image’s quality. If, however, you use 4-bit Flate compression with 8-bit data, you can affect the image quality since data is lost.  
|                       | Valid values are:  
|                       | **yes** The transform produces compressed output.  
|                       | **no** The transform does not produce compressed output. However, throughput is higher.  
|                       | **Default:** AOP_FLATE -> yes |
| AOP_FONTLIB           | The AFP system resource libraries that contain fonts. Specify 1 to 8 data set names. Separate each name with a space. Libraries are searched in the order listed. Specify the AFP 300-pel raster and outline font libraries used by your installation:  
|                       | ▶ Specify AFP 300-pel raster font libraries if either (1) AOP_CHARS specifies a raster font or (2) documents to be transformed reference raster fonts and AOP_FONTMAP -> no.  
|                       | ▶ Specify AFP outline font libraries if either (1) data to be transformed references outline fonts or (2) AOP_FONTMAP -> yes.  
|                       | **Default:** AOP_FONTLIB -> "sys1.font300 sys1.fontoln"  
|                       | **Examples:**  
|                       | `environment={AOP_FONTLIB -> "inst.font300 sys1.font300 \  
|                       | sys1.fontoln"}  
|                       | **Note:** The back slash (\) in this example indicates that the text within the quotation marks continues on the next line.  
|                       | `environment={AOP_FONTLIB -> "sys1.font300"}` |
### Appendix C. Transform environment variables

<table>
<thead>
<tr>
<th>Environment Variables</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AOP_FONTMAP</td>
<td>Indicates whether the transform maps AFP raster fonts to outline fonts. Although the AFP to PDF transform can use either raster or outline fonts, outline fonts provide higher quality output for printing and viewing. Valid values are:</td>
</tr>
<tr>
<td></td>
<td><strong>yes</strong>  The transform maps fonts.</td>
</tr>
<tr>
<td></td>
<td><strong>no</strong>   The transform does not map fonts.</td>
</tr>
<tr>
<td></td>
<td>Enable font-mapping because outline fonts provide higher quality output for viewing and printing.</td>
</tr>
<tr>
<td><strong>Default:</strong> AOP_FONTMAP -&gt; yes</td>
<td></td>
</tr>
<tr>
<td><strong>Example:</strong> environment = {AOP_FONTMAP -&gt; yes}</td>
<td></td>
</tr>
<tr>
<td>AOP_FORMDEF</td>
<td>The default form definition used to format the input data stream and create PDF output. Specify the 1 to 8 character form definition name, with or without the F1 prefix. If you omit the F1 prefix, the transform adds it. The transform uses this form definition only if no other form definition is specified.</td>
</tr>
<tr>
<td><strong>Default:</strong> AOP_FORMDEF -&gt; F1CP0110</td>
<td></td>
</tr>
<tr>
<td>AOP_FORMDEFLIB</td>
<td>The AFP system resource libraries that contain form definitions. Specify from 1 to 8 data set names. Separate each name with a space. Libraries are searched in the order listed.</td>
</tr>
<tr>
<td><strong>Default:</strong> AOP_FORMDEFLIB -&gt; &quot;sys1.fdeflib&quot;</td>
<td></td>
</tr>
<tr>
<td>AOP_INDEX</td>
<td>Indicates whether the transform is to create PDF bookmarks in the PDF document when the input AFP document contains Tag Logical Elements (TLE) structured fields. For example, a table of contents in an AFP document can be converted to a set of PDF bookmarks. Adobe Acrobat Reader displays PDF bookmarks in its navigation pane. For more information about bookmarks, see the online help provided with Adobe Acrobat. Valid values are:</td>
</tr>
<tr>
<td></td>
<td><strong>yes</strong>  The transform creates bookmarks in the PDF document.</td>
</tr>
<tr>
<td></td>
<td><strong>no</strong>   The transform does not create bookmarks in the PDF document.</td>
</tr>
<tr>
<td><strong>Default:</strong> AOP_INDEX -&gt; no</td>
<td></td>
</tr>
<tr>
<td>AOP_INDEX_LANG</td>
<td>The default code page the transform uses to translate text in the PDF bookmarks. This code page is used for line-data documents and when the AFP Begin Document structured field in an AFP document does not contain a document language. Valid values are:</td>
</tr>
<tr>
<td><strong>Default:</strong> AOP_INDEX_LANG -&gt; 037</td>
<td></td>
</tr>
<tr>
<td>Code page</td>
<td>Description</td>
</tr>
<tr>
<td>------------</td>
<td>---------------------------------------------------</td>
</tr>
<tr>
<td>037</td>
<td>USA, Canada, Australia, New Zealand</td>
</tr>
<tr>
<td>273</td>
<td>Austria, Germany</td>
</tr>
<tr>
<td>274</td>
<td>Belgium 275 Brazil</td>
</tr>
<tr>
<td>277</td>
<td>Denmark, Norway</td>
</tr>
<tr>
<td>278</td>
<td>Finland, Sweden</td>
</tr>
<tr>
<td>280</td>
<td>Italy</td>
</tr>
<tr>
<td>281</td>
<td>Japan Latin</td>
</tr>
<tr>
<td>282</td>
<td>Portugal</td>
</tr>
<tr>
<td>284</td>
<td>Spain, Latin America</td>
</tr>
<tr>
<td>285</td>
<td>UK</td>
</tr>
<tr>
<td>297</td>
<td>France</td>
</tr>
<tr>
<td>500</td>
<td>International</td>
</tr>
<tr>
<td>871</td>
<td>Iceland</td>
</tr>
</tbody>
</table>
**AOP_LINEARIZE**
Indicates whether the transform is to linearize PDF documents so that they can be viewed faster from the Web. This function is especially important for large PDF documents that can take a long time to download. In Adobe Acrobat online help, this function is called optimization and Fast Web View. Adobe Acrobat Reader displays the first page of a linearized PDF document before the entire document has been loaded from a Web site. For more information about optimization, see the online help provided with Adobe Acrobat. Valid values are:

- **yes**: The transform optimizes the PDF document for fast viewing from the Web.
- **no**: The transform does not optimize the PDF document for fast viewing from the Web.

When you select linearization, the transform uses additional memory because the entire PDF document is kept in memory while it is being linearized.

**Default**: AOP_LINEARIZE -> no

**AOP_LINKS**
Indicates whether the transform is to create links in the PDF document when the input AFP document contains Link Logical Elements (LLE) structured fields. If the Object Classification Triplet in the LLE indicates an executable program link, the transform creates a Uniform Resource Locator (URL) in the PDF document. Valid values are:

- **yes**: The transform creates links in the PDF document.
- **no**: The transform does not create links in the PDF document.

**Default**: AOP_LINKS -> no

**AOP_MSGFORMDEF**
The form definition used to format transform error messages. Specify the 1 to 8 character form definition name, with or without the F1 prefix. If you omit the F1 prefix, the transform adds it. This form definition must be located in one of the libraries specified in the AOP_FORMDEFLIB variable.

**Default**: AOP_MSGFORMDEF -> F1CP0110

**AOP_MSGPAGEDEF**
The page definition used to format transform error messages. Specify the 1 to 8 character page definition name, with or without the P1 prefix. If you omit the P1 prefix, the transform adds it. This page definition must be located in one of the libraries specified in the AOP_PAGEDEFLIB variable. The transform formats messages for the first paper size defined in the AOP_PAPER variable.

**Recommendation**: Use page definition P1P08682 for letter size paper, and page definition P1Q09182 for A4 paper.

**Default**: AOP_MSGPAGEDEF -> P1P08682
### Environment Variables Description

<table>
<thead>
<tr>
<th>Environment Variables</th>
<th>Description</th>
</tr>
</thead>
</table>
| **AOP_OUTLINES**      | Specifies the type of processing the transform does for outline fonts. Valid values are:  
  - **builtin**: The transform includes only the names of outline fonts in the PDF output. Outline fonts themselves are not included in the output. When the PDF output is viewed or printed, the PDF driver (for example, Adobe Acrobat or a PDF printer) maps the requested typeface to an equivalent typeface. This option produces smaller output files than when fonts are included in the output. However, differences in output appearance can result due to the mapping done by the PDF driver. For example, font spacing might be different, and if you use special characters or if you use characters or glyphs that are unknown to the PDF driver, these characters will not be visible.  
  - **yes**: The transform includes outline fonts in the PDF output. This option provides better output fidelity than the builtin option. However, each typeface increases the size of the PDF output file by approximately 110K.  
  Default: AOP_OUTLINES -> yes |
| **AOP_OVERLAYLIB**    | The AFP system resource libraries that contain overlays. Specify from 1 to 8 data set names. Separate each name with a space. Libraries are searched in the order listed.  
  Default: AOP_OVERLAYLIB -> "sys1.overlib" |
| **AOP_PAGEDEF**       | The default page definition used to format line data. Specify the 1 to 8 character page definition name, with or without the P1 prefix. If you omit the P1 prefix, the transform adds it. The transform uses this page definition only if no other page definition is specified.  
  **Recommendation**: Use page definition P1P08682 for letter size paper, and page definition P1Q09182 for A4 paper.  
  Default: AOP_PAGEDEF -> P1P08682 |
| **AOP_PAGEDEFLIB**    | The AFP system resource libraries that contain page definitions. Specify from 1 to 8 data set names. Separate each name with a space.  
  Default: AOP_PAGEDEFLIB -> "sys1.pdeflib" |
| **AOP_PAGESEGLIB**    | The AFP system resource libraries that contain page segments. Specify from 1 to 8 data set names. Separate each name with a space.  
  Default: AOP_PAGESEGLIB -> "sys1.pseglib" |
The name of the paper that is typically installed in each AFP input tray. The transform formats the PDF output for the paper in the first AFP input tray ID the document selects.

The position (1 through 9) of each paper name represents the number of the AFP input tray. The 10th position represents any AFP input tray number greater than 9.

You can specify from 1 to 10 paper names. Separate each name with a space. If you specify fewer than 10 paper names, the transform sends an error message and uses the default tray number. To avoid an error message, specify 10 tray IDs.

For more information about the paper names you can specify and how to add custom paper names, see “Adding paper sizes” on page 116. Valid paper names and their sizes (width x height) are:

<table>
<thead>
<tr>
<th>Paper Name</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>a3</td>
<td>292.25 x 413.25 mm</td>
</tr>
<tr>
<td>a4</td>
<td>210 x 297 mm a4ee 210 x 297 mm</td>
</tr>
<tr>
<td>a5</td>
<td>148.3 x 210 mm</td>
</tr>
<tr>
<td>b4</td>
<td>250 x 353 mm</td>
</tr>
<tr>
<td>b5</td>
<td>176 x 250 mm</td>
</tr>
<tr>
<td>c5</td>
<td>162 x 229 mm (6.48 x 9.16 in.)</td>
</tr>
<tr>
<td>com10</td>
<td>4.125 x 9.5 in. (104.8 x 241.3 mm)</td>
</tr>
<tr>
<td>dl</td>
<td>110 x 220 mm (4.4 x 8.8 in)</td>
</tr>
<tr>
<td>executive</td>
<td>7.25 x 10.5 in (185 x 267 mm)</td>
</tr>
<tr>
<td>ledger</td>
<td>11 x 17 in (279 x 432 mm)</td>
</tr>
<tr>
<td>legal</td>
<td>8.5 x 14.0 in (216 x 356 mm)</td>
</tr>
<tr>
<td>letter</td>
<td>8.5 x 11.0 in (216 x 279 mm)</td>
</tr>
<tr>
<td>letteree</td>
<td>8.5 x 11.0 in (216 x 279 mm)</td>
</tr>
<tr>
<td>monarch</td>
<td>3.875 x 7.5 in. (98.4 x 190.5 mm)</td>
</tr>
</tbody>
</table>

Note: Use the a4ee and letteree paper names when the printer is configured to print edge-to-edge. However, if the printer does not support edge-to-edge printing, documents created for edge-to-edge printing have the outside 50 pels, approximately 4 millimeters, of output cut off.

Default: AOP_PAPER -> "letter letter letter letter letter letter letter letter letter"

Example: environment = (AOP_PAPER -> "letter legal letteree letter letter letter letter"

Note: The back slash (\) in this example indicates that the text within the quotation marks continues on the next line.
### Environment Variables Description

<table>
<thead>
<tr>
<th>Environment Variables</th>
<th>Description</th>
</tr>
</thead>
</table>
| **AOP_PASSWORD_EXIT**  | The name of the Password exit and optional arguments. The transform calls the Password exit to obtain PDF owner and user passwords for encryption when a user identifier or owner identifier is specified. Specify the full path name of the Password exit unless it is in a directory named in the LIBPATH environment variable. Also, specify any optional arguments that your Password exit accepts. A sample Password exit is provided in /usr/lpp/Printsrv/lib/aoppdfexit.dll. This sample exit accepts the name of the password database as an optional argument (if none is specified, the sample exit uses /etc/Printsrv/aoppdfexit.db). For information about the sample Password exit, see "Writing a Password exit" on page 98. **Default:** AOP_PASSWORD_EXIT -> “/usr/lpp/Printsrv/lib/aoppdfexit.dll” **Example:** In this example, the argument passed to the exit is the name of the sample password database that the transform provides. environment = {AOP_PASSWORD_EXIT -> 
*/usr/lpp/Printsrv/lib/aoppdfexit.dll 
/usr/lpp/Printsrv/samples/aoppdfexit.db“} |
| **AOP_PROTECT**        | The actions the transform is to restrict in all PDF documents when the transform job does not specify either a user or owner identifier. The transform encrypts the PDF documents and restricts the specified actions. However, the encrypted PDF documents do not have user passwords. This means that anyone can read them without a password. The transform uses a low level encryption (a 40-bit encryption key). If a transform job specifies a user or owner identifier, the transform ignores this variable and encrypts the PDF document using the passwords associated with the identifiers. It restricts the actions specified in the pdf-protect job attribute or in the printer definition. Valid values for actions are: **Value:** Actions users cannot do:  
select  
- Copy or extract content to another document  
- Extract content for accessibility  
print  
- Print at low resolution (150 dpi)  
- Print at high resolution  
modify  
- Change the document  
- Assemble the document (insert, delete, rotate pages)  
- Create template pages  
**Default:** None  
**Example:** environment = {AOP_PROTECT -> “select print modify”} |
### AOP_POSITIONING_METHOD
Indicates how the transform is to position GOCA characters. For information about GOCA character positioning, see Data Stream and Objects Architectures: Graphics Object Content Architecture for Advanced Function Presentation Reference, S544-5498. Valid values are:

- **cell**: The transform scales the GOCA characters using the cell size in the GOCA data stream, with a default cell size of (1,1) graphic units. Characters are scaled using the maximum baseline extent in the y direction and the maximum character increment in the x direction. The printed output from the transform is similar to the output on older IBM AFP printers, such as the IBM 3812 printer.
  
  **Note**: This positioning method can cause undesirable output if the GOCA data stream does not specify a cell size and the transform uses the default cell size. In this case, specify either the SCALE or FONT method.

- **font**: Positions the characters like normal text using the font size in the GOCA data stream. Characters are not scaled, and the cell size in the GOCA data stream is ignored. The printed output will be similar to the output on most newer IBM AFP printers. This positioning method is likely to produce the most readable output.

- **scale**: Scales the GOCA characters using the cell size in the GOCA data stream, with a default cell size of (140,230) graphic units. Characters are scaled using 0.6 of the point size in both the x and y directions. The scaled characters are proportionately spaced and positioned 10% over the baseline. The printed output from the transform is similar to the output on IBM AFP printers with “GCS=CHAR SCALE” selected, such as the IBM Infoprint 40 printer.

**Default**: `AOP_POSITIONING_METHOD -> cell`

**Example**: `environment = {AOP_POSITIONING_METHOD -> font}`

### AOP_TRAYID
Indicates whether an AFP tray number is valid. The position (1 - 9) of each value corresponds to the AFP tray number. The 10th position corresponds to all AFP input tray numbers greater than 9. Valid values are:

- **1**: The AFP tray number is valid.
- **0**: The AFP tray number is not valid. If the AFP document requests this tray, the transform writes an error message.

To avoid transform error messages, either omit this variable or specify the default value.

**Default**: `AOP_TRAYID -> "1 1 1 1 1 1 1 1 1 1"`

### AOP_ROTATE_PDF
Indicates how the transform is to rotate PDF documents. For example, some pages (such as those that contain tables) might require the PDF document to be turned sideways for reading. Valid values are:

- **no**: The rotation in the form definition is used (PRESENT keyword).
- **auto**: The rotation on each page is the same as the direction of the first character on the page.
- **0**: The PDF document is not rotated.
- **90**: The entire PDF document is rotated 90 degrees.
- **180**: The entire PDF document is rotated 180 degrees.
- **270**: The entire PDF document is rotated 270 degrees.

Rotating PDF documents can change the way the PDF documents print. If PDF documents do not print correctly using the **no** value, try the **0** value.

**Default**: `AOP_ROTATE_PDF -> no`
## C.5 Environment variables for AFP to PS

### Table C-7  Transform environment variables for AFP to PS

<table>
<thead>
<tr>
<th>Environment Variables - APF-PS</th>
<th>Description</th>
</tr>
</thead>
</table>
| **_BPX_JOBNAME**              | The job name for this transform. When you assign a different job name to each class of transform, the operator can manage the transform daemons more effectively. Specify a job name of 1 to 8 alphanumeric characters. Incorrect job names are ignored.  
**Default:** The job name is AOPXFD.  
**Example:** `environment = {_BPX_JOBNAME –> AFP2PSD}` |
|                               |             |
| **AOP_CHARS**                 | The default coded font. The transforms use this font to format error messages unless the page definition specified in the AOP_MSGPAGEDEF variable names a font. The transforms also use this font for (1) line data when no other font is specified in the page definition used to print the document and (2) AFP data when no other font is specified in the AFP data stream. If this environment variable is not specified, the default font is X060D9. The default font you specify, or font X060D9, must exist in one of the AFP font libraries so that the transform can write error messages in the output.  
Specify the 1 to 4 character coded font name. You can specify the X0 (raster) or XZ (outline) prefix of the coded font name. If you do not specify a prefix, the transform adds an X0 prefix. Some coded fonts have 6-character names, not counting the X0 or XZ prefix. For these fonts, use the 4-character alternate coded font name. For font names and alternate font names, see *IBM AFP Fonts: Font Summary for AFP Font Collection*.  
If the default font is a raster font (indicated by the X0 prefix), the transform maps it to an outline font if AOP_FONTMAP –> yes is set.  
**Attention:** Specify only one default coded font in this environment variable. The font you specify, or default font X060D9, must exist in one of the AFP font libraries so that the transform can write error messages in the output. The coded font member for a raster font, for example X060D9, must exist in an AFP font library specified to the transform, even if the transform maps raster to outline fonts. The code page associated with this coded font must be an EBCDIC code page.  
- You might want to specify the same font as the resident font in the AFP printer to which output is usually printed.  
- Specify an outline font because outline fonts provide higher quality output for printing and viewing.  
**Default:** AOP_CHARS –> 60d9 (This is font X060D9 because the transform adds prefix X0.) |
| **AOP_COLOR**                 | Indicates whether the transform is to produce color output. Specify yes if the printer supports color. Valid values are:  
**yes**  The transform produces color output.  
**no**  The transform does color simulation.  
**Default:** AOP_COLOR –> no |
<table>
<thead>
<tr>
<th>Environment Variables - APF-PS</th>
<th>Description</th>
</tr>
</thead>
</table>
| AOP_CUTSHEET                  | Indicates whether the transform is to prepare the output for printing on a cut-sheet printer. Valid values are:  
  yes The output is to be printed on a cut-sheet printer. Therefore, the transform uses the cut-sheet specification in the form definition to determine whether to send medium orientation information to the printer. For more information about the cut-sheet specification in the form definition, see the description of the CUTSHEET command in IBM Page Printer Formatting Aid: User’s Guide. Select this option if your output is incorrectly printing in the down direction on a cut-sheet printer.  
  no The output is not to be printed on a cut-sheet printer. Therefore, the transform always sends medium orientation information to the printer.  
  Default: AOP_CUTSHEET -> no |
| AOP_FONTLIB                   | The AFP system resource libraries that contain fonts. Specify 1 to 8 data set names. Separate each name with a space. Libraries are searched in the order listed.  
  Specify the AFP 300-pel raster and outline font libraries used by your installation:  
  - Specify AFP 300-pel raster font libraries if either (1) you specify a raster font in the AOP_CHARS environment variable or (2) documents to be transformed reference raster fonts and you specify AOP_FONTMAP -> no.  
  - Specify AFP outline font libraries if either (1) data to be transformed references outline fonts or (2) you specify AOP_FONTMAP -> yes.  
  Default: AOP_FONTLIB -> "sys1.font300 sys1.fontln"  
  Example: environment={AOP_FONTLIB -> "inst.font300 sys1.font300 \ sys1.fontln"}  
  Note: The back slash (\) in this example indicates that the text within the quotation marks continues on the next line.  
  Example: environment={AOP_FONTLIB -> "sys1.font300"} |
| AOP_FONTMAP                   | Indicates whether the transform maps AFP raster fonts to outline fonts. Valid values are:  
  yes The transform enables font-mapping. You should enable font-mapping because outline fonts provide superior viewing and printing.  
  no The transform does not map fonts. If your installation has not installed outline font libraries, you can disable font-mapping to avoid error messages. If the transform does not find an outline font in the system font libraries, the transform automatically disables font-mapping for the raster font. However, the transform writes an error message in the transform error log once for each missing outline font.  
  Default: AOP_FONTMAP -> yes |
| AOP_FORMDEF                   | The default form definition used to format the input data stream. Specify the 1 to 8 character form definition name, with or without the F1 prefix. If you omit the F1 prefix, the transform adds it. The transform uses this form definition only if no other form definition is specified.  
  Default: AOP_FORMDEF -> F1CP0110 |
| AOP_FORMDEFLIB                | The AFP system resource libraries that contain form definitions. Specify from 1 to 8 data set names. Separate each name with a space. Libraries are searched in the order listed.  
  Default: AOP_FORMDEFLIB -> "sys1.fdeflib" |
### Environment Variables - APF-PS

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>AOP_MSGFORMDEF</td>
<td>The form definition used to format transform error messages. Specify the 1 to 8 character form definition name, with or without the F1 prefix. If you omit the F1 prefix, the transform adds it. This form definition must be located in one of the libraries specified in the AOP_FORMDEFLIB variable.</td>
<td>AOP_MSGFORMDEF -&gt; F1CP0110</td>
</tr>
<tr>
<td>AOP_MSGPAGEDEF</td>
<td>The page definition used to format transform error messages. Specify the 1 to 8 character page definition name, with or without the P1 prefix. If you omit the P1 prefix, the transform adds it. This page definition must be located in one of the libraries specified in the AOP_PAGEDEFLIB variable. The transform formats messages for the first paper size defined in the AOP_PAPER variable.</td>
<td>AOP_MSGPAGEDEF -&gt; P1P08682</td>
</tr>
<tr>
<td>AOP_OVERLAYLIB</td>
<td>The AFP system resource libraries that contain overlays. Specify from 1 to 8 data set names. Separate each name with a space. Libraries are searched in the order listed.</td>
<td>AOP_OVERLAYLIB -&gt; &quot;sys1.overlib&quot;</td>
</tr>
<tr>
<td>AOP_PAGEDEF</td>
<td>The default page definition used to format line data. Specify the 1 to 8 character page definition name, with or without the P1 prefix. If you omit the P1 prefix, the transform adds it. The transform uses this page definition only if no other page definition is specified.</td>
<td>AOP_PAGEDEF -&gt; P1P08682</td>
</tr>
<tr>
<td>AOP_PAGEDEFLIB</td>
<td>The AFP system resource libraries that contain page definitions. Specify from 1 to 8 data set names. Separate each name with a space. Libraries are searched in the order listed.</td>
<td>AOP_PAGEDEFLIB -&gt; &quot;sys1.pdeflib&quot;</td>
</tr>
<tr>
<td>AOP_PAGESEGLIB</td>
<td>The AFP system resource libraries that contain page segments. Specify from 1 to 8 data set names. Separate each name with a space.</td>
<td>AOP_PAGESEGLIB -&gt; &quot;sys1.pseglib&quot;</td>
</tr>
</tbody>
</table>
The name of the paper that is typically installed in each AFP input tray. The transform formats the PostScript output for the paper in the AFP input tray ID the document selects.

The position (1 through 9) of each paper name represents the number of the AFP input tray. The 10th position represents any AFP input tray number greater than 9. You can specify from 1 to 10 paper names. Separate each name with a space. If you specify fewer than 10 paper names, the transform sends an error message and uses the default tray number. For more information about the paper names you can specify and how to add custom paper names, see "Adding paper sizes" on page 116.

Valid paper names and their sizes (width x height) are:

<table>
<thead>
<tr>
<th>Paper Name</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>a3</td>
<td>292.25 x 413.25 mm</td>
</tr>
<tr>
<td>a4</td>
<td>210 x 297 mm</td>
</tr>
<tr>
<td>a4ee</td>
<td>210 x 297 mm</td>
</tr>
<tr>
<td>a5</td>
<td>148.3 x 210 mm</td>
</tr>
<tr>
<td>b4</td>
<td>250 x 353 mm</td>
</tr>
<tr>
<td>b5</td>
<td>176 x 250 mm</td>
</tr>
<tr>
<td>c5</td>
<td>162 x 229 mm (6.48 x 9.16 in.)</td>
</tr>
<tr>
<td>com10</td>
<td>4.125 x 9.5 in. (104.8 x 241.3 mm)</td>
</tr>
<tr>
<td>dl</td>
<td>110 x 220 mm (4.4 x 8.8 in)</td>
</tr>
<tr>
<td>executive</td>
<td>7.25 x 10.5 in (185 x 267 mm)</td>
</tr>
<tr>
<td>ledger</td>
<td>11 x 17 in (279 x 432 mm)</td>
</tr>
<tr>
<td>legal</td>
<td>8.5 x 14.0 in (216 x 356 mm)</td>
</tr>
<tr>
<td>letter</td>
<td>8.5 x 11.0 in (216 x 279 mm)</td>
</tr>
<tr>
<td>letteree</td>
<td>8.5 x 11.0 in (216 x 279 mm)</td>
</tr>
<tr>
<td>monarch</td>
<td>3.875 x 7.5 in. (98.4 x 190.5 mm)</td>
</tr>
</tbody>
</table>

**Note:** Use the a4ee and letteree paper names when the printer is configured to print edge-to-edge. However, if the printer does not support edge-to-edge printing, documents created for edge-to-edge printing have the outside 50 pels, approximately 4 millimeters, of output cut off.

**Default:** AOP_PAPER -> "letter letter letter letter letter letter letter letter letter letter"

**Example:** environment = {AOP_PAPER -> "letter legal letteree letter \letter letter letter letter letter"}

**Note:** The back slash (\) in this example indicates that the text within the quotation marks continues on the next line.
### Environment Variables - APF-PS

<table>
<thead>
<tr>
<th>Environment Variables - APF-PS</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>AOP_POSITIONING_METHOD</strong></td>
<td>Indicates how the transform is to position GOCA characters. For information about GOCA character positioning, see <em>Data Stream and Objects Architectures: Graphics Object Content Architecture for Advanced Function Presentation Reference, S544-5498.</em> Valid values are:</td>
</tr>
<tr>
<td>cell</td>
<td>The transform scales the GOCA characters using the cell size in the GOCA data stream, with a default cell size of (1,1) graphic units. Characters are scaled using the maximum baseline extent in the y direction and the maximum character increment in the x direction. The printed output from the transform is similar to the output on older IBM AFP printers, such as the IBM 3812 printer. <strong>Note:</strong> This positioning method can cause undesirable output if the GOCA data stream does not specify a cell size and the transform uses the default cell size. In this case, specify either the SCALE or FONT method.</td>
</tr>
<tr>
<td>font</td>
<td>Positions the characters like normal text using the font size in the GOCA data stream. Characters are not scaled, and the cell size in the GOCA data stream is ignored. The printed output will be similar to the output on most newer IBM AFP printers. This positioning method is likely to produce the most readable output.</td>
</tr>
<tr>
<td>scale</td>
<td>Scales the GOCA characters using the cell size in the GOCA data stream, with a default cell size of (140,230) graphic units. Characters are scaled using 0.6 of the point size in both the x and y directions. The scaled characters are proportionately spaced and positioned 10% over the baseline. The printed output from the transform is similar to the output on IBM AFP printers with “GCS=CHAR SCALE” selected, such as the IBM Infoprint 40 printer.</td>
</tr>
<tr>
<td><strong>Default:</strong> AOP_POSITIONING_METHOD -&gt; cell</td>
<td></td>
</tr>
</tbody>
</table>

| **AOP_TRAYID** | A mapping of AFP input tray numbers to PostScript tray IDs. The position (1 - 9) of each PostScript tray ID corresponds to the AFP tray number. The 10th position corresponds to all AFP input tray numbers greater than 9. Specify 1 to 10 PostScript tray numbers, separating each number with a space. Number 0 (zero) indicates that an input tray is not installed in the printer. If the input document requests an input tray that is not installed, the transform writes an error message in the output file and uses printer tray 1. The 10th position is reserved for the manual input tray. That is, if the AFP document selects tray M (manual) or any tray greater than 9, the transform maps the tray to the manual input tray on the PostScript printer regardless of the PostScript tray number you specify in the 10th position. Specify the ID used by the PostScript printer to select each tray. This value, minus 1, corresponds to an entry in the Priority array in the InputAttributes dictionary for the PostScript printer. Printer-specific values are described in the PostScript printer description (PPD) file for the printer. **Default:** AOP_TRAYID -> "1 2 0 0 0 0 0 0 2" **Example:** environment = {AOP_TRAYID -> "1 4 1 1 1 1 1 1 1"} |

---

Appendix C. Transform environment variables 227
C.6 Search hierarchies for forms and page definitions

During the execution of the transform, it is necessary to determine where some of the resources to be used exist. Since the same resource can be specified in many different specifications, a search is done in a specific order to determine where the resource to be used is. This search is done for the following resources:

- Form definitions
- Page definitions
- Fonts to be used
- Message fonts

Search hierarchy for form definitions

The search begins by checking from options 5 down to 1, with option 1 being the most preferred option.

1. The form definition specified in the form-definition job attribute or FORMDEF parameter on a OUTPUT JCL statement.
2. The form definition specified in the Form definition field of the Resource Related Values in the printer definition on the IP PrintWay Printer Definition Allocation panel.
3. The first inline form definition.
4. The form definition specified in the AOP_FORMDEF environment variable in the transform configuration file.
5. Form definition F1CP0111. This default is coded in the transform.

Important:

1. If form definition name dummy is specified, the transform uses the first inline form definition.
2. After the transform determines the name of the form definition, the transform first searches for the form definition inline in the data set, and then searches in the user and system resource libraries.

Search hierarchy for page definitions

The search begins by checking from options 6 down to 1, with option 1 being the most preferred option.

1. The page definition specified in the page-definition job attribute or the PAGEDEF or FCB JCL parameter. (The PAGEDEF parameter overrides the FCB parameter.)
2. The default page definition supplied by JES to IP PrintWay basic mode. (See Note.)
3. The page definition specified in the Page definition field in the printer definition.
4. The first inline page definition.
5. The page definition specified in the AOP_PAGEDEF environment variable in the transform configuration file.
6. Page definition P1P08682. This default is coded in the transform.
Appendix C. Transform environment variables

Important:
1. If page definition name dummy is specified, the transform uses the first inline page definition.
2. After the transform determines the name of the page definition, the transform first searches for the page definition inline in the data set, and then searches in the user and system resource libraries.
3. JES supplies a default page definition for batch jobs submitted to IP PrintWay basic mode with OUTPUT and DD JCL statements. If you want to use the default page definition specified in the transform configuration file, you can request that JES not supply a default page definition to the IP PrintWay basic mode FSA.

Search hierarchy for the default font
The search begins by checking from options 6 down to 1, with option 1 being the most preferred option.
1. The font named in the page definition.
2. The font specified in the chars job attribute or the CHARS or UCS JCL parameter. (The CHARS JCL parameter overrides the UCS parameter.)
3. The default font supplied by JES to IP PrintWay basic mode. (See Note.)
4. The font specified in the Character set field in the printer definition.
5. The font specified in the AOP_CHARS environment variable in the transform configuration file. The transform prefixes X0 to the font named in the variable if you do not specify a prefix.

Note: JES supplies a default font for jobs submitted to IP PrintWay basic mode with OUTPUT and DD JCL statements. If you want to use the default font specified in the transform configuration file, you can request that JES not supply a default font to the IP PrintWay basic mode FSA.

If you want the transforms to use a font and page definition specified in a printer definition or in the transform configuration file, do this:

- **JES2:**
  - To prevent JES2 from supplying a default page definition name, (1) do not specify the NIFCB parameter in the PRINTDEF statement and (2) do not specify the FCB parameter in the PRTnnnnnn statement.
  - To prevent JES2 from supplying a default font, specify UCS=0 in the PRTnnnnnn statement. When UCS=0, JES2 ignores any font specified in the NIUCS parameter.

- **JES3:**
  - To prevent JES3 from supplying a default page definition name, specify PDEFAULT=FCB on the DEVICE statement.
  - To prevent JES3 from supplying a default font specify PDEFAULT=CHARS on the DEVICE statement.

To prevent JES3 from supplying a default font or FCB, specify PDEFAULT=(CHARS,FCB) on the DEVICE statement.

For more information about the JES initialization parameters to do this, see z/OS Infoprint Server Customization.
**Search hierarchy for the message font**
The search begins by checking from options 3 down to 1, with option 1 being the most preferred option.

1. The font named in the page definition specified in the AOP_MSGPAGEDEF environment variable.
2. The font specified in the AOP_CHARS environment variable in the transform configuration file. The transform prefixes X0 to the font named in the variable if you do not specify a prefix.
3. Font X060D9 is the default to be coded in the transform.

**C.6.1 Search hierarchy for the AFP resource libraries**
The search begins by checking from options 4 down to 1, with option 1 being the most preferred option.

1. User resource libraries specified in the resource-library job attribute or USERLIB parameter on the OUTPUT JCL statement.
2. Default user resource libraries specified in the Resource library field of the Resource Related Values in the printer definition on the IP PrintWay Printer Definition Allocation panel. These libraries are searched only when the job submitter does not specify any user resource libraries (see step 1).
4. Default system resource libraries that are hard-coded in the transform. These libraries are searched only if no system resource libraries are specified in the transform configuration file (see step 3).

**C.7 AFP architecture limitations for transforms**
There are some items in the AFP architecture that the following transforms do not currently support. However, future enhancements may add new functional capabilities. IBM suggests that you test how your AFP applications print on your non-AFP printers to make sure the output is acceptable.

**C.7.1 PCL to AFP limitations**
These limitations apply to the PCL to AFP transform:

- The transform does not produce color output. It produces monochrome output only.
- DBCS fonts are not built in to the transform, but the transform can process DBCS fonts that are embedded in the PCL document.
- The transform cannot create IOCA Function Set 45 (FS45) image objects, bilevel or color tiled images, which some IBM color printers require.
- PCL data can contain device commands (for example, to begin or end duplexing or to change the input bin). Because the AFP architecture defines those device functions in an AFP form definition, the transform ignores the device commands in the PCL data stream. To obtain these device functions, you must specify them in the AFP form definition resource objects or Infoprint Server job attributes.
Resolution conversion algorithms might produce a degraded appearance when used to reduce the resolution of a data stream. For this reason, the transform might degrade the appearance of higher-resolution data streams when used with 240-pel printers. You should verify that print fidelity is satisfactory.

Subtle differences exist between PCL4 and PCL5e related to handling fonts. While many PCL4 files work with the transform, some might not produce the expected output.

C.7.2 PDF to AFP limitations

These limitations apply to the PDF to AFP transform:

- The transform does not produce color output. It produces monochrome output only.
- The transform cannot process DBCS fonts.
- The PDF to AFP transform might not be able to transform very large PDF documents that contain non-balanced page trees. For information about page trees, see the Adobe PDF Reference, which is available on the Adobe Web site (www.adobe.com). Tip: Applications that create PDF documents by combining a large number of separate PDF documents sometimes create PDF documents with non-balanced tree structures.
- Resolution conversion algorithms might produce a degraded appearance when used to reduce the resolution of images imbedded in a data stream. For this reason, the transform might degrade the appearance of higher-resolution images when used with 240-pel printers. You should verify that print fidelity is satisfactory.
- This transform cannot create AFP FS45 image objects, which some IBM color printers require. However, the PDF to AFP transform available with IBM Infoprint Manager for AIX and Windows can create FS45 images. If your installation has installed Infoprint Manager for AIX or Windows, the administrator can set up the printer definitions for the color printers to run the transform remotely using Infoprint Manager.

C.7.3 PS to AFP limitations

These limitations apply to the PostScript to AFP transform:

- The transform does not produce color output. It produces monochrome output only.
- The transform cannot process DBCS fonts.
- PostScript data can contain device commands (for example, to begin or end duplexing or to change the input bin). Because the AFP architecture defines those device functions in a form definition resource, the transform command ignores the device commands in the print data. To access those device functions, you must specify them in the form definition or attributes file, or on a print command, when you print the job.
- Resolution conversion algorithms might produce a degraded appearance when used to reduce the resolution of images imbedded in a data stream. For this reason, the transform might degrade the appearance of higher-resolution images when used with 240-pel printers. You should verify that print fidelity is satisfactory.
- This transform cannot create AFP FS45 image objects, which some IBM color printers require. However, the PostScript to AFP transform available with IBM Infoprint Manager for AIX and Windows can create FS45 images. If your installation has installed Infoprint Manager for AIX or Windows, the administrator can set up the printer definitions for the color printers to run the transform remotely using Infoprint Manager.
C.7.4 Transform limitations

The following transforms have these limitations:

- AFP to PCL
- AFP to PDF
- AFP to PostScript

The items not supported by these transforms are:

- Include Page (IPG)
- Internal copy groups
- Map Color Attribute Table (MCA)
- Map Media Type (MMT)
- Map Page (MPG)
- Medium Finishing Control (MFC)
- Preprocess Presentation Object (PPO)
- Presentation Fidelity Control (PFC)
- QR bar codes
- SOSI4 for DBCS data

Following fonts are AFP to PostScript only:

- TrueType and OpenType fonts in AFP documents.
- UTF-16 fonts.

The transform cannot scale outline fonts. Therefore, the page definition you use for line-data and XML documents must specify a RATIO of 100 or omit the RATIO keyword.

Fonts not supported by the transforms:

- DBCS fonts in XML and record-format line data.
- DBCS outline fonts
- OpenType fonts
- TrueType fonts
- UTF-16 fonts
- Outline fonts - AFP to PCL only

**Note:** AFP to PCL transform can map single-byte outline fonts to raster fonts, but the transform cannot scale outline fonts. Therefore, the page definition you use for line-data and XML documents must specify a RATIO of 100 or omit the RATIO keyword.
IBM presentation architecture

The presentation architecture environment is a coordinated set of services architected to meet the presentation needs of current applications. The ability to create, store, retrieve, view, and print data in presentation formats friendly to people is a key requirement in almost every application of computers and information processing. This requirement is becoming increasingly difficult to meet because of the number of applications, servers, and devices that must interoperate to satisfy today’s presentation needs.

The solution is a presentation architecture base that is both robust and open ended, and easily adapted to accommodate the growing needs of the open system environment. IBM presentation architectures provide that base by defining interchange formats for data streams and objects that enable applications, services, and devices to communicate with one another to perform presentation functions. These presentation functions may be part of an integrated system solution or they may be totally separated from one another in time and space. IBM presentation architectures provide structures that support object-oriented models and client/server environments.
D.1 Supported objects and controls for AFP data streams

IBM presentation architectures provide the means for representing documents in a data format that is independent of the methods used to capture or create them. Documents may contain combinations of text, image, graphics and bar code objects in device-independent and resolution-independent formats. Documents may contain fonts, overlays and other resource objects required at presentation time to present the data properly. Finally, documents may contain resource objects, such as a document index and tagging elements supporting the search and navigation of document data, for a variety of application purposes.

In IBM, the presentation architecture components are divided into two major categories:

- Data streams
- Objects

It is important to understand what type of data streams that a transform supports or does not support. A data stream is a continuous ordered stream of data elements and objects conforming to a given format. Application programs can generate data streams destined for a presentation service, archive library, presentation device or another application program. The strategic presentation data stream architectures are:

- Mixed Object Document Content Architecture (MO: DCA)
- Intelligent Printer Data Stream (IPDS) Architecture

Documents can be made up of different kinds of data, such as text, graphics, image, and bar code. Object content architectures (OCAs) describe the structure and content of each type of data format that can exist in a document or appear in a data stream. Objects can be either data objects or resource objects.

D.1.1 Objects

Documents can be made up of different kinds of data, such as text, graphics, image, and bar code. Object content architectures describe the structure and content of each type of data format that can exist in a document or appear in a data stream. Objects can be either data objects or resource objects.

All object content architectures (OCAs) are totally self-describing and independently defined. When multiple objects are composed on a page, they exist as peer objects, which can be individually positioned and manipulated to meet the needs of the presentation application.

**Data objects**

A data object contains a single type of presentation data, that is, presentation text, vector graphics, raster image, or bar codes, and all of the controls required to present the data.

**Resource objects**

A resource object is a collection of presentation instructions and data. These objects are referenced by name in the presentation data stream and can be stored in system libraries so that multiple applications and the print server can use them.
D.1.2 Object content architecture (OCA)

All object content architectures (OCAs) are totally self-describing and independently defined. When multiple objects are composed on a page, they exist as peer objects, which can be individually positioned and manipulated to meet the needs of the presentation application.

The object content architectures are as follows:

- **Presentation Text Object Content Architecture (PTOCA)**
  A data architecture for describing text objects that have been formatted for all-points-addressable presentations. Specifications of fonts, text color, and other visual attributes are included in the architecture definition.

- **Image Object Content Architecture (IOCA)**
  A data architecture for describing resolution-independent image objects captured from a number of different sources. Specifications of recording formats, data compression, color and gray-scale encoding are included in the architecture definition.

- **Graphics Object Content Architecture (GOCA)**
  A data architecture for describing vector graphic picture objects and line art drawings for a variety of applications. Specification of drawing primitives, such as lines, arcs, areas, and their visual attributes, are included in the architecture definition.

- **Graphics Object Content Architecture for Advanced Function Presentation (AFP GOCA)**
  A version of GOCA that is used in Advanced Function Presentation (AFP) environments.

- **Bar Code Object Content Architecture (BCOCA)**
  A data architecture for describing bar code objects, using a number of different symbologies. Specification of the data to be encoded and the symbology attributes to be used are included in the architecture definition.

- **Font Object Content Architecture (FOCA)**
  A resource architecture for describing the structure and content of fonts referenced by presentation data objects in the document.

In addition to object content architectures, the MO:DCA architecture defines envelope architectures for objects of common value in the presentation environment. Examples of these are:

- **Form Definition resource objects** for managing the production of pages on the physical media
- **Overlay resource objects** that accommodate electronic storage of forms data
- **Index resource objects** that support indexing and tagging of pages in a document.

D.1.3 Supported objects for AFP transforms

AFP’s MO:DCA architecture is a device-independent data stream that governs the interchange of documents. Without such an architecture, information exchange is difficult and unpredictable. A subset of MO:DCA is Mixed Object Document Content Architecture for Presentation (MO:DCA-P), which defines presentation documents.

The AFP to PCL transform and AFP to PDF transform supports MO:DCA-P objects as follows:
Bar Code Object Content Architecture (BCOCA): All bar codes except QR bar codes. A QR Code is a two-dimensional bar code created by Japanese corporation Denso-Wave. The “QR” is derived from “Quick Response”.

Font Object Content Architecture (FOCA) that contains:
- Single-byte (SBCS) fonts:
  - 300-pel raster fonts, fixed metrics, and relative metrics.
  - The transform can map AFP outline fonts to equivalent raster fonts.
- Double-byte (DBCS) fonts:
  - 300-pel raster fonts, fixed metrics, and relative metrics. However, DBCS raster fonts are not supported in XML or record-mode line data.
  - AFP outline fonts are not supported.

Graphic Object Content Architecture (GOCA): All functions.

IM image data object: All functions, in single and double dot, in all rotations. An IM image data object specifies the content of a raster image and its placement on a page.

Image Object Content Architecture (IOCA):
- FS45 tiled and uncompressed image objects except: Tile Set Color parameter, Include Tile parameter, Referencing Tile structure, IOCA Tile Resource structure.
- Uncompressed, compressed MMR, G3, G4, RL4, ABIC (non-concatenated), JPEG baseline and extended.
- IDE 1-8, 24 (lookup table).

Object containers for GIF, JFIF, JPEG, PDF, and TIFF image objects.

PTOCA1, PTOCA2.

D.1.4 AFP resources for transforms

The transform supports the following AFP resources:
- Page definitions, including conditional processing, record format page definitions, and XML page definitions. A page definition is the resource that contains formatting specifications for line data or XML data.
- Form definitions, including basic N_UP processing and enhanced N_UP processing. N_UP allows printing of multiple pages on a sheet. A form definition contains one or more copy groups. The copy group assigns horizontal and vertical offsets to position the top left corner of the logical page on the physical form (sheet).
- Overlays. An overlay is a collection of predefined data that can be printed on a page by itself or merged with other data on a page as the page is printed.
- Page segments. A page segment is an object that can be merged with the variable data on a page being printed. It usually contains image data such as bar codes, signatures, logos, or graphics converted into image format.
- User resource libraries. AFP to PCL resources are accessed from user libraries that are not defined in the transform entry being used.

The job submitter must have RACF authorization to read all AFP resource libraries that the transform uses.
Line data
The AFP to PCL transform supports the following data streams:

- Carriage control (ANSI, machine). CC characters are used in traditional line data to control writing, spacing, and skipping operations as the data is being formatted.
- Table reference characters (TRC). In traditional line data, you can use different fonts on different lines of a file by specifying TRCs at the beginning of each line after the carriage control characters, if any are present.
- Shift out/shift (SOSI) in DBCS data. Special EBCDIC or ASCII characters that exist in the data stream to indicate the switches between double-byte fonts and single-byte fonts.
- SOSI1, SOSI2, and SOSI3 options. A data set that contains both single-byte and double-byte character codes can necessitate coding the structured fields in the data set to change from one kind of character code to the other. To avoid doing this, you can request special processing by specifying SOSI1, SOSI2, SOSI3, or SOSI4. The PRMODE JCL parameter identifies the process mode required to process a sysout data set.
- Mixed line data and AFP records (including Invoke Data Map (IDM), Invoke Media map (IMM)). In print data an IDM structured field can change the page format. In print data an IMM structured field can change the copy group.
- Record-format line data. In record format line data, each data record contains a record identifier, which selects the record descriptor (RCD) in a record format page definition that is used to format the line data.

Line data
For line data, the AFP to PDF transform supports the following:

- Carriage control (ANSI, machine).
- Table reference characters.
- Shift out/shift in DBCS data - SOSI1, SOSI2, and SOSI3 options.
- Mixed line data and AFP records (including IDM, IMM).
- Record-format line data.

XML data
XML data encoded in EBCDIC, ASCII, UTF-8, or UTF-16 is supported.
Transform filters

This appendix describes the transform filters that transform documents to another data format. The following transform filters are described:

- Infoprint Transform for AFP to HP PCL for z/OS
  - afp2pcl.dll filter for AFP to PCL transform
- Infoprint Transform for AFP to Adobe PDF for z/OS
  - afp2pdf.dll filter for AFP to PDF transform
- Infoprint Transform for AFP to Adobe PostScript for z/OS
  - afp2ps.dll filter for AFP to PS transform
- Infoprint Transforms to AFP for z/OS
  - pcl2afp.dll filter for PCL to AFP transform
  - ps2afp.dll filter for PDF to AFP and PS to AFP transforms
  - sap2afp.dll filter for SAP to AFP transform
- Infoprint Transform Manager for Linux
  - aoprxf.so filter to send documents to Linux for all transforms
- Infoprint XML Extender for z/OS
  - xml2afp.dll filter for XML to AFP transform
  - xml2pdf.dll filter for XML to PDF transform
- Infoprint XT Extender for z/OS
  - x2afp.dll filter for Xerox to AFP transform
- Infoprint Manager for AIX (5785–E42) and Infoprint Manager for Windows (5639–I27)
  - aoprform.dll filter to send documents to Infoprint Manager for AIX or Windows.

A filter program can accept options which control the transform processing. This appendix is a summary of the options that you can specify for the various transform filters.
E.1 Filter options for transform filters

You can specify filter options for the following transform filters:

- pcl2afp.dll, pdf2afp.dll, ps2afp.dll, sap2afp.dll, afp2pcl.dll, afp2ps.dll, afp2pdf.dll, xml2afp.dll, and xt2afp.dll, as follows:

  [%filter-options] [option ...]

<table>
<thead>
<tr>
<th>Transform filter option</th>
<th>Filter description</th>
</tr>
</thead>
<tbody>
<tr>
<td>%filter-options</td>
<td>Causes any options that a job submitter specified in the filter-options job attribute to be passed directly to the transform. You can type %filter-options in any position relative to other filter options. If you specify transform options to the right of %filter-options, those options override the same options that were specified in the filter-options job attribute, with the exception of any options that are cumulative. If an option is cumulative, the transform applies all occurrences of the option.</td>
</tr>
<tr>
<td>option</td>
<td>Any option that the transform accepts. Separate multiple options with a space. The transforms from AFP use options that are specified in the transform configuration file aopxfd.conf. Transform entries in the transform configuration file specify: Environment variables that control the transforms Attributes that control how the Infoprint Server Transform Interface manages the transform. For more information about the transform options, see “Infoprint Server transforms” on page 25 or one of these books: IBM Infoprint Transforms from AFP for z/OS Version 2.1, G550-0444 IBM Infoprint Transforms to AFP for z/OS Version 2.1, G550-0443 Infoprint XML Extender for z/OS, S544-5855 Infoprint XT Extender for z/OS: Customization and Usage, S544-5879 Note: Infoprint Server does not check the syntax of the options. It passes them directly to the transforms. <strong>Attention:</strong> The -c option is used by all filters to specify the transformclass.</td>
</tr>
<tr>
<td>-c transformclass</td>
<td>Specifies the name of a transform class that is defined in the transform configuration file, aopxfd.conf. The name is case-sensitive. The job submitter can also specify this option in the filter-options job attribute.</td>
</tr>
</tbody>
</table>
E.2 Filter options for the pcl2afp.dll filter

In the printer definitions for AFP printers, you can specify the PCL to AFP transform filter and associate it with the PCL data format. (A filter is a program that modifies the input data before it is sent to the printer.) When you associate the transform filter with the PCL data format, Infoprint Server automatically calls the PCL to AFP transform when it processes a document with the PCL data format.

pcl2afp.dll

[\%filter-options] [-a imagetype] [-c transformclass]
[-p pagerange] [-t outputtype]

Table E-2  Filter options for transform PCL to AFP

<table>
<thead>
<tr>
<th>Transform filter options</th>
<th>Filter descriptions for -apt for PCL to AFP</th>
</tr>
</thead>
<tbody>
<tr>
<td>-a imagetype</td>
<td>Determines the type of AFP data stream image to generate for each page in the PCL file. Values are:</td>
</tr>
<tr>
<td></td>
<td>io1-g4  Compressed Image Object Content Architecture (IOCA) image in Modified Telecommunication Standardization Sector (TSS) T.6 G4 Facsimile Coding Scheme (G4 MMR) format. This is the recommended output type because it takes up less space on the hard disk, and it prints faster.</td>
</tr>
<tr>
<td></td>
<td>Note: Some older AFP printers do not support printing with an image type of io1-g4. For these printers, specify an image type of io1-mmr because it is the compressed image type that they support. This image type results in faster printing than uncompressed image types.</td>
</tr>
<tr>
<td></td>
<td>TSS was formerly the International Telegraph and Telephone Consultative Committee (CCITT).</td>
</tr>
<tr>
<td></td>
<td>im1 IM1 image. This type of image is not compressed.</td>
</tr>
<tr>
<td></td>
<td>io1 IOCA image. This type of image is not compressed.</td>
</tr>
<tr>
<td></td>
<td>io1-mmr Compressed IOCA image in Modified Modified Read (MMR) format.</td>
</tr>
<tr>
<td>-p pagerange</td>
<td>Specifies that the output should contain only selected pages. The -p option counts pages by their actual sequence in the document, not by page number. For example, to write only the last page of a document whose pages are numbered i, ii, 1, 2, 3, 4, specify -p 6.</td>
</tr>
<tr>
<td></td>
<td>Examples of values include:</td>
</tr>
<tr>
<td></td>
<td>-p 1-10  Write the first through tenth pages.</td>
</tr>
<tr>
<td></td>
<td>-p 10-   Write pages from the tenth page until the end of the file.</td>
</tr>
<tr>
<td>-t outputtype</td>
<td>Determines the type of output to generate. Values are:</td>
</tr>
<tr>
<td></td>
<td>document  Printable document.</td>
</tr>
<tr>
<td></td>
<td>overlay    Graphic image that can be printed on each page of a printable document.</td>
</tr>
<tr>
<td></td>
<td>pagesegment Graphic image that can be embedded in a printable document.</td>
</tr>
<tr>
<td></td>
<td>When generating overlays or page segments from multiple-page documents, use the -p option to select a single page. Otherwise, the output will contain multiple overlays or page segments (one for each page). The AFP architecture does not allow multiple, concatenated overlays or page segments.</td>
</tr>
</tbody>
</table>

Note: For the -apt options, see IBM Infoprint Transforms to AFP for z/OS Version 2.1, G550-0443.
E.3 Filter options for the pdf2afp.dll filter

ps2afp.dll

[-r resolution] [-t outputtype] [-w width] [-x xmargin] [-y ymargin]

Table E-3  Filter options for transform PDF to AFP

<table>
<thead>
<tr>
<th>Transform filter options</th>
<th>Filter descriptions for PDF to AFP</th>
</tr>
</thead>
<tbody>
<tr>
<td>-a imagetype</td>
<td>Determines the type of AFP data stream image to generate for each page in the PCL file. Values are:</td>
</tr>
<tr>
<td></td>
<td>io1-g4</td>
</tr>
<tr>
<td></td>
<td>im1</td>
</tr>
<tr>
<td></td>
<td>io1</td>
</tr>
<tr>
<td></td>
<td>io1-mmr</td>
</tr>
<tr>
<td>Transform filter options</td>
<td>Filter descriptions for PDF to AFP</td>
</tr>
<tr>
<td>--------------------------</td>
<td>-----------------------------------</td>
</tr>
<tr>
<td><strong>-l length</strong></td>
<td>Specifies the length of the generated image. In general, specify the length of the physical page.</td>
</tr>
<tr>
<td></td>
<td>In general, to position data on the page:</td>
</tr>
<tr>
<td></td>
<td>▶ Use <code>-l</code> and <code>-w</code> to set the physical page dimensions.</td>
</tr>
<tr>
<td></td>
<td>▶ Use <code>-x</code> and <code>-y</code> to set the amount of white space between the physical page dimensions and the image.</td>
</tr>
<tr>
<td></td>
<td>These options do not shift or scale the image on the page. If the image is defined to print in the unprintable areas, it is cropped.</td>
</tr>
<tr>
<td></td>
<td>Specify a number followed by one of units:</td>
</tr>
<tr>
<td></td>
<td>- <code>in</code> Inches</td>
</tr>
<tr>
<td></td>
<td>- <code>mm</code> Millimeters</td>
</tr>
<tr>
<td></td>
<td>- <code>pel</code> Pels, the default unit</td>
</tr>
<tr>
<td></td>
<td>Inch values and millimeter values can contain a decimal point. Pel values cannot.</td>
</tr>
<tr>
<td></td>
<td>Values are:</td>
</tr>
<tr>
<td></td>
<td>11in 11inches, the default for all printers</td>
</tr>
<tr>
<td></td>
<td>0.1334in to 53in Inch values for 240-pel printers</td>
</tr>
<tr>
<td></td>
<td>0.1067in to 53in Inch values for 300-pel printers</td>
</tr>
<tr>
<td></td>
<td>0.0667in to 53in Inch values for 480-pel printers</td>
</tr>
<tr>
<td></td>
<td>0.0534in to 53in Inch values for 600-pel printers</td>
</tr>
<tr>
<td></td>
<td>3.3867mm to 1346.2mm Millimeter values for 240-pel printers</td>
</tr>
<tr>
<td></td>
<td>2.7094mm to 1346.2mm Millimeter values for 300-pel printers</td>
</tr>
<tr>
<td></td>
<td>1.6934mm to 1346.2mm Millimeter values for 480-pel printers</td>
</tr>
<tr>
<td></td>
<td>1.3547mm to 1346.2mm Millimeter values for 600-pel printers</td>
</tr>
<tr>
<td></td>
<td>32pel to 12720pel Pel values for 240-pel printers</td>
</tr>
<tr>
<td></td>
<td>32pel to 15900pel Pel values for 300-pel printers</td>
</tr>
<tr>
<td></td>
<td>32pel to 25440pel Pel values for 480-pel printers</td>
</tr>
<tr>
<td></td>
<td>32pel to 31800pel Pel values for 600-pel printers</td>
</tr>
<tr>
<td></td>
<td>If a text margin is already built into the file, try <code>-l 11in</code> to set the length to 11 inches.</td>
</tr>
<tr>
<td><strong>-p pagerange</strong></td>
<td>Specifies that the output should contain only selected pages. The <code>-p</code> option counts pages by their actual sequence in the document, not by page number.</td>
</tr>
<tr>
<td></td>
<td>Examples of values include:</td>
</tr>
<tr>
<td></td>
<td><code>-p 1-10</code> Write the first through tenth pages.</td>
</tr>
<tr>
<td></td>
<td><code>-p 10-</code> Write pages from the tenth page until the end of the file.</td>
</tr>
<tr>
<td><strong>-r resolution</strong></td>
<td>Specifies the resolution of the output image. Select the correct resolution for the printer on which you intend to print the image. Values are:</td>
</tr>
<tr>
<td></td>
<td>240 240 pels per inch (for example, IBM 3812, 3825, 3827, 3835, and 3900 printers)</td>
</tr>
<tr>
<td></td>
<td>300 300 pels per inch (for example, IBM 4019, 4028, 4029, and 4039 printers and some Hewlett-Packard printers)</td>
</tr>
<tr>
<td></td>
<td>480 480 pels per inch</td>
</tr>
<tr>
<td></td>
<td>600 600 pels per inch (for example, IBM Infoprint 60 and Infoprint 4000 printers), the default</td>
</tr>
<tr>
<td></td>
<td>If you specify a resolution that the printer does not support, PSF prints the image under most conditions, but with degraded results.</td>
</tr>
</tbody>
</table>
### Transform filter options

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>-t outputtype</code></td>
<td>Determines the type of output to generate. Values are:</td>
</tr>
<tr>
<td></td>
<td><strong>document</strong></td>
</tr>
<tr>
<td></td>
<td><strong>overlay</strong></td>
</tr>
<tr>
<td></td>
<td><strong>pagesegment</strong></td>
</tr>
</tbody>
</table>

When generating overlays or page segments from multiple-page documents, use the `-p` option to select a single page. Otherwise, the output will contain multiple overlays or page segments (one for each page). The AFP architecture does not allow multiple, concatenated overlays or page segments.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>-w width</code></td>
<td>Specifies the maximum width of the generated image. In general, specify the width of the physical page. For valid values see the <code>-l length</code> option.</td>
</tr>
<tr>
<td><code>-x xmargin</code></td>
<td>Specifies a horizontal margin or border around the generated image to avoid the non-printable areas of some printers. Specify a number followed by one of units:</td>
</tr>
<tr>
<td></td>
<td><strong>in</strong> Inches</td>
</tr>
<tr>
<td></td>
<td><strong>mm</strong> Millimeters</td>
</tr>
<tr>
<td></td>
<td><strong>pel</strong> Pels, the default unit</td>
</tr>
</tbody>
</table>

Inch values and millimeter values can contain a decimal point. Pel values cannot.

Values are:

- 0 Zero, the default for all printers
- 0in to 12.75in Inch values for all printers
- 0mm to 323.85mm Millimeter values for all printers
- 0pel to 3060pel Pel values for 240-pel printers
- 0pel to 3825pel Pel values for 300-pel printers
- 0pel to 6120pel Pel values for 480-pel printers
- 0pel to 7650pel Pel values for 600-pel printers

Because the X value specifies margins on both the left and right sides of the page, the X value can be no more than half of the width (`-w`) of the generated image. For example, if you specify a width of 8 inches, the X value can be no larger than 4 inches. If you specify an X value of 5 inches, a blank page is printed because the sum of the left and right margins exceeds the width of the paper.

The X value does not shift the image on the page. The image is cropped if it is defined to print in the left or right margin.
E.4 Filter options for the ps2afp.dll filter

The ps2afp command converts a PostScript data file into an Advanced Function Presentation (AFP) data stream file. If you specify the same option multiple times, the command uses only the last option, with the exception of the -i option. You can specify the -i option multiple times.

You can specify one or more input files to be transformed. If you do not specify an input file name, or if you specify a dash (-) as the file name, ps2afp uses standard input. The output file name is also optional. If you do not specify one, the ps2afp command writes the results to standard output.

```
ps2afp.dll
[filter-options] [-a imagetype] [-c transformclass]
[-l length] [-p pagerange] [-r resolution] [-t outputtype]
[-w width] [-x xmargin] [-y ymargin]
```

**Note:** The filter options for this transform are identical to PDF to AFP and are shown in Table E-3 on page 242. Option -i is used for this transform in addition to the other options and is shown in Table E-4 on page 246.
Table E-4  Filter options for PS to AFP

<table>
<thead>
<tr>
<th>Transform filter options</th>
<th>Filter description for transform PS to AFP</th>
</tr>
</thead>
<tbody>
<tr>
<td>-i initializationfile</td>
<td>Specifies an ASCII PostScript file that is added to the start of the job to set up and initialize the PostScript transform. If you specify the -i option multiple times, the initialization files are processed in the order that you specify them.</td>
</tr>
</tbody>
</table>

Note: For the -ailprtwxy options for the PS to AFP transform, see IBM Infoprint Transforms to AFP for z/OS Version 2.1, G550-0443.

E.5 Filter options for the sap2afp.dll filter

In the printer definitions for AFP printers, you can specify the SAP to AFP transform filter and associate it with the PCL data format. (A filter is a program that modifies the input data before it is sent to the printer.) When you associate the transform filter with the SAP data format, Infoprint Server automatically calls the SAP to AFP transform when it processes a document with the SAP data format.

**sap2afp.dll**

```
[%filter-options] [-p pagerange] ... [-r resolution] [-s]
```

Table E-5  Filter options for transform SAP to AFP

<table>
<thead>
<tr>
<th>Transform filter options</th>
<th>Filter descriptions for transform SAP to AFP</th>
</tr>
</thead>
</table>
| -p pagerange             | Specifies that the output should contain only selected pages. If you specify the -p option multiple times, the pages are processed in the order you specify them. You can specify the -p option up to 20 times. The -p option counts pages by their actual sequence in the document, not by page number. For example, to write only the last page of a document whose pages are numbered i, ii, 1, 2, 3, 4, specify -p 6.

Examples of values include:

- -p even Write even pages.
- -p odd Write odd pages.
- -p 1-10 Write the first through tenth pages.
- -p 1-10- Write pages from the tenth page until the end of the job.
- -p 1 -p 3 -p 6 Write the first, third, and sixth pages.

<table>
<thead>
<tr>
<th>-r resolution</th>
<th>Specifies the resolution of the output image. Select the correct resolution for the printer on which you intend to print the image. Values are:</th>
</tr>
</thead>
<tbody>
<tr>
<td>240 240 pels per inch (for example, IBM 3812, 3825, 3827, 3835, and 3900 printers)</td>
<td></td>
</tr>
<tr>
<td>300 300 pels per inch (for example, IBM 4019, 4028, 4029, and 4039 printers and some Hewlett-Packard printers)</td>
<td></td>
</tr>
<tr>
<td>480 480 pels per inch</td>
<td></td>
</tr>
<tr>
<td>600 600 pels per inch (for example, IBM Infoprint 60 and Infoprint 4000 printers), the default</td>
<td></td>
</tr>
</tbody>
</table>

The default resolution is the resolution defined by the administrator. If the resolution has not been set, the transform fails with error message AOP2009E.

If you specify a resolution that the printer does not support, PSF prints the image data under most conditions, but with degraded results.

| -s                        | Suppresses Graphic Object Content Architecture (GOCA) boxes. Some older printers do not print these boxes. |

Note: For the -prs options, see IBM Infoprint Transforms to AFP for z/OS Version 2.1, G550-0443.
E.6 Filter options for the afp2pcl.dll filter

In the printer definitions for PCL printers, you can specify the AFP to PCL transform filter and associate it with the line-data, MO:DCA-P, and XML data formats. (A filter is a program that modifies the input data before it is sent to the printer.) When you associate the transform filter with these data formats, Infoprint Server automatically calls the AFP to PCL transform before it sends documents with these formats to the printer.

**Note:** If you run IP PrintWay basic mode, you must also select the resubmit for filtering function in the printer definition. IP PrintWay basic mode calls transform filters only when you select the IP PrintWay resubmit for filtering function.

```
afp2pcl.dll
   [%filter-options] [-c transformclass] [-i inputcodepage]
   [-F tracefile] [-T traceoptions]
```

**Note:** For the `-iFT` options, see Appendix , “AFP to PCL, AFP to PDF, and AFP to PostScript %filter-options” on page 248 or also see IBM Infoprint Transforms from AFP for z/OS Version 2.1, G550-0444.

E.7 Filter options for the afp2ps.dll filter

AFP to PostScript transform filter and associate it with the line-data, MO:DCA-P, and XML data formats. (A filter is a program that modifies the input data before it is sent to the printer.) When you associate the transform filter with these data formats, Infoprint Server automatically calls the AFP to PostScript transform before it sends documents with these data formats to the printer.

**Note:** If you run IP PrintWay basic mode, you must also select the resubmit for filtering function in the printer definition. IP PrintWay basic mode calls transform filters only when you select the IP PrintWay resubmit for filtering function.

```
afp2ps.dll
   [%filter-options] [-c transformclass] [-i inputcodepage]
   [-F tracefile] [-T traceoptions]
```

**Note:** For the `-iFT` options, see Appendix , “AFP to PCL, AFP to PDF, and AFP to PostScript %filter-options” on page 248 or also see IBM Infoprint Transforms from AFP for z/OS Version 2.1, G550-0444.

E.8 Filter options for the afp2pdf.dll filter

AFP to PDF transform filter and associate it with the line-data, MO:DCA-P, and XML data formats. (A filter is a program that modifies the input data before it is sent to the printer.) When you associate the transform filter with these data formats, Infoprint Server automatically calls the AFP to PDF transform before it sends documents with these data formats to the e-mail recipients.
**Note:** If you run IP PrintWay basic mode, you must also select the resubmit for filtering function in the printer definition. IP PrintWay basic mode calls transform filters only when you select the IP PrintWay resubmit for filtering function.

**afp2pdf.dll**

```bash
[filter-options] [-c transformclass] [-i inputcodepage]
[-F tracefile] [-T traceoptions]
```

**Note:** For the `-iFT` options, see Appendix, “AFP to PCL, AFP to PDF, and AFP to PostScript %filter-options” on page 248 or also see *IBM Infoprint Transforms from AFP for z/OS Version 2.1*, G550-0444.

**AFP to PCL, AFP to PDF, and AFP to PostScript %filter-options**

The filter options are identical for these three transforms and are shown in Table E-6 on page 249.
### Table E-6 Filter options for transforms AFP to PCL, AFP to PDF, and AFP to PS

<table>
<thead>
<tr>
<th>Transform filter options -iFT</th>
<th>Filter descriptions for AFP to PCL, AFP to PDF, and AFP to PS</th>
</tr>
</thead>
</table>
| `-i inputcodepage`           | This option applies only to the Line data data format. If you specify this option for any other data formats, it is ignored. The option identifies the code page to which line data is converted before it is transformed. Specify a code page that corresponds to the coded fonts that the transform uses to transform the line data. (Coded fonts can be specified in the page definition, in the chars job attribute or CHARS JCL parameter, and in the Character sets field of the printer definition.) To transform line data that is already encoded in the code page that corresponds to the coded fonts, do not specify this option. When this option is not specified, line data is not converted from one code page to another before it is transformed. For example, to transform line-data documents that specify coded fonts (for example, in the CHARS JCL parameter) and currently print correctly on an AFP printer, do not specify this option. You must specify this option to correctly transform documents that are encoded in code pages that do not correspond to the code page for the coded fonts. This is most likely to occur when the Print Interface LPD receives print requests with a print command of `r` in the LPD control file. Print command `r` indicates that the file contains ANSI carriage control characters (FORTRAN carriage control) and is, therefore, line data. To specify a code page, first determine the AFP code page for each character set. For AFP code pages, see *IBM AFP Fonts: Font Summary for AFP Font Collection*. Then determine the name of the corresponding code page that IBM provides and that the iconv utility supports. For valid code page names, see *z/OS C/C++ Programming Guide*. The AFP code page and the name of the code pages that iconv uses are different. For example, if the coded fonts in the list below are specified (for example, in the CHARS JCL parameter), specify the IBM-500 code page in the `-i` option: `-i IBM-500.

- Coded font 40D0, 40F0, 40E0, 4100 (CHARS JCL parameter)
  - AFP code page T1V10500
  - iconv code page IBM-500

- Coded font 60D9 (CHARS JCL parameter - default font)
  - AFP code page T1V10500
  - iconv code page IBM-500

When you specify the `-i` option, also make sure that the code page specified in the Document code page field (on Processing ISPF panel) or document-codepage (Printer Inventory Definition Utility) of the printer definition identifies the code page in which input documents are encoded. A job submitter can also specify a document code page for a specific print job in the document-codepage job attribute. When you specify this option, you might need to create a separate printer definition for use only by those applications that require the `-i` option and code page conversion. |
| `-F tracefile` `-T traceoptions` | These options trace the transforms. Your IBM service representative might ask you to specify these options to help IBM diagnose problems. These options can also be specified in the filter-options job attribute. For an explanation of these options, see “Infoprint Server transform error messages” on page 44. |
E.9 Filter options for the xml2afp.dll and xt2afp.dll filters

xml2afp.dll

[-s {stylesheet | none}] [-w pagewidth]

Table E-7 Filter options for transform XML to AFP

<table>
<thead>
<tr>
<th>Transform filter options</th>
<th>Filter descriptions for Transform XML to AFP</th>
</tr>
</thead>
<tbody>
<tr>
<td>-T</td>
<td>Requests a trace for diagnostic purposes. Trace output is written to stderr.</td>
</tr>
<tr>
<td>-d destinationtype</td>
<td>Indicates the type of destination, which affects the output generated. For example, different types of destinations can support different font technologies and different color specifications. If the destination type is not specified, the default is outline fonts, no color, with 600 dpi resolution. Otherwise, valid values are those listed in the device types file, which determines the properties for a destination type.</td>
</tr>
<tr>
<td>-l pagelength[in</td>
<td>mm]</td>
</tr>
<tr>
<td></td>
<td>in</td>
</tr>
<tr>
<td></td>
<td>mm</td>
</tr>
<tr>
<td></td>
<td>The default is 11 inches. If the transform is managed by Infoprint Server, the default is specified in the definition for the transform class. See the -c option.</td>
</tr>
<tr>
<td>-s {stylesheet</td>
<td>none}</td>
</tr>
<tr>
<td></td>
<td>stylesheet - Specifies the file or URL for an XSL style sheet that is used to transform XML data to XSL-FO data. This option should only be specified when the input file contains XML data that must be transformed according to a specified style sheet, not when the input file contains XSL-FO data.</td>
</tr>
<tr>
<td></td>
<td>none - Indicates that a style sheet should be suppressed because it has been specified with XSL-FO data. This value is only valid when the transform is managed by Infoprint Server.</td>
</tr>
<tr>
<td></td>
<td>When XML Extender is a standalone transform, it runs with the user’s authority. When XML Extender is a managed transform, it runs with limited authority (as determined by the system administrator), which usually means it can only access unprotected data. Therefore, if the transform is managed by Infoprint Server, XML Extender must be able to access the file or resource named by the style sheet URL. For example, if report.xsl is the style sheet file a URL refers to, its file permissions must allow others to read it.</td>
</tr>
<tr>
<td>-w pagewidth[in</td>
<td>mm]</td>
</tr>
<tr>
<td></td>
<td>in</td>
</tr>
<tr>
<td></td>
<td>mm</td>
</tr>
<tr>
<td></td>
<td>If the transform is managed by Infoprint Server, the default is specified in the definition for the transform class. See the -c option.</td>
</tr>
</tbody>
</table>
E.10 Filter options for the aoprform.dll filter

The Infoprint Server remote transform filter for Infoprint Manager for AIX or Windows (aoprform.dll). The aoprform.dll transform sends data to be transformed to Infoprint Manager for AIX or Windows.

**aoprform.dll**

```
[-q transformattributes]... [-r resolution] [-t outputtype] [-w width]
[-x xoffset] [-y yoffset] ipaddress
```

Table E-8 Transform options for filter aoprform.dll

<table>
<thead>
<tr>
<th>Transform filter options</th>
<th>Filter option descriptions for aoprform.dll</th>
</tr>
</thead>
<tbody>
<tr>
<td>-a imagetype</td>
<td>The type of AFP data stream image that the transform generates for each page in the PCL, PostScript, or PDF file. Valid values are:</td>
</tr>
<tr>
<td></td>
<td><strong>fs45</strong> - IOCA color FS45 images. Specify this value for IBM color printers such as the IBM Infoprint Color 1454 or 1464 printer. Specify this value only for the PostScript and PDF data formats.</td>
</tr>
<tr>
<td></td>
<td><strong>im1</strong> - IM1 image. This type of image is not compressed. Specify it only if you know that your printer does not support compressed images.</td>
</tr>
<tr>
<td></td>
<td><strong>io1</strong> - IOCA image. This type of image is not compressed. Specify it only if you know that your printer does not support compressed images.</td>
</tr>
<tr>
<td></td>
<td><strong>io1-g4</strong> - Compressed Image Object Content Architecture (IOCA) image in Modified Telecommunication Standardization Sector (TSS) T.6 G4 Facsimile Coding Scheme (G4 MMR) format. This is the recommended output type because it takes up less space on the hard disk, and it prints faster. <strong>Note:</strong> Some older AFP printers do not support printing with an image type of io1-g4. For these printers, specify an image type of io1-mmr because it is the compressed image type supported by these printers. This image type results in faster printing than uncompressed image types. TSS was formerly the International Telegraph and Telephone Consultative Committee (CCITT).</td>
</tr>
<tr>
<td></td>
<td><strong>io1-mmr</strong> - Compressed IOCA image in Modified Modified Read (MMR) format.</td>
</tr>
<tr>
<td></td>
<td><strong>Tip: Image Object Content Architecture (IOCA).</strong> A data architecture for describing resolution-independent image raster objects captured from a number of different sources. Specifications of recording formats, data compression, color and gray-scale encoding are included in the architecture definition.</td>
</tr>
<tr>
<td></td>
<td><strong>Graphics Object Content Architecture (GOCA).</strong> A data architecture for describing vector graphic picture objects and line art drawings for a variety of applications. Specification of drawing primitives, such as lines, arcs, areas, and their visual attributes, are included in the architecture definition.</td>
</tr>
<tr>
<td>-l length</td>
<td>This option is passed directly to the Infoprint Manager transform daemon. For the values you can specify, see the ps2afp or pcl2afp command description in IBM Infoprint Manager: Reference, SS44-5475.</td>
</tr>
<tr>
<td>Transform filter options</td>
<td>Filter option descriptions for aoprform.dll</td>
</tr>
<tr>
<td>--------------------------</td>
<td>-------------------------------------------</td>
</tr>
</tbody>
</table>
| -P                       | The TCP/IP port number on the AIX or Windows system at which the transform daemon is receiving data. Consult the AIX or Windows administrator for the correct value to specify. If you omit this option, the default port for the type of input data is used. Valid values are:  
  8251 The default port number for the PostScript and PDF data formats.  
  8253 The default port number for the PCL data format.  
  924 - nnnnn Any valid port number greater than or equal to 924 at which the transform daemon is receiving data. |
| -p pagerange             | This option is passed directly to the Infoprint Manager transform daemon. For the values you can specify, see the ps2afp or pcl2afp command description in IBM Infoprint Manager: Reference, S544-5475. |
| -q transformattributes   | For FS45 image output (–a fs45), you can specify transform attributes and values. For a detailed description of the attributes and values, see IBM Infoprint Manager: Reference, S544-5475. |
| color-profile={euroscale | Specifies the color profile that the transform is to use. The default value is none. You can also specify synonym Euroscale for euroscale and SWOP for swop. |
  | none | swop} |
| color-rendering-intent={relative | Specifies how the transform is to process shades that the color printer cannot reproduce exactly. The default value is relative. |
  | perceptual} |
| color-toner-saver={no | Specifies whether the transform is to reduce the amount of color toner used by the printer. The default value is no. You can also specify synonym false for no and true for yes. |
  | yes} |
| presentation-object-contain er-extraction-mode={ignore |
  | inline} |
| -r resolution            | This option is passed directly to the Infoprint Manager transform daemon. |
| -t outputtype            | Determines the type of output to generate. Valid values are:  
  document Printable document. This is the default value.  
  overlay Graphic image that can be printed on each page of a printable document.  
  pagesegment Graphic image that can be embedded in a printable document. Note: When you generate overlays or page segments from multiple-page documents, the user might want to use the -g or -p option to select pages. Otherwise, one overlay or page segment is created for each page of the input file. |
| -w width                 | This option is passed directly to the Infoprint Manager transform daemon. |
| -x xoffset               | This option is passed directly to the Infoprint Manager transform daemon. |
| y yoffset                | This option is passed directly to the Infoprint Manager transform daemon. |
## E.11 Filter options for the aoprxf.so filter

The aoprxf.so transform sends data to be transformed to Infoprint Transform Manager for Linux. To alleviate demanding and processor intensive datastream transformations, IBM Infoprint Transform Manager for Linux allows PostScript, PDF, TIFF, GIF and JPEG RIPing to be offloaded to a cost effective dedicated system for processing and then returned to Infoprint Manager for spooling and printing on AFP printers. Infoprint Manager continues to manage and monitor print jobs, but transforms are shifted to a dedicated server to help improve output workflow and efficiency.

```plaintext
aoprxf.so
 [%filter-options] [%xf-options] [-xf "transformattributes"]... [option ...]
```

### Table E-9   Filter options for the aoprxf.so filter

<table>
<thead>
<tr>
<th>Transform filter options</th>
<th>Filter option descriptions for aoprform.dll</th>
</tr>
</thead>
<tbody>
<tr>
<td>ipaddress</td>
<td>The host name or dotted decimal address of the AIX or Windows system on which the transform daemon is running. This is a required option. For example, 9.99.9.23 or AIX4. You can specify the IP address in the dotted-decimal format. If the AIX or Windows system has an IPv6 address, you must specify its host name instead of its colon-hexadecimal format address. <strong>Note</strong>:.Infoprint Manager transforms, by default, create 32K records. To limit the maximum AFP record length to 8K bytes, the Infoprint Manager administrator must add this statement to the Infoprint Manager configuration file for the transform (for example, configuration file ps2afpd.cfg): pragma=AFPDSRECORDLENGTH=8000.</td>
</tr>
</tbody>
</table>

| %xf-options              | Causes transform attributes that a job submitter specified in the xf-options job attribute to be used. You can type %xf-options in any position relative to -xf option. If you specify the -xf option to the right of %xf-options, the transform attributes specified in the -xf option override the transform attributes that were specified in the xf-options job attribute. |

---
### Transform filter options

<table>
<thead>
<tr>
<th>-xf &quot;transformattributes&quot;</th>
<th>Filter descriptions for for aoprxf.so</th>
</tr>
</thead>
</table>

In this option, you can specify one or more of these transform attributes:

- **fail-on-datastream-error** - Indicates whether the transformation of the document fails if the data stream has errors. Use this transform attribute if you want to make sure that output is produced only if it does not have errors. Values are:
  - **yes** The transformation of the document fails if the data stream has any errors, including recoverable data stream errors. Output is produced only if it does not have any errors. If any data stream errors occur, the exit value for the transform is >0. Error messages for data stream errors are not included on a trailer error page (a page that is written at the end of the output data).
  - **no** The transformation of the document does not fail if the data stream has a recoverable error. Output is produced even if it has errors. If a recoverable data stream error occurs, the exit value for the transform is 0.

- **trailer-error-page** - Indicates whether error messages for recoverable data stream errors are included on a trailer error page (a page that is written at the end of the output data). Use this transform attribute if you want to make sure that error pages are not created. Values are:
  - **yes** Error messages are included on a trailer error page.
  - **no** A trailer error page is not created.

- **transform-id** - Specifies an Infoprint Transform Manager for Linux transform ID for a specific transform. You do not need to specify this transform attribute to use any of the transforms that Infoprint Transform Manager for Linux currently supports. However, if it adds new transforms to the AFP format in the future, you can use this transform attribute to specify them.

**Note:**
- Specify attributes in this format: attribute=value.
- If you specify more than one attribute, separate the attributes with a space.
- You can abbreviate the attribute names and values.
- Use the exact uppercase and lowercase letters for the attribute and values.
- If the -xf value contains any spaces, enclose the entire value in single or double quotation marks.
- You can specify the -xf option multiple times. If you specify the same attribute multiple times, the last value specified for the attribute is used.

**Examples:**

- `aoprxf.so -xf "fail-on-datastream-error = yes"`
- `aoprxf.so -xf "fail-on-datastream-error=yes trailer-error-page=no"`
- `aoprxf.so -xf fail-on-datastream-error=yes -xf trailer-error-page=no`
### Transform filter options

<table>
<thead>
<tr>
<th>option</th>
<th>Filter descriptions for for aoprfx.so</th>
</tr>
</thead>
<tbody>
<tr>
<td>option</td>
<td>Any option that the transform accepts. Separate multiple options with a space. For information about the options, see Infoprint Transform Manager for Linux: Using the transforms.</td>
</tr>
</tbody>
</table>

**Note:**

- Infoprint Server does not check the syntax of these options. It passes them directly to the transform.
- Some of the Infoprint Transform Manager for Linux transform options are different from the transform options you can specify with other Infoprint transforms. For example, the Infoprint Transform Manager for Linux -a option, which specifies the type of IOCA image to create, accepts different values. Also, the Infoprint Transform Manager for Linux transforms use different abbreviations for millimeters (m), inches (i), and pels (pels).

**Examples:**

```
ps2afp.dll -l 297mm -w 210mm -r 240 -a io1-g4
aoprfx.so -l 297m -w 210m -r 240 -a IO1_G4
```

- The Infoprint Transform Manager for Linux transforms, by default, create AFP records that are 32K bytes in length. To set the maximum number of bytes in a record to 8K, specify this -pragma option:
  ```
  -xf -pragma afpdsrecordlength=8000
  ```

  To create 8K records, you must specify 8000, which is actually less than 8K. This is because the length of the actual AFP records is greater than the value specified in the -pragma option.
Transform UNIX command parameters

This appendix describes the z/OS UNIX transform command parameters when using the transform commands.

The following transform commands are described:

- Infoprint Transforms to AFP for z/OS
  - pcl2afp command for PCL to AFP transform
  - pdf2afp command for PDF to AFP transform
  - ps2afp command for PDF to AFP and PS to AFP transforms
  - sap2afp command for SAP to AFP transform
- Infoprint Transform for AFP to HP PCL for z/OS
  - afp2pcl command for AFP to PCL transform
- Infoprint Transform for AFP to Adobe PDF for z/OS
  - afp2pdf command for AFP to PDF transform
- Infoprint Transform for AFP to Adobe PostScript for z/OS
  - afp2ps command for AFP to PS transform
- XML transform commands
  - xml2afp command for XML to AFP transform
  - xml2pdf command for XML to PDF transform
F.1 Transform commands for z/OS UNIX

Some Infoprint transform products provide z/OS UNIX transform commands. For example, Infoprint Transform for AFP to Adobe PDF for z/OS provides the `afp2pdf` command, which transforms data from AFP to PDF format. There are transform commands in z/OS UNIX that transform data streams from and to AFP format.

F.1.1 z/OS UNIX transform commands to AFP format

This section describes the z/OS UNIX transform commands and the parameter options that you can use with Version 2 Release 1 (V2.1) of IBM Infoprint Transforms to AFP for z/OS (program number 5655-N60). Transform commands let you convert files to an AFP data stream without printing. The transform commands can write the output AFP data stream to a UNIX file or to an MVS data set.

These transforms let you print non-AFP data on IBM AFP printers, also known as Intelligent Printer Data Stream (IPDS) printers. You can transform documents to AFP format from:

- HP Printer Control Language (PCL)
- Adobe Portable Document Format (PDF)
- Adobe PostScript
- SAP R/3 System Generic Output Format (SAPGOF)

The following transform commands and the parameters are described in Table F-1 on page 258 and are shown as follows:

- `pcl2afp [-a imagetype] [-c transformclass] [-o outputfile] [-p pagerange] [-t outputtype] [inputfile ...]`
- `sap2afp [-st] [-o outputfile] [-p pagerange] [-r resolution] [inputfile ...]`

<table>
<thead>
<tr>
<th>Keyword parameter</th>
<th>Description</th>
<th>Transform Command</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>-a imagetype</code></td>
<td>Determines the type of AFP data stream image to generate for each page in the file.</td>
<td><code>pcl2afp</code> <code>pdf2afp</code> <code>ps2afp</code> <code>sap2afp</code></td>
</tr>
</tbody>
</table>
### Keyword parameter Description

<table>
<thead>
<tr>
<th>Keyword parameter</th>
<th>Description</th>
<th>Transform Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>-c transformclass</td>
<td>Specifies the name of a transform class that identifies an entry in the aopxfd.conf file. Options that the transform class determines are:</td>
<td>pcl2afp, pdf2afp, ps2afp</td>
</tr>
<tr>
<td></td>
<td>- The length and width of the generated image</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- The page margins</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- The resolution of the output image</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- The amount of memory that the transform allocates</td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> For the pcl2afp transform, if the transform class specifies a resolution that the printer does not support, PSF prints the image under most conditions, but with degraded results.</td>
<td></td>
</tr>
<tr>
<td>-o outputfile</td>
<td>Specifies the output path and file into which the transform output (that is, AFP data) is written. The transform overwrites any existing data in the output file. If you specify more than one output file, the last path and file name are used. If you do not specify an output file, the result is written to standard output (STDOUT).</td>
<td>pcl2afp, pdf2afp, ps2afp</td>
</tr>
<tr>
<td></td>
<td>To specify an MVS data set, such as a sequential or partitioned data set, precede the data set name with //. When you specify a fully-qualified name, two sets of quotation marks are required. For example, &quot;//hlq.PDS(MYDOC)&quot; or &quot;//hlq.SEQDS&quot;. When you specify a not fully-qualified name, you only need one set of quotation marks. For example, &quot;//PDS(MYDOC)&quot; or &quot;//SEQDS&quot;. (TSO/E data set naming conventions.)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>If you specify an MVS data set, you might need to allocate and catalog the data set before you run this command. Allocate a data set that is large enough to hold the output data stream. The size of the output data stream depends on the complexity of the document and the type of image compression you select in the -a option. Typically, an output AFP data stream is several times as large as the input data stream. The MVS output data set's record format is VBM and record length 8K (8192) bytes or larger.</td>
<td></td>
</tr>
<tr>
<td>-p pagerange</td>
<td>Specifies that the output should contain only selected pages. The -p option counts pages by their actual sequence in the document, not by page number. For example, to write only the last page of a document whose pages are numbered i, ii, 1, 2, 3, 4, specify -p 6. Examples are:</td>
<td>pcl2afp, pdf2afp, sap2afp</td>
</tr>
<tr>
<td></td>
<td>-p even - Write even pages or -p odd - write odd pages</td>
<td></td>
</tr>
<tr>
<td></td>
<td>-p 1-10 specifies first through tenth page to be written</td>
<td></td>
</tr>
<tr>
<td></td>
<td>-p 10- requests pages from the tenth page until the end of the file.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>-p 1, -p 3, -p 6 - Write the first, third, and sixth pages</td>
<td></td>
</tr>
<tr>
<td>-t outputtype</td>
<td>Determines the type of output to generate:</td>
<td>pcl2afp, pdf2afp, ps2afp</td>
</tr>
<tr>
<td></td>
<td>- document - Printable document.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- overlay - Graphic image that can be printed on each page of a printable document.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- pagesegment - Graphic image that can be embedded in a printable document.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>When generating overlays or page segments from multiple-page documents, use the -p option to select a single page.</td>
<td></td>
</tr>
<tr>
<td>inputfile</td>
<td>Specifies an input file to be transformed. If you specify more than one input file name, the transform command concatenates the files. The results are written to a single output file or data set (if one is specified) or to standard output. If you do not specify an input file, or if you specify a dash (--) as the file name, the transform uses standard input.</td>
<td>pcl2afp, pdf2afp, ps2afp</td>
</tr>
</tbody>
</table>
F.1.2 z/OS UNIX transform commands from AFP format

This section describes the z/OS UNIX transform commands and the parameter options that you can use. The transform commands let you convert files from an AFP data stream without printing. The transform commands can also write the output data stream to a UNIX file or to an MVS data set.

<table>
<thead>
<tr>
<th>Keyword parameter</th>
<th>Description</th>
<th>Transform Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>-l length</td>
<td>Specifies the length of the generated image. In general, specify the length of the physical page.</td>
<td>pdf2afp, ps2afp</td>
</tr>
<tr>
<td>-r resolution</td>
<td>Specifies the resolution of the output image. Select the correct resolution for the printer on which you intend to print the image. If you specify a resolution that the printer does not support, PSF prints the image under most conditions, but with degraded results.</td>
<td>pdf2afp, ps2afp, sap2afp</td>
</tr>
<tr>
<td>-w width</td>
<td>Specifies the maximum width of the generated image. In general, specify the width of the physical page.</td>
<td>pdf2afp, ps2afp</td>
</tr>
<tr>
<td>-x xmargin</td>
<td>Specifies a horizontal margin or border around the generated image to avoid the non-printable areas of some printers. Note: 1. Because the X value specifies margins on both the left and right sides of the page, the X value can be no more than half of the width (-w) of the generated image. For example, if you specify a width of 8 inches, the X value can be no larger than 4 inches. If you specify an X value of 5 inches, a blank page is printed because the sum of the left and right margins exceeds the width of the paper. 2. The X value does not shift the image on the page. The image is cropped if it is defined to print in the left or right margin.</td>
<td>pdf2afp, ps2afp</td>
</tr>
<tr>
<td>-y ymargin</td>
<td>Specifies a vertical margin or border around the generated image to avoid the non-printable areas of some printers. Note: 1. Because the Y value specifies margins on both the top and bottom of the page, the Y value can be no more than half of the length (-l) of the generated image. For example, if you specify a length of 12 inches, the Y value can be no larger than 6 inches. If you specify a Y value of 7 inches, a blank page is printed because the sum of the top and bottom margins exceeds the length of the paper. 2. The Y value does not shift the image on the page. The image is cropped if it is defined to print in the top or bottom margin.</td>
<td>pdf2afp, ps2afp</td>
</tr>
<tr>
<td>-i initializationfile</td>
<td>Specifies an ASCII PostScript file that is added to the start of the job to set up and initialize the PostScript transform. If you specify the -i option multiple times, the initialization files are processed in the order that you specify them.</td>
<td>ps2afp</td>
</tr>
<tr>
<td>-st</td>
<td>s suppresses Graphic Object Content Architecture (GOCA) boxes. Some older printers do not print these boxes. t requests a trace. Specify this option only if instructed by IBM service personnel. For information about trace option, see z/OS Infoprint Server Messages and Diagnosis, G544-5747. You can use the filter-options job attribute with, for example, the lp command to pass any of these options except -o outputfile to the transform</td>
<td>sap2afp</td>
</tr>
</tbody>
</table>
The following transform commands and the parameters are described in Table F-2 on page 261 and are shown as follows:

```
afp2pcl [-c transformclass] [-F tracefile] [-i inputcodepage] [-j jobattributes] [-o outputfile] [-T traceoptions] [inputfile ...]
afp2ps [-c transformclass] [-F tracefile] [-i inputcodepage] [-j jobattributes] [-o outputfile] [-T traceoptions] [inputfile ...]
```

<table>
<thead>
<tr>
<th>Keyword parameter</th>
<th>Description</th>
<th>Transform</th>
</tr>
</thead>
<tbody>
<tr>
<td>-c transformclass</td>
<td>Specifies the name of a transform class that your administrator has defined. The transform class determines options such as:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- The characteristics of the printer, such as whether it supports color</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- The size of paper in each input tray, such as letter, ledger, A4, B4, B5, or a custom paper size</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- Defaults for page formatting options, such as the default page definition, form definition, and font</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- Resource libraries</td>
<td></td>
</tr>
<tr>
<td></td>
<td>You do not always have to specify a transform class. If you do need to specify one, however, ask your administrator for the name of a transform class suitable for the printer and the type of job.</td>
<td></td>
</tr>
<tr>
<td>-F tracefile</td>
<td>Specifies the file in which to store the trace. This option should only be used as instructed by IBM service personnel. The -F option specifies the name of the file where the transform writes all messages and trace information. If you specify this option without the -T option, the transform writes all messages to this file, but no trace information. When you specify this option, the transform does not write messages to the output data set or to the stderr file. The file name can contain up to 120 characters and can include an extension. The file you specify must not already exist. If it exists, the transform fails.</td>
<td></td>
</tr>
<tr>
<td>-i inputcodepage</td>
<td>This option applies only when you transform line data. If you specify this option for AFP data, it is ignored. This option identifies the code page to which line data is converted before it is transformed. Specify a code page that corresponds to the coded fonts specified in the page definition or in the chars job attribute. To transform line data that is already encoded in the code page that corresponds to the coded fonts, do not specify this option. If this option is not specified, line data is not converted before it is transformed. For example, to transform a line data document that specifies coded fonts in the chars job attribute and currently prints correctly on an AFP printer, do not specify this option. You must specify this option to correctly transform documents encoded in code pages that do not correspond to the code page for the coded fonts. This is most likely to occur when you transform an ASCII file. In the -i option, you must specify an IBM-supplied or custom code page that the iconv utility supports. When you specify this option, also make sure that the code page specified in the document-codepage job attribute correctly identifies the code page in which the input document is encoded. If you do not specify the document-codepage attribute, the default is the code page of the locale, which is usually an EBCDIC code page.</td>
<td></td>
</tr>
</tbody>
</table>
### -T traceoptions

The `-T` option specifies the type of trace you want. To request more than one type of trace, specify multiple options in parentheses, or specify the `-T` option multiple times. Allowed values are:

- **all**  
  All trace options (generates a lot of output)
- **allocate**  
  File and memory allocations
- **flow**  
  Program flow information
- **io**  
  Input output trace
- **trans**  
  Internal transform

**Example:** `-T (allocate io)`

**Default:** No tracing is done

You can use the filter-options job attribute with, for example, the `lp` command to pass the `-c transformclass` and `-i inputcodepage` options to the transform. For information about the filter-options job attribute, see Appendix, "AFP to PCL, AFP to PDF, and AFP to PostScript %filter-options" on page 248.

### -j jobattributes

Specifies an option, that is, one or more attribute value assignments in the format `attribute=value`, separated by spaces. You can specify `-j` multiple times. If job attributes are repeated, the last value specified for the attribute is used.

- If a value contains spaces, enclose the value in single or double quotation marks:
  ```
  attribute='value with spaces'
  attribute="value with spaces"
  ```
- If an option contains spaces or characters that might be interpreted by the shell (such as `$ & ( ) > < | ' "`), enclose the option in single or double quotation marks:
  ```
  -j 'attribute1=value1 attribute2=value2'
  -j "attribute='value with spaces'"
  -j "attribute=value(1)"
  ```
- If both value and option require quotation marks use two pairs of double quotation marks and place a backslash before each quotation mark that surrounds the value:
  ```
  -j "attribute="value with spaces"
  ```
- Use different quotation marks around the option and value. For example:
  ```
  -j 'attribute='value with spaces''
  -j "attribute='value with spaces'"
  ```

Instead of entering a string of attributes on the command line, you can store attributes and values in a file. You use a special attribute called `attributes` to specify the file. You can specify any of these attributes, which apply to all files to be transformed with the command:

- `carriage-control-type`
- `input-tray-number`
- `shift-out-shift-in`
- `chars`
- `output-bin-number`
- `table-reference-characters`
- `document-codepage`
- `overlay-back`
- `x-image-shift-back`
- `document-format`
- `overlay-front`
- `x-image-shift-front`
- `duplex`
- `page-definition`
- `y-image-shift-back`
- `form-definition`
- `resource-library`
- `y-image-shift(front)

**Note:** For information about the attributes, see `z/OS Infoprint Server User's Guide`, S544-5746.

<table>
<thead>
<tr>
<th>Keyword parameter</th>
<th>Description</th>
<th>Transform</th>
</tr>
</thead>
<tbody>
<tr>
<td>-T traceoptions</td>
<td>The <code>-T</code> option specifies the type of trace you want. To request more than one type of trace, specify multiple options in parentheses, or specify the <code>-T</code> option multiple times. Allowed values are:</td>
<td><code>afp2pcl</code> <code>afp2pdf</code> <code>afp2ps</code></td>
</tr>
<tr>
<td>-j jobattributes</td>
<td>Specifies an option, that is, one or more attribute value assignments in the format <code>attribute=value</code>, separated by spaces. You can specify <code>-j</code> multiple times. If job attributes are repeated, the last value specified for the attribute is used.</td>
<td><code>afp2pcl</code> <code>afp2pdf</code> <code>afp2ps</code></td>
</tr>
</tbody>
</table>
F.2 XML transform commands

The `xml2afp` and `xml2pdf` transform commands and the parameters are described in Table F-1 on page 258 and are shown as follows:

### Table F-3 Transform parameters from xml format

<table>
<thead>
<tr>
<th>Keyword parameter</th>
<th>Description</th>
<th>Transform</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>-o outputfile</code></td>
<td>Specifies the output path and file into which the transform output (that is, PCL data) is written. The transform overwrites any existing data in the output file. If you do not specify an output file, the result is written to standard output (STDOUT). To specify an MVS data set, such as a sequential or partitioned data set, precede the data set name with <code>//</code>. When you specify a fully-qualified name, two sets of quotation marks are required. For example, <code>&quot;//HLQ.PDS(MYDOC)&quot;</code> or <code>&quot;//HLQ.SEQDS&quot;</code>. When you specify a not fully-qualified name, you only need one set of quotation marks. For example, <code>&quot;//PDS(MYDOC)&quot;</code> or <code>&quot;//SEQDS&quot;</code>. If you specify an MVS data set, you might need to allocate the data set before you run this command, especially when you transform a large document. Allocate a data set that is large enough to hold the output data stream. The size of the output data stream depends on the complexity of the document. Allocate the output data set with these characteristics record format VB record length 1024 or larger is recommended.</td>
<td><code>afp2pcl</code> <code>afp2pdf</code> <code>afp2ps</code></td>
</tr>
<tr>
<td><code>inputfile</code></td>
<td>Specifies an input file to be transformed. If you specify more than one input file name, the <code>afp2pcl</code> command concatenates the files. The results are written to a single output file (if one is specified in -o) or to standard output. If you do not specify an input file, or if you specify a dash (-) for the file name, <code>afp2pcl</code> uses standard input. To specify an MVS data set, precede the data set name with <code>//</code>. When you specify a fully-qualified name, two sets of quotation marks are required. For example, <code>&quot;//HLQ.PDS(MYDOC)&quot;</code> or <code>&quot;//HLQ.SEQDS&quot;</code>. When you specify a partially qualified name, you only need one set of quotation marks. For example, <code>&quot;//PDS(MYDOC)&quot;</code> or <code>&quot;//SEQDS&quot;</code>.</td>
<td><code>afp2pcl</code> <code>afp2pdf</code> <code>afp2ps</code></td>
</tr>
<tr>
<td><code>-T</code></td>
<td>Requests a trace for diagnostic purposes. Trace output is appended to the HFS file, MVS data set, or SYSPUT data set that is specified in the STDERR DD statement.</td>
<td><code>xml2afp</code> <code>xml2pdf</code></td>
</tr>
<tr>
<td><code>-c transformclass</code></td>
<td>Indicates a transform class the administrator has defined in the transform configuration file, acopxfd.conf. The transform class is 1 to 63 characters, data including letters, numbers, or special characters. This option is only valid if the transform is managed by Infoprint Server. If no transform class is specified, the default for the transform is used.</td>
<td><code>xml2afp</code> <code>xml2pdf</code></td>
</tr>
<tr>
<td><code>-d destinationtype</code></td>
<td>Indicates the type of destination, which affects the output generated. For example, different types of destinations can support different font technologies and different color specifications. If the destination type is not specified, the default is outline fonts, no color, with 600 dpi resolution. Otherwise, valid values are those listed in the device types file, which determines the properties for a destination type. You can use the device types file supplied with XML Extender or your administrator can create a device types file. The sample device types file, that is shipped with XML Extender, is <code>/usr/lpp/xml2afp/samples/device_types</code>. If the transform is managed by Infoprint Server, the default is specified in the definition for the transform class.</td>
<td><code>xml2afp</code></td>
</tr>
<tr>
<td>Keyword parameter</td>
<td>Description</td>
<td>Transform</td>
</tr>
<tr>
<td>-------------------</td>
<td>-------------</td>
<td>-----------</td>
</tr>
<tr>
<td>-l pagelength[=in</td>
<td>Indicates the length of the page on which the output is printed. Specify a number</td>
<td></td>
</tr>
<tr>
<td>mm]</td>
<td>followed directly by the units in Inches (default) or mm Millimeters. If the</td>
<td></td>
</tr>
<tr>
<td></td>
<td>transform is managed by Infoprint Server, the default is specified in the definition</td>
<td></td>
</tr>
<tr>
<td></td>
<td>for the transform class.</td>
<td></td>
</tr>
<tr>
<td>-o outputfile</td>
<td>Specifies the file or MVS data set to which the AFP output is written. The</td>
<td></td>
</tr>
<tr>
<td></td>
<td>transform overwrites any existing data in the output file. If you do not specify an</td>
<td></td>
</tr>
<tr>
<td></td>
<td>output file name, or if you specify a dash (-), the output is written to stdout.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>To specify an MVS data set, such as a sequential or partitioned data set,</td>
<td></td>
</tr>
<tr>
<td></td>
<td>precede the data set name with //. For example: &quot;/PDS(MYDOC)&quot; or</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;/SEQDS&quot;. When you specify a fully-qualified name, you must use two sets of</td>
<td></td>
</tr>
<tr>
<td></td>
<td>quotation marks. For example: &quot;/hlq.PDS(MYDOC)&quot; or &quot;/hlq.SEQDS&quot;.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>If you specify an MVS data set, you might need to allocate and catalog the data</td>
<td></td>
</tr>
<tr>
<td></td>
<td>set before you run the transform command. The data set you allocate must be</td>
<td></td>
</tr>
<tr>
<td></td>
<td>large enough to hold the output data stream, whose size depends on the</td>
<td></td>
</tr>
<tr>
<td></td>
<td>complexity of the document. The output data set must have record format, (VBM</td>
<td></td>
</tr>
<tr>
<td></td>
<td>or VBA for xml2afp and VB for xml2pdf) and record length 32752 bytes.</td>
<td></td>
</tr>
<tr>
<td>s {stylesheet</td>
<td>Specifies whether a style sheet is used to transform XML data to XSL-FO data</td>
<td></td>
</tr>
<tr>
<td>none}</td>
<td>or whether a specified style sheet is suppressed when the transform is managed</td>
<td></td>
</tr>
<tr>
<td></td>
<td>by Infoprint Server.</td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>stylesheet</strong> - Specifies the file or URL for an XSL style sheet that is used to</td>
<td></td>
</tr>
<tr>
<td></td>
<td>transform XML data to XSL-FO data. The -s option should only be specified</td>
<td></td>
</tr>
<tr>
<td></td>
<td>when the input file contains XML data that must be transformed according to a</td>
<td></td>
</tr>
<tr>
<td></td>
<td>specified style sheet, not when the input file contains XSL-FO data.</td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>none</strong> - Indicates that a style sheet should be suppressed because it has been</td>
<td></td>
</tr>
<tr>
<td></td>
<td>specified with XSL-FO data. This value is only valid when the transform is</td>
<td></td>
</tr>
<tr>
<td></td>
<td>managed by Infoprint Server.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>When XML Extender is a standalone transform, it runs with the user's authority.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>When XML Extender is a managed transform, it runs with limited authority (as</td>
<td></td>
</tr>
<tr>
<td></td>
<td>determined by the system administrator), which usually means it can only</td>
<td></td>
</tr>
<tr>
<td></td>
<td>access unprotected data. Therefore, if the transform is managed by Infoprint</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Server, XML Extender must be able to access the file or resource named by the</td>
<td></td>
</tr>
<tr>
<td></td>
<td>style sheet URL.</td>
<td></td>
</tr>
<tr>
<td>-w pagewidth[=in</td>
<td>Indicates the width of the page on which the output is printed. Specify a number</td>
<td></td>
</tr>
<tr>
<td>mm]</td>
<td>followed directly by the units in Inches (default) or mm Millimeters. If the</td>
<td></td>
</tr>
<tr>
<td></td>
<td>transform is managed by Infoprint Server, the default is specified in the definition</td>
<td></td>
</tr>
<tr>
<td></td>
<td>for the transform class.</td>
<td></td>
</tr>
<tr>
<td>inputfile</td>
<td>Specifies one or more files or MVS data sets that are to be transformed to AFP</td>
<td></td>
</tr>
<tr>
<td></td>
<td>output. An input file contains either XML data with no presentation information</td>
<td></td>
</tr>
<tr>
<td></td>
<td>or XML data with XSL formatting objects. If you specify multiple input files with</td>
<td></td>
</tr>
<tr>
<td></td>
<td>a style sheet, each file is transformed according to the style sheet. Therefore,</td>
<td></td>
</tr>
<tr>
<td></td>
<td>IBM recommends that when specifying a style sheet, all input files contain XML</td>
<td></td>
</tr>
<tr>
<td></td>
<td>data only.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>If you do not specify an input file name, or if you specify a dash (-), the transform</td>
<td></td>
</tr>
<tr>
<td></td>
<td>command uses stdin for the input file. If you specify more than one input file</td>
<td></td>
</tr>
<tr>
<td></td>
<td>name, the xml2afp command writes the output to a single AFP data stream that</td>
<td></td>
</tr>
<tr>
<td></td>
<td>consists of multiple documents.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>To specify an MVS data set, such as a sequential or partitioned data set,</td>
<td></td>
</tr>
<tr>
<td></td>
<td>precede the data set name with //. For example: &quot;/PDS(MYDOC)&quot; or</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;/SEQDS&quot;. When you specify a fully-qualified name, you must use two sets of</td>
<td></td>
</tr>
<tr>
<td></td>
<td>quotation marks. For example: “/hlq.PDS(MYDOC)” or “/hlq.SEQDS”.</td>
<td></td>
</tr>
</tbody>
</table>
Related publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this book.

IBM Redbooks

For information about ordering these publications, see "How to get Redbooks" on page 265. Note that some of the documents referenced here may be available in softcopy only.

- ABCs of z/OS System Programming Volume 7, SG24-6987
- z/OS Infoprint Server Implementation, SG24-6234

Other publications

These publications are also relevant as further information sources:

- z/OS Infoprint Server Customization, S544-5744
- z/OS Infoprint Server Introduction, S544-5742
- z/OS Infoprint Server Messages and Diagnosis, G544-5747
- z/OS Infoprint Server Operation and Administration, S544-5745
- z/OS Infoprint Server User's Guide, S544-5746
- z/OS XL C/C++ Programming Guide, SC09-4765
- IBM Infoprint Transforms to AFP for z/OS Version 2.1, G550-0443
- IBM Infoprint Transforms from AFP for z/OS Version 2.1, G550-0444
- z/OS Language Environment Debugging Guide, GA22-7560
- z/OS MVS Installation Exits, SA22-7593
- z/OS MVS JCL Reference, SA22-7597
- z/OS Open Cryptographic Services Facility Application Programming, SC24-5899
- Data Stream and Objects Architectures: Graphics Object Content Architecture for Advanced Function Presentation Reference, S544-5498
- Infoprint XML Extender for z/OS, S544-5855
- IBM Infoprint XT Extender for z/OS Customization and Usage, S544-5879
- IBM Infoprint Manager: Reference, S544-5475

How to get Redbooks

You can search for, view, or download Redbooks, Redpapers, Technotes, draft publications and Additional materials, as well as order hardcopy Redbooks, at this Web site:

ibm.com/redbooks
Help from IBM

IBM Support and downloads
ibm.com/support

IBM Global Services
ibm.com/services
Index

Symbols
_BPX_SHAREAS environment variable
  AOPBATCH 159
/etc/Printsrv/aopd.conf 131, 140
/etc/Printsrv/aopsapd.conf 75
%filter-options 127
$HOME/.aopconf 140

Numerics
240-pel fonts 121
300-pel fonts 121

A
ABAP 75
Advanced Business Application Programming 75
Advanced Business Application Programming (ABAP) 149
AFP Presentation Text Object Content Architecture 75
AFP resource libraries 50
afp2pcl command 81, 150
afp2pdf comman 151
afp2pdf command 13, 89, 91, 95, 258
afp2ps command 154
AFPFIX REXX program 150
alias command 73
AOP_ENCRYPT 168
AOP_FONTMAP environment variable
  AFP to PCL transform 83
  AFP to PS transform 105
AOP_SAP2AFP_RESOURCES 140
AOPBATCH JCL procedure 159
remotexf command 47
AOPBATCH program 15, 24, 142, 157
AOPSTART JCL procedure 134
AOPCONF 140
aopd.conf configuration file
  start-daemons=xfd 31
aopd.conf file 32
AOPIPDRR panel fields 128
aopmsg.conf message configuration file 134
AOPPAPER table 117
AOPPRINT JCL procedure 15, 21–22, 142
attributes files 144
aopform.dll filter 27, 251
AOPRXF_CONF environment variable 29
aopxf.conf 28
aopxf.conf configuration file 47
aopxf.so filter 27
aopsapd.conf
  configuration file 76
aopsapd.conf configuration file 75
aopstart command 50, 72, 132
aopstart EXEC 29
AOPSTART JCL procedure 43, 51, 134
aopstop command 132–133
AOPSTOP JCL procedure 43, 134
aopxfd.conf 28, 32
  transform configuration file 32, 63, 67, 80, 86, 103,
    107, 116
aopxfd.conf configuration file
  AFP to PDF transform 86
aopxfd.conf file 43, 57
AOXCFF30 font-conversion program 121
AOXCFF30 program 121
  parameters 123
  return codes 123
ASCII code page 181
ASCII line-feed controls 127
ASCII PCL data streams
  NetSpool 5
attribute files 144

B
Bar Code fonts 122
Bar Code Object Content Architecture 235
barcode.tab configuration file 76
BCOCA 235
BPX.SUPERUSER
  FACILITY class 36
bpxwunix() 165

C
C++ run-time options 158
cancel command 141
chown command
  using the Password exiti 102

d

d daemon 14
DBCS fonts 66
DD JCL statement
  SUBSYS parameter 145
defcp.tab configuration file 76
DEST=IP parameter 176

E
EBCDIC line data
  NetSpool 5
environment variables
  Transform Interface 28
ERRCLASS parameter 22
Expanded Core Fonts feature 118
Expanded Core Fonts feature 118
pcl2afp command 147
pdf2afp command 147
PIDU 2, 130
pidu command 130
PJFORM keyword
  SAP data stream 76
PJL 10
Presentation Text Object Content Architecture 235
Print Interface 6
Print Interface subsystem 166–167
Printer Inventory 2
Printer Inventory Definition Utility 2
Printer Inventory Manager 2
printer job language 10
PRINTER variable 145
PRTATTRS parameter 19
ps2afp command 148
PTOCA 235
PTOCA data streams 75
R
Redbooks Web site 265
  Contact us xii
remotexf command 13, 27, 47
Resubmit for filtering 128
resubmit for filtering function 176
RFC 1179 6, 27
S
SAP ABAP data files 149
SAP Callback daemon 75
SAP OTF data files 149
SAP R/3 ABAP data streams 75
SAP R/3 data streams 78
SAP to AFP transform configuration files 76
SAP to AFP transform tasks 75
sap2afp command 140, 149
SAPI 3
sendmail 176
Server message block 6
SMB printing protocol 6
start-daemons = { xfd }
  Infoprint Server transforms 31
STDENV DD statement 159
stderr file 44
stderr files 44
STDOUT data set 22
su command 36, 78, 132, 134
SUBSYS JCL parameter 15, 167
SUBSYS parameter 145
SYS1.PROCLIB
  AOPPRINT JCL procedure 21
SYS1.SAMPLIB
  AOXCFS30J 121, 124
  AOXFONTS 118, 120
  AOXPAPER 117
SYSOUT Application Programming Interface 3
T
transform class 67
  transform entry 60
transform configuration file
  aopxfd.conf 32
transform daemons 56
transform filter
  aoprform.dll 114
transform filters 126
Transform Interface 26
transform products 26
transform XML to AFP 138
TSO/E LOGON panel
  size parameter 72
U
uconvdef command 78
uconvTable 78
universal-coded character sets 78
V
value pairs 144
X
Xerox files 26
Xerox line-conditioned data stream 114
Xerox metacode 139
Xerox print resources 139
xfd daemon 44
XFDFERR REXX program 44
XML 11, 138
XML data 109
XML Extender 24, 112
  execution mode 113
XML Extender for z/OS 109
XML files 110, 112
xml2afp command 24, 155
xml2pdf command 24, 157
XSL style sheet processor 113
XSL-FO files 113
Infoprint Server for z/OS
Infoprint Transforms

Infoprint Server Transforms

Customizing transforms and daemons

Using transforms

One component of Infoprint Server is the Transform Interface. This component communicates with transforms that IBM transform products provide. Transforms convert data from one format to another, for example, from PDF to AFP format, or from AFP to PCL format. Administrators can set up the transforms to automatically transform documents before they are printed. Users can also use the z/OS UNIX command line to transform documents, which can then be saved in the converted format and later printed or sent to other users.

This IBM Redbooks publication presents a comprehensive discussion of the Infoprint Server transforms and provides detailed steps for customizing and using them. It includes the following topics:

- An introduction to Infoprint Transforms
- How to customize the transforms
- How to customize the transform daemons
- How end users use the transforms

For more information: ibm.com/redbooks