Printing Systems Division

Using OpenType Fonts in an AFP System

G544-5876-02

Printing Systems Division

Using OpenType Fonts in an AFP System

G544-5876-02

Note
Before using this information and the product it supports, be sure to read the information in “Notices” on page 63.

Third Edition (July 2004)

Requests for IBM publications should be made to your IBM representative or to the IBM branch office serving your locality. If you request publications from the address given below, your order will be delayed because publications are not stocked there. Many of the IBM Printing Systems Division publications are available from the web page listed below.

Internet
Visit our home page at: `http://www.ibm.com/printers`

A Reader’s Comments form is provided at the back of this publication. If the form has been removed, you can send comments by fax to 1-800-524-1519 (USA only) or 1-303-924-6873; by E-mail to `printpub@us.ibm.com`; or by mail to:

IBM Printing Systems Division
Department H7FE Building 004N
Information Development
PO Box 1900
Boulder CO 80301-9191 USA

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

Contents

Figures

Tables

About this publication

This book explains how to install and reference TrueType® and OpenType® fonts in Microsoft® Unicode® format on systems that use the IBM® Advanced Function Presentation™ (AFP™) architecture to print or display data.

Audience

This book is for developers, planners, and print administrators in environments that use any of the applications listed in Chapter 9, "What AFP products have TrueType and OpenType font support?," on page 43 to generate or process AFP data.

Most recent information

The most recent product information (including updates of many publications), is found on the World Wide Web at both the Printing Systems Information Center and the IBM Printing Systems site (**http://www.ibm.com/printers**). See Table 1 to see the web addresses for the latter site that you should bookmark and consult frequently.

Table 1. Where to find information on the IBM Printing Systems web pages

Information	Web address (URL)
Printing Systems Information Center	http://publib.boulder.ibm.com/infocenter/printer/index.jsp **Note:** This URL may format across several lines. If you are using this source to copy into your Web browser, ensure that you join all lines together as a single string, without any spaces.
Infoprint® Manager for AIX® Version 4 Release 1	http://www.ibm.com/printers/ipmaixnew
Infoprint Manager for Windows® Version 2 Release 1	http://www.ibm.com/printers/ipmwinnew
Service updates for Infoprint Manager for AIX Version 4 Release 1	http://www.ibm.com/printers/ipmaixptfs
Service updates for Infoprint Manager for Windows Version 2 Release 1	http://www.ibm.com/printers/ipmwinptfs
Publications for Infoprint Manager for AIX Version 4 Release 1	http://www.ibm.com/printers/ipmaixlib
Publications for Infoprint Manager for Windows Version 2 Release 1	http://www.ibm.com/printers/ipmwinlib
Print Services Facility™ for OS/400® (PSF/400)	http://www.printers.ibm.com/internet/wwsites.nsf/vwwebpublished/psfhome_i_ww
Documentation for PSF/400	http://www.ibm.com/eserver/iseries/infocenter
PSF for z/OS®	http://www.printers.ibm.com/internet/wwsites.nsf/vwwebpublished/psfhome_z_ww
Publications for PSF for z/OS	http://www.printers.ibm.com/R5PSC.NSF/Web/psfm

Table 1. Where to find information on the IBM Printing Systems web pages (continued)

Information	Web address (URL)
AFP Windows Driver	http://www-1.ibm.com/support /all_download_drivers.html
AFP Conversion and Indexing Facility (ACIF)	http://www.printers.ibm.com/R5PSC.NSF /Web/afpm
Page Printer Formatting Aid (PPFA)	http://publib.boulder.ibm.com/infocenter /printer/index.jsp; select **AFP Products —> Page Printer Formatting Aid User's Guide**
Infoprint Fonts for Multiplatforms	http://publib.boulder.ibm.com/infocenter /printer/index.jsp; select **Infoprint fonts**
AFP fonts	http://www.printers.ibm.com/R5PSC.NSF/Web /font
IBM code pages	http://www-1.ibm.com/servers/eserver /iseries/software/globalization /codepages.html
AFP Viewer Plug-in	http://www.printers.ibm.com/pbin-afp/go? /pdocs/fldu3mst.html
IBM printers	http://www.printers.ibm.com/internet /wwsites.nsf/vwwebpublished/pselect_ww

Chapter 1. What are TrueType and OpenType fonts?

The TrueType font (TTF) technology was developed by Apple and Microsoft in the late 1980s and early 1990s. Apple first included TrueType font support in its Macintosh operating systems in 1990; Microsoft first included TrueType font support in its Windows operating system in 1991. The TrueType format is based on scalable outline technology with flexible hinting. Mathematically, TrueType shapes are based on quadratic curves; this is in contrast to Adobe™ Type 1 outlines which are based on cubic curves. Due primarily to the extensive operating system support, TrueType is the most prevalent font technology in the printing industry today. TrueType is an open font standard and is widely published. The technology is described in the following documents available from the Microsoft and Apple web sites:

- *OpenType Specification* Version 1.4 (Microsoft Corporation: October 11, 2002), at **http://www.microsoft.com/typography/otspec/default.htm**
- *TrueType Reference Manual* (Apple Computer, Inc.: December 18, 2002), at **http://developer.apple.com/fonts/TTRefMan/index.html**.

The OpenType font (OTF) format is an extension of the TrueType font format that allows better support for international character sets and broader multi-platform support. OpenType defines layout tables that can be used to carry the formatting information needed to fully support Unicode complex text, which includes bidirectional text, text that requires contextual shaping, and text with combining characters. These tables contain the script-specific information on glyph substitution, glyph positioning, composition and decomposition, and justification that is required to render Unicode complex text..

Additionally, the OpenType format settles the TrueType versus Type 1 competition by allowing either TrueType or Adobe Type 1 outlines to be packaged as an OpenType font. The OpenType font format was developed jointly by the Adobe and Microsoft Corporations. It is described in the *OpenType Specification*.

Typefaces

Due to the wide acceptance of the TrueType/OpenType font technology on consumer platforms, such as Windows and Macintosh, many applications and tools are available for using TrueType and OpenType fonts. This has led to the availability of a large number of TrueType and OpenType typefaces and continues to encourage the development of additional typefaces. Some of the major font foundries and their typefaces are:

- Adobe (Adobe Caslon®, Adobe Garamond®)
- Agfa® Monotype™ (Arial®, Gill Sans®, Times New Roman®)
- Bigelow & Holmes (Lucida®)
- Fundicion Tipográfica Neufville (Futura®)
- Haas Typefoundry Ltd. (Univers®)
- ITC® (ITC Garamond®, ITC Zapf Dingbats®)
- Linotype-Hell AG (Frutiger™, Helvetica™, Optima™, Palatino™)

Encodings

Character data in computer files, such as print files, is represented as a sequence of hexadecimal bytes (*code points*) in accordance with some predefined encoding scheme. Legacy encoding schemes that have been used since the advent of computing are EBCDIC and ASCII. In scripts, such as Latin, that use a limited set of characters, a single-byte encoding scheme is normally sufficient. In far-eastern scripts, such as Japanese, a double-byte encoding scheme is required. In IBM system environments, the encoding scheme is precisely identified with a code page that maps the hexadecimal code points to graphic character global identifiers (GCGIDs). The code page is identified either with a code page name or with a code page global identifier (CPGID).

Unicode is the new universal standard that defines a single encoding scheme to represent all of the characters used in all of the world's scripts.

It is important to keep in mind that in document presentation, two encoding schemes need to be dealt with. One is the encoding scheme used in the user data or the print data. The other is the encoding scheme tied to the font that is to be used to present the user data. Ideally the two encoding schemes are the same; in cases where they are not, an encoding conversion needs to take place.

Legacy encoding schemes, such as EBCDIC and ASCII, are very much language dependent. An application written in Germany may use an EBCDIC code page that includes mappings for the characters *ü* (u-umlaut) and *ä* (a-umlaut). An application written in France may use an EBCDIC code page that includes mappings for the characters *é* (e-aigu) and *è* (e-grave). A multinational application that needs to deal with both German and French text must be aware of the language for a particular string and needs to tie this string to the appropriate code page when the text is rendered.

The Unicode standard was developed to address the problems with legacy encodings and language-dependent code pages. It is a universal character encoding that includes all of the major scripts of the world. The primary encoding for Unicode characters is the Unicode Transformation Format UTF-16. This is a double-byte encoding with a scheme to use a pair of double-byte codes to access additional characters. This is also one of the major encodings supported in TrueType and OpenType fonts, and it forms the basis for the TrueType and OpenType support in Advanced Function Presentation (AFP) systems.

The equivalent of a code page in a TrueType font (TTF) is a character map (cmap). A cmap defines the mapping of code points to glyph indices, which are used to index the actual character shape information. A TTF may support multiple encodings, each of which is defined by a subtable in the cmap. A subtable is selected based on two parameters: platform ID and platform-specific encoding ID. Only the Microsoft Unicode subtables are supported for TrueType fonts and OpenType fonts in AFP environments, and this encoding is identified by platform ID = 3 (Microsoft), and platform-specific encoding ID= 1 (Unicode, UTF-16). In general, a TrueType or OpenType font that is to be installed and referenced in an AFP system must have the following characteristics:

1. It must contain a Microsoft Unicode subtable identified by platform ID = 3 (Microsoft) and platform-specific encoding ID = 1 (Unicode, UTF-16).
2. It must specify a full font name (Name ID 4) using the same encoding in the naming table.

Such fonts are referred to as *Unicode-enabled*.

The TTF tables used to access a specific cmap subtable are shown in Figure 1.

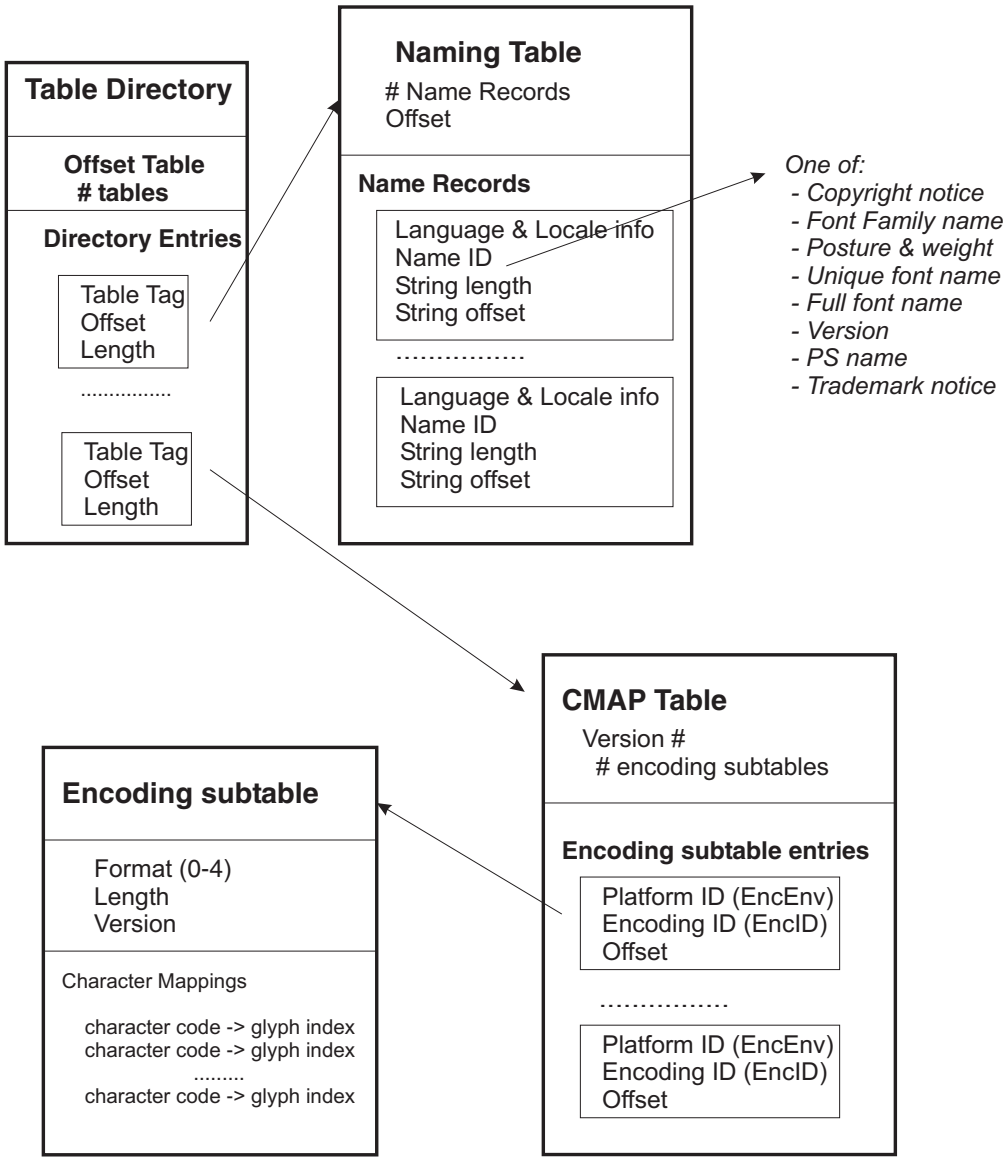


Figure 1. TrueType and OpenType font tables: encodings

Packaging

A TrueType font is described using a set of tables packaged in a TrueType font file. Such files normally have a file extension of **.ttf**. For backward compatibility, OpenType fonts that are based on TrueType outlines also use the file extension **.ttf**. OpenType font files that are based on Adobe Type 1 outlines often have the file extension **.otf**. The tables in the font file are used by a TrueType/OpenType rasterizer to render the glyphs in the font. The structure of the TrueType and OpenType font files and their tables are defined in the TrueType and OpenType Font Specifications.

A TrueType font may also be packaged in a font collection file called a TrueType Collection (TTC). Fonts in such collections can share tables and glyphs; therefore, the TTC file size can be significantly smaller than the sum of the individual font files.

Chapter 2. What is Unicode?

The Unicode Standard defines a universal character encoding that is designed to include the characters of all of the major scripts in the world. It covers phonetic scripts like Latin and ideographic scripts like Chinese and Korean. AFP support for Unicode is based on Unicode Version 3.2, a minor revision published on the web at **http://www.unicode.org/unicode/reports/tr28/tr28-3.html** in March 2002. The previous major revision was Unicode Version 3.0, which was published in hard copy in January 2000.[1]

Unicode Version 3.2 defines over 94,000 characters, but has the ability to encode more than 1 million characters. For each character, the Unicode Standard defines a numeric value, a character name, and other character properties, such as case and direction.

Additional information can be found in the Unicode 3.0 Standard, the Unicode 3.2 Standard, and at the Unicode web site: **http://www.unicode.org/**.

Unicode planes

The Unicode encoding space consists of 17 planes with each plane consisting of 65,536 code points. The addressing for this space is expressed in terms of Unicode scalar values, which are denoted as U+*xxxxxx*, and which span the range U+000000 to U+10FFFF. Unicode defines simple algorithms that relate the two main Unicode character encodings (UTF-8 and UTF-16) to Unicode scalar values. Plane 0 is called the *basic multilingual plane* (BMP) and is addressed with two-byte UTF-16 code points in the range U+0000 to U+FFFF. Planes 1–16 are called the *supplemental planes* and are addressed with UTF-16 surrogates.

UTF-16 *surrogates* consist of a sequenced pair of two-byte codes. The high-order code is called the high surrogate and must be in the range U+D800–U+DBFF. The low-order code is called the low surrogate and must be in the range U+DC00–U+DFFF. Each of the two-byte surrogate codes span a range of X'400' = 1024 codes, and therefore, the surrogate pairs span a range of 1024 × 1024 = 1,048,576 code points, which covers the 16 supplemental planes.

Scripts

The Unicode BMP and supplemental planes include the characters of all of the major scripts in the world. Table 2 shows how characters are grouped in the BMP.

Table 2. Character groupings in the Unicode BMP

Name	Unicode Character Code Range	Description
General Scripts	0000–1FFF	Phonetic scripts consisting of relatively small character sets; examples are Latin, Greek, Middle Eastern scripts, Thai

1. The current major revision of the standard is Unicode Version 4.0, which was published in hard copy in August 2003. For differences between Version 4.0 and Version 3.0, see **http://www.unicode.org/versions/Unicode4.0.0/appD.pdf**.

Table 2. Character groupings in the Unicode BMP (continued)

Name	Unicode Character Code Range	Description
Symbols	2000–2DFF	Symbols and dingbats that are used in punctuation, mathematics, science, and so on.
CJK Phonetics and Symbols	2E00–33FF	Symbols used in Chinese, Japanese, and Korean (CJK)
CJK Ideographs	3400–9FFF	Unified CJK ideographs
Yi Syllables	A000–A4CF	Yi syllables and radicals
Hangul Syllables	AC00–D743	Precomposed Korean Hangul syllables
Surrogates	D800–DFFF	1024 high-order surrogates and 1024 low-order surrogates
Private Use	E000–F8FF	Private use characters
Compatibility and Specials	F900–FFFD	Characters from other standards

Transformation formats

Unicode defines two primary encoding forms: a default two-byte form called UTF-16, and a one- to four-byte ASCII-compatible form called UTF-8, where UTF stands for Unicode Transformation Format. In the latter form, the base ASCII code points X'00'–X'7F' are represented unchanged as one-byte UTF-8 codes. The UTF-16 form supports a 4-byte encoding using the surrogate mechanism. It can support the encoding of up to one million additional characters.

UTF-16

UTF-16 is the default Unicode encoding format. Character codes for characters in the BMP map directly to UTF-16 codes, with the exception of the surrogate range (X'D800'–X'DFFF'). Code values in the surrogate range do not represent characters, since high and low surrogates must be combined into a pair to address a character in planes 1–16. Therefore, UTF-16 values in the surrogate range do not represent characters.

The relationship between Unicode scalar values and UTF-16 values is as follows:

- **Unicode scalar value = UTF-16 value** for codes outside the surrogate block
- **Unicode scalar value = (H – X'D800') × X'400' + (L – X'DC00') + X'10000'** for codes inside the surrogate block, where H is the high-order surrogate code (in the range X'D800'–X'DBFF'), and L is the low-order surrogate code (in the range X'DC00'–X'DFFF').

The byte order in AFP environments is always big-endian, which means that within a two-byte UTF-16 code point the high-order byte is specified first, followed by the low-order byte, and within each byte the high-order bit is specified first, followed by the low-order bits. In a surrogate pair, the high-order surrogate is specified first, followed by the low-order surrogate.

UTF-8

UTF-8 is an ASCII-compatible encoding that encodes the Unicode code values in a sequence of 1 to 4 bytes. ASCII transparency is achieved by mapping the ASCII range of the Unicode standard (U+0000 to U+007F) to the single-byte UTF-8 values X'00' to X'7F'. For example, the ASCII code X'41', which represents the Latin

character A and is assigned the Unicode value X'0041', is mapped to the UTF-8 value X'41'. For the detailed algorithm that describes the relationship between UTF-8, Unicode scalar values, and UTF-16, see the Unicode 3.2 Standard on the Unicode web site at http://www.unicode.org/unicode/reports/tr28/tr28-3.html.

Presentation semantics

In addition to being a universal standard for character encoding, the Unicode Standard also deals with text presentation to a limited extent. This is done through the definition of special characters that have presentation semantics, such as line and paragraph separators, and through the definition of the Unicode bidi algorithm that is applied to bidirectional text like Arabic and Hebrew. The current version of Unicode support in the AFP architecture is limited to one-to-one rendering of Unicode code points and assumes that all presentation semantics are defined by the controlling environment, which is AFP. All Unicode code points, including control characters and special characters such as byte-order marks, are treated as printable characters. If the active font does not contain a glyph for these characters, the "undefined/missing character" glyph will be printed, which is normally the glyph that corresponds to glyph index 0 in the TrueType or OpenType font.

When Unicode text is generated, it is stored in *logical* order, also called *storage* or *memory* order. This correlates roughly to the order in which the characters that make up the text would be typed in on a keyboard. This may be different than the order in which the characters are rendered, which is called the *display* order in Unicode.

For example, assume that four Unicode characters are stored in logical order as `u1 u2 u3 u4`. If they are Latin characters, their display order is `u1 u2 u3 u4`. However, if they are Arabic characters, their display order is `u4 u3 u2 u1`.

Chapter 3. Why was the AFP architecture extended to support TrueType and OpenType fonts?

The new support for TrueType and OpenType font technology in AFP provides customers with significant benefits and is a key component of the strategy to embrace emerging standards and *de facto* standards in the AFP architecture. The primary reasons for making this extension to the AFP architecture include:

- Providing customers with more choices for typefaces, particularly non-Latin typefaces
- Providing a truly multilingual presentation environment through support of Unicode
- Allowing customers to migrate towards a single font technology across all presentation environments
- Providing the basis for future, more advanced support of complex non-Latin scripts and typefaces

Supporting more industry-standard typefaces

Application development on the Windows platform has led to and continues to lead to the development of new typefaces in the TrueType and OpenType formats. In addition, emerging script requirements, such as the Chinese GB18030 requirement, cause the development of new fonts in these formats. By adding support for TrueType and OpenType fonts, the AFP architecture has enabled AFP applications to use a large set of typefaces that were previously unavailable to these applications, and has also enabled these applications to better participate in global markets that have requirements for extended character set support.

Supporting Unicode

Most TrueType and OpenType fonts contain cmap subtables for the UTF-16 encoding. This means that most TrueType and OpenType typefaces can be used by presentation data that is encoded in UTF-16. This differs from AFP Font Object Content Architecture (FOCA)-based fonts, where the encoding is either ASCII or EBCDIC. Only the FOCA fonts available with the AFP Unicode Migration Fonts RPQ (PRPQ 8A8087, 8A8090) support UTF-16 encoding in user presentation data. Therefore, the AFP extensions to support TrueType and OpenType fonts represent a significant step forward in the support of the Unicode Standard and support of Unicode encodings in presentation data.

Providing consistent font support across multiple presentation platforms

The TrueType/OpenType font format is supported by the majority of presentation platforms, including PostScript, PDF, PCL, Windows, and Macintosh. By integrating such a widely-used font format into AFP, customers can now move strategically to a single font format across all of their presentation environments. This leads to:

- Better presentation consistency since it eliminates the need for font substitution
- Lower costs for font purchase, installation, and maintenance
- Less education for technical professionals

Providing a basis for Unicode complex text support

Unicode complex text includes bidirectional text, text that requires contextual shaping, and text with combining characters. Unicode complex text cannot be rendered with traditional one-code-point to one-glyph rendering; it requires analysis and processing of runs of text.

OpenType fonts allow the Unicode support to be taken one step further than what is enabled with TrueType fonts. The OpenType layout tables contain the script-specific information on glyph substitution, glyph positioning, composition and decomposition, and justification that is required to render Unicode complex text. Therefore, support of OpenType fonts enables AFP systems for future support of Unicode complex text.

Chapter 4. How are TrueType and OpenType fonts installed on an AFP system?

To install TrueType and OpenType fonts, you use the Font Installer for AFP Systems to add the fonts to resource libraries and update the resource access table (RAT), which allows access to the resource libraries.

Note: OS/400 supports an alternative installation method for migration purposes: copying the fonts to /QIBM/UserData/OS400/Fonts/TTFonts. This method does *not* support font linking, font capture, or verification that the fonts are Unicode-enabled. To take advantage of the full power of TrueType and OpenType fonts, use the Font Installer for AFP Systems.

Resource libraries

TrueType and OpenType fonts are installed into *resource libraries*. Resources in these libraries are accessible to the print servers on that platform, as well as to other applications that run on the platform, such as document formatters. This is particularly important with fonts since the best text fidelity is achieved when both the formatter and the printing system use the same fonts and font metrics. Resource libraries can have platform-specific characteristics and names:

- On z/OS, they are path libraries in either a z/OS UNIX® System Services file system (USS) or a Hierarchical File System (HFS); they cannot be partitioned data sets.
- On OS/400, they are folders in the Integrated File System (IFS):
  - /QIBM/ProdData/OS400/Fonts/TTFonts contains fonts shipped with the operating system (option 43, Additional Fonts).
  - /QIBM/UserData/OS400/Fonts/TTFonts contains fonts installed by users.
- On AIX, they are directories in the file system.
- On Windows, they are folders in the file system.

Font Installer for AFP Systems

TrueType and OpenType fonts are installed using an IBM-provided utility program called the Font Installer for AFP Systems, which is an optional feature of the Infoprint Fonts for Multiplatforms product. Font Installer for AFP Systems runs on Windows systems and can install fonts in resource libraries for print servers on z/OS, OS/400, AIX, and Windows systems. Part of the installation process is the generation or update of a RAT for the resource library. This table contains an entry for each installed font and provides a mapping from the full font name of the font to the file name of the font and to other parameters needed to process the font. The RAT is used by print servers to access the font file when a reference to the font is specified in the AFP (MO:DCA-P) print file. The font installation process consists of the following parts:

1. Physically adding the font to the resource library
2. Updating the library's resource access table (RAT) with an entry for the font
3. Generating an object identifier (OID) for the font
4. Setting permissions for the font

The Font Installer installs only *Unicode-enabled* TrueType and OpenType fonts. These are fonts that:

- Contain a platform ID (PID) of 3 (Microsoft)
- Use an encoding ID (EID) of 1 (Unicode, UTF-16) for the cmap subtable and the full font name, specified as the name ID (NID)

Attempts to install a TrueType or OpenType font that is not Unicode-enabled result in an error message, and the font is not installed. To determine if a font is Unicode-enabled, see Appendix B, "Is the font Unicode-enabled?," on page 55.

Font Installer allows the user to install fonts from local folders or from media such as CDs on local systems to resource libraries on local and remote hosts. A resource library consists of a number of font files, such as TrueType Font files, OpenType Font files, and TrueType Collection files, together with the RAT. The resource library may also contain non-font resources or font resources that are not reflected in the RAT. The Font Installer contains an FTP client to work with libraries on remote systems. The Font Installer also supports a "verify" operation. That is, the Font Installer checks the contents of the library against the RAT and generates the following lists:

- Fonts that are in the library and that are installed in the RAT
- Fonts that are in the library but are not installed in the RAT
- Fonts that are not in the library but are in the RAT

Resource access table (RAT)

The resource access table (RAT) is used to map a resource name specified in the MO:DCA™ data stream to information used to find and process the resource on a given system. For TrueType and OpenType fonts, the resource name is a full font name. It is specified in the data stream in a Map Data Resource (MDR) structured field.

The Font Installer updates the RAT whenever it installs new fonts in the resource library. Whenever fonts already in the library and in the RAT are updated, such as when a new version of a font replaces an existing version, they should be reinstalled. The installed RAT remains active until it is updated or replaced. If no RAT is present in a library, the TrueType and OpenType fonts in that library cannot be processed. The RAT resides in the library that it represents. There can be multiple RATs in a system, one for each library, and the file names in the RAT do not contain path information. In AFP systems, the file name of the RAT in a given library is hard-coded as **IBM_DataObjectFont.rat**.

RAT structure

The RAT is an architected MO:DCA object; for a formal definition, see the *Mixed Object Document Content Architecture Reference*. The table consists of a table header followed by zero or more variable-length repeating groups. The table header specifies information that applies to the whole table including an identifier for the table, length of the table, and a time stamp indicating when the table was created or updated. The table header may also optionally contain information on the Font Install application that generated the RAT and the service level of that application.

A RAT repeating group consists of a header followed by zero or more variable-length triplet-like structures called table vectors. Each repeating group specifies the information needed to access and process a specific resource.

The repeating group content including the table vector definition is defined by the resource object type, which is identified by the resource object-type OID. Repeating groups for a specific resource object type, such as repeating groups for TrueType

and OpenType fonts, have the same syntax. Only a single repeating group is allowed for a specific resource object. That is, a single resource object may only be defined and indexed once in a RAT.

Table vectors in a repeating group are defined to be either mandatory or optional, and their data fields are defined to be either mandatory or optional. If an optional data field in a table vector is omitted, the table vector is said to be empty. In a given repeating group, all mandatory table vectors must be specified, although the vectors can be empty if allowed.

RAT content

A RAT repeating group for TrueType or OpenType fonts contains the following information:

- The full font name encoded in UTF-16BE. This full font name is specified multiple times in all languages used in the font's naming table. The names are encoded in UTF-16, which matches the encoding defined by EncEnv = Microsoft (X'0003') and EncID = Unicode (X'0001') in the naming table.
- A mapping of the full font name (in each language) to the name of the file that contains the font, encoded in UTF-16BE. For example, if the Naming Table contains two name records for the full font name (Name ID 4), one in English–United States (LCID = X'0409'), and one in German–Standard (LCID = X'0407'), both in the encoding defined by EncEnv = Microsoft (X'0003') and EncID = Unicode (X'0001'), the RAT must map both language versions of this full font name to the same file name.
- If the font is to be enabled for capture or can be resident in the printer, the repeating group also includes the OID for the font.
- If the font is contained in a TrueType Collection file (TTC), the full font name is mapped to the name of the TTC file, encoded in UTF-16BE, and to an index to the font in the TTC. If the collection has an OID assigned, the mapping also includes the OID for the collection. When a mapped TrueType or OpenType font is part of a TTC, the complete TTC (if not already in the presentation device) is downloaded to the device.
- If the font has linked fonts, the RAT links the full font name of the base font to the full font names of the linked fonts. Linked fonts are valid TrueType and OpenType fonts. All linked fonts for a base font are downloaded to the device and are treated as extensions to the font by the device if not already in the presentation device. The order in which the linked fonts are specified determines the order in which they are processed by the device.

Font collections

A TrueType Collection (TTC) is a file that consists of multiple TrueType and OpenType fonts. Significant file size reductions can be achieved if fonts that share large numbers of glyphs are packaged in a TTC. Conceptually, a TTC may be thought of as a file with a common header structure and multiple TrueType or OpenType fonts. The Font Installer for AFP Systems can install TTCs as well as TrueType and OpenType fonts. When a TTC is installed, the RAT contains a repeating group for each font in the TTC. Each repeating group is indexed with the full font name of the font and points to the file name of the TTC.

TTCs are never referenced in the MO:DCA data stream. This keeps the data stream independent of the packaging of the fonts. The MO:DCA data stream always references the TrueType or OpenType font by full font name, which the presentation system resolves to either a font file (TTF or OTF) or to a font in a collection file (TTC).

Font linking

TrueType and OpenType fonts can be linked to a TrueType or OpenType font (which is then called a base font) to form an ordered list of fonts that are essentially processed as a single font. When indexing such a list of fonts with a code point, the base font is always accessed first, followed by the first linked font, followed by the second linked font, and so on. The font linking function fulfills two primary requirements:

- Supports the ability to add user-defined characters to a given font. This requirement is particularly strong in Japanese, Simplified Chinese, and Traditional Chinese markets.
- Supports the ability to extend a font with additional characters. These are not user-defined characters but characters that did not fit into a single font due to the 64K character restriction for TrueType fonts. The most important example for this requirement is the extension of the base Chinese character set for GB18030 support.

Notes:

1. If characters in a font have to be replaced, you should create a new font containing the replacement characters and give it a different full font name from the original. Then install the new font as a base font and make the original base font the first link to the new base.
2. User-defined characters are often defined in fonts with a **.tte** extension. Because all TTE fonts have the same full font name, EUDC, only one TTE font can be installed in a font library or linked to base fonts in that library.
3. TTE fonts define character values for all code points in the Unicode Private Use Area (X'E000' through X'F8FF'); when no character is explicitly defined in this code point range, the code point defaults to a space character. If a TTE font is linked to a base TrueType or OpenType font, processing ends after the first TTE font because all code points have already been defined. Spaces are printed in place of characters at code points that are defined in subsequent linked fonts. You should therefore make the TTE font the last linked font.
4. Linked fonts should normally have typeface and metric characteristics that are similar to the base font. Since each character is rendered based on the parameters in its font, inconsistencies between base font and linked fonts may lead to inconsistent rendering, for example rendering that has the characteristic of a “kidnap note”.

As a sample scenario for the second requirement, two fonts are built:

- **Monotype Sans Duospace™ WT SC** contains about 50K characters in the Unicode BMP range with a Simplified Chinese glyph style.
- **Monotype Sans Duospace Ext B** contains the 43,253 glyphs from the supplementary planes required for Simplified Chinese.

Both are packaged as complete TTFs, but the Ext B font is linked to the WT SC font so that a data stream reference to the WT SC font would result in both fonts being downloaded and activated, with the access order being:

1. WT SC
2. Ext B

For fonts in resource libraries, font linking is accomplished with the RAT. A specific table vector type is used to specify the full font name of a linked font, and this table vector is specified in the repeating group for the base font. The linked font table vector can be specified multiple times to link multiple fonts in this manner. The order in which the linked font table vectors are specified determines

the order in which the fonts are linked. Inline fonts can also be linked; see Chapter 6, “How are inline TrueType and OpenType fonts supported?,” on page 31.

Note: If an inline resource group is specified for the print file, this resource group is always searched first for a specified linked font, even if the base font is accessed through the RAT. If the linked font is not found in the resource group, the RAT is accessed to locate the linked font in a library.

Font capture

Font capture is a function whereby a downloaded font can be captured by the printer and treated as if it were a printer-resident font. Captured fonts are retained in the printer across job boundaries and power cycles. Unlike resident fonts which are not deleted, captured fonts may be deleted in the printer when storage space is required. The deletion is performed using printer-specific algorithms. Font capture is enabled by several parameters in the RAT:

- Font permission bits
- Font OID

Permission bits

Two flags that are set in the RAT by the Font Installer for AFP Systems are used by the print server to determine whether a TrueType or OpenType font is eligible for capture. These flags are a summary indication of

- The permission bits in the font itself
- The intent of the person running the Font Installer application

Private

- A zero value for this flag indicates that the Font Installer considers this font to be a public resource, and therefore, a candidate for font capture by printers.
- A one value for this flag indicates that the Font Installer considers this font to be a private resource, and therefore, not a candidate for font capture by printers. Private resources are always downloaded to the printer. Some applications delete private resources when the job that caused the resource to be downloaded is completed; others, including Infoprint Manager for AIX, Infoprint Manager for Windows, and PSF for z/OS, do not.

Capture

- A zero value for this flag indicates that the Font Installer does not allow this font to be captured.
- A one value for this flag indicates that the Font Installer allows this font to be captured. The following additional requirements must be met before the presentation server will actually let font capture take place:
  - The font must be identified as public by the Font Installer (Private bit = B'0').
  - The font must be in a location that the presentation system considers secure.

    Notes:

    1. An inline resource group is normally not considered secure.
    2. PSF for z/OS does not consider user library resources secure.

Object identifiers (OIDs)

Object identifiers (OIDs) are unique identifiers that are based on ISO standard ISO/IEC 8824:1990(E). They are encoded using the Basic Encoding Rules for ASN.1, definite short form, specified in ISO/IEC 8825:1990(E).

OIDs are used to provide universally-unique identifiers for specific resource objects that are to be captured in the printer. Without a unique identification scheme, the integrity of the object capture process cannot be guaranteed. For example, assume a font is built and named `Custom Serif Font 1`. Also assume that the font is captured by this name. Sometime later a second user builds a totally different font, but by coincidence also calls the font `Custom Serif Font 1`. A data stream that references a font by that name would now use the captured font, whether it is the right font or not.

Font Installer for AFP Systems generates an OID for the font and places this OID (using a specific table vector type) into the RAT repeating group for the font. If the font is installed as part of a collection, the Font Installer generates an OID for the collection.

OID generation algorithm: The OID that is placed into the RAT by the Font Installer consists of two parts:

1. A constant part that is based on a fixed sequence of nodes in the ISO OID naming tree
2. A variable part that is algorithmically generated based on the font file content

This scheme allows the OID to be regenerated and verified by any component in the presentation system, such as the print server or the printer control unit.

- **Constant part:** This part has 7 components defined as follows:
  1. ISO, value = 1
  2. Identified organization, value = 3
  3. IBM, value = 18
  4. Objects, value = 0
  5. Print, value = 4
  6. Document and object formats, value = 1
  7. Data stream OID algorithm, value = 5

  Therefore, the constant part for a font or TTC OID takes the following form: `1.3.18.0.4.1.5`.
- **Variable part:** This part consists of 5 components defined as follows:
  1. MD5 fingerprint, 16 byte hexadecimal value in the human readable form of the OID. This is a checksum calculated from the entire object (in stream format; that is, after any environment-specific encapsulation or blocking has been removed). The algorithm is the RSA Data Security, Inc. MD5 Message-Digest Algorithm described in RFC 1321. This algorithm claims to be unique to a 1 in 2**64 probability, given two different byte strings of independent size.
  2. Size of the object, number of bytes in the object; this is the actual size of the object after any environment-specific encapsulation or blocking has been removed. This is a variable-length value.
  3. Supplier ID, value = 0 (for unspecified supplier); additional supplier IDs could be registered in the future.
  4. Customer ID, value = 0 (for unspecified customer ID); actual customer IDs could be registered in the future.
  5. Component reserved for future use, value = 0 (for unspecified)

Since all OID components except the last four components are a fixed size, component growth is minimal; an OID computed from this algorithm is approximately 33 bytes long in ASN.1 definite short form.

Following is an example showing an OID in two forms:

Human-readable form of OID

```
1.3.18.0.4.1.5.X'FFEEDDCCBBAA99887766554433221100'.X'0F4240' .0.0.0
```

ASN.1 definite short form OID

```
X'061F2B120004010583FFEEEEF397BAD4E690F7B395A8C39988A200BD8440000000'
```

For a description of the ASN.1 short form notation, see the *Mixed Object Document Content Architecture Reference*.

Chapter 5. How are TrueType and OpenType Fonts referenced in an AFP data stream?

References to TrueType and OpenType fonts can be made anywhere that a reference to a FOCA font can be made; that is, they can be made in Presentation Text Object Content Architecture (PTOCA), AFP Graphic Object Content Architecture™ (BCOCA™) objects. References to TrueType and OpenType fonts and to FOCA fonts can also be mixed in any combination. Whereas FOCA fonts are referenced with coded font names, Global Resource Identifiers (GRIDs), or code page and font character set name pairs, TrueType and OpenType fonts are always referenced with a name that is reflected in the actual font file: the *full font name*.

The system model for font referencing and glyph rendering is shown in Figure 2 on page 20.

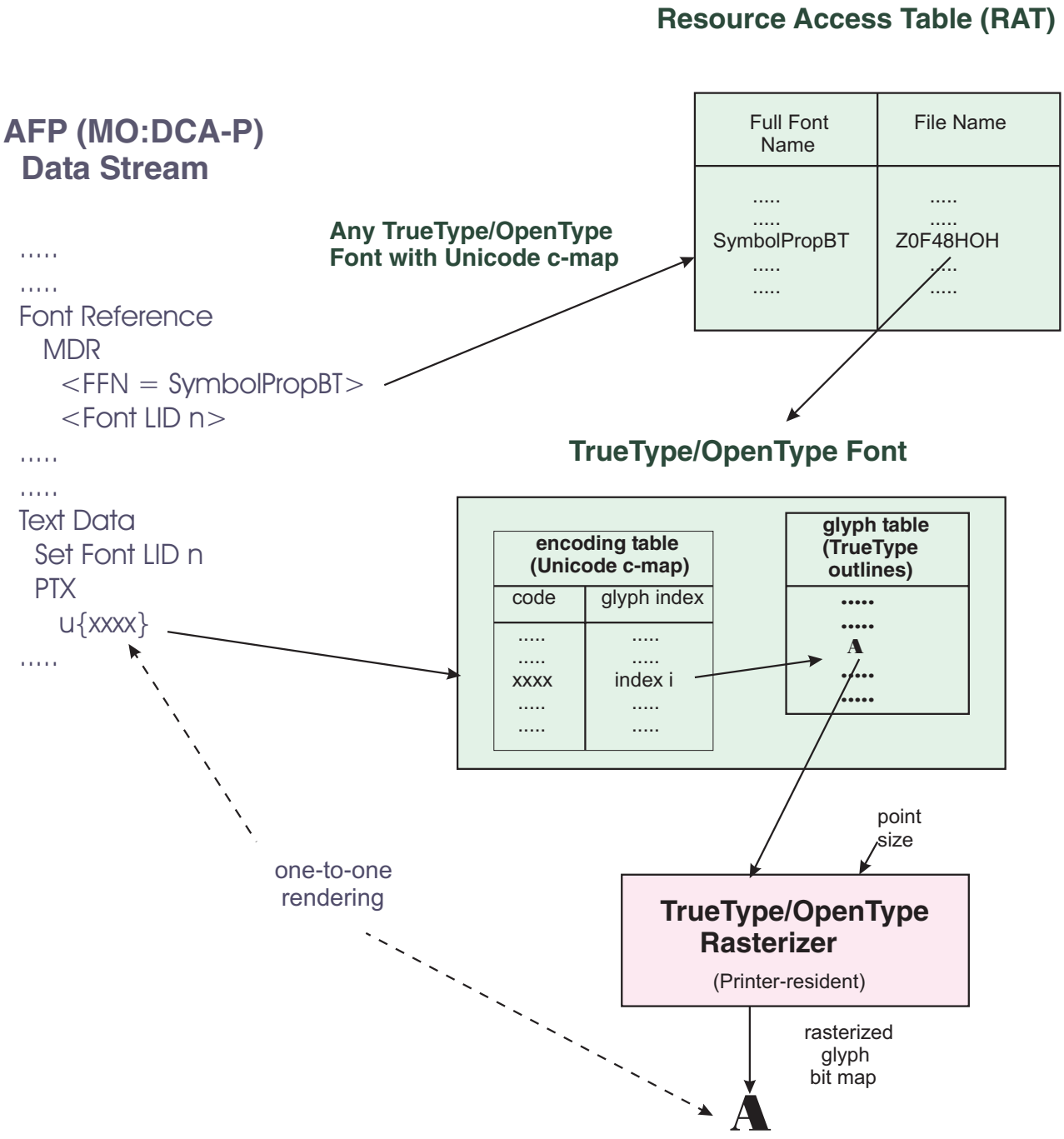


Figure 2. System model for font referencing and glyph rendering

Full font names

The identifier that is used to reference a TrueType or OpenType font in the Mixed Object Document Content Architecture™ (MO:DCA) data stream is a *full font name*, for example, `Times New Roman Bold`. This is an identifier that:

- Is defined within the font file
- Should be distinctive, although it is not required to be unique
- Is the name that the Windows operating systems uses to identify the font

It is a combination of the font family name (for example, `Times New Roman`) and the font subfamily (style) name (for example, `Bold`). The full font name is specified in a name record with Name ID 4 in the mandatory naming table of the font file.

Characteristics of full font names

Two characteristics of the full font name must be taken into account when using it to reference a TrueType or OpenType font in a MO:DCA data stream:

Language

The full font name may be specified in a number of languages in name records of a font file. The language used for a given name record is specified with a language identifier (LCID in the TrueType specification or LID in the IBM Font Installer for AFP Systems). For example, English–United States is assigned LCID X'0409' (1033). The language used to specify the full font name in the MO:DCA font reference may be any of the languages specified in a name record for the full font name with the encoding defined by EncEnv = Microsoft (X'0003') and EncID = Unicode (X'0001'). For example, a font might have two full font name records:

- `Our Company Logo` in Canadian English (LCID X'1009')
- `Logo de notre société` in Canadian French (LCID X'0C0C')

Encoding

The encoding used to specify the full font name in the MO:DCA font reference is defined by a Coded Graphic Character Set Global Identifier (X'01') triplet that precedes the full font name in the font reference. If this triplet is not specified, AFP print servers default to code page 500, which defines an EBCDIC encoding and is identified formally with CPGID = CCSID = 500.

Notes:

1. The encoding for the full font name in the MO:DCA font reference does not need to match the encoding for the full font name in the font Naming Table.
2. In the RAT, the full font name is always encoded in UTF-16.

Identifying the encoding for a full font name

The Coded Graphic Character Set Global Identifier (X'01') triplet is used in MO:DCA data streams to specify the encoding for character strings, such as names in structured fields and triplets. This is not to be confused with the encoding of the presentation data, which is normally defined indirectly by the font that is used to render the data, but in some cases may also be defined with the Encoding Scheme Identifier (X'50') triplet or with an explicit reference to a code page; see Chapter 8, “How can TrueType and OpenType fonts be used with legacy character encodings?,” on page 39. The X'01' triplet can specify the encoding in two ways:

- By defining the code page and character set for the encoding. The character set is specified with a Graphic Character Set Global Identifier (GCSGID), and the code page is specified with a Code Page Global Identifier (CPGID).
- By specifying a Coded Character Set Identifier (CCSID) as defined and registered by the Character Data Representation Architecture (CDRA). The CCSID can be resolved to identify the value of the code page and character set. See the *Character Data Representation Architecture: Reference and Registry* or the iSeries™ CCSID Information web page at http://www-1.ibm.com/servers/eserver/iseries/software/globalization/ccsid.html for detailed information.

It is recommended that the CCSID form of the X'01' triplet be used since most conversion services use the CCSID as input.

The X'01' triplet must be specified before the structures whose encoding it defines. For example, when the X'01' triplet is specified on a Map Data Resource (MDR) structured field to define the encoding for the full font name, it must precede the FQN type X'DE' triplet that carries this name. For the format of the MDR structured field, refer to *Mixed Object Document Content Architecture Reference*.

Font descriptive information

In addition to specifying the full font name of the desired font, the font reference in the MO:DCA data stream also specifies a number of parameters needed to render glyphs from the font. These parameters are specified on the MO:DCA font reference using the Data-Object Font Descriptor (X'8B') triplet. The parameters are as follows:

Font technology
This parameter is set to X'20' to indicate TrueType or OpenType font technology.

Vertical font size
The desired vertical size for the rendered glyphs, in 1440ths of an inch. If the desired point size is known, multiply it by 20 to determine the size in 1440ths.

Horizontal scale factor
A value in 1440ths of an inch that specifies whether the glyphs should be scaled uniformly or anamorphically. If this parameter is set to X'0000' or if it is equal to the vertical font size, glyphs are scaled uniformly to the desired size. If this parameter is set to a non-zero value that is different from the vertical font size, glyphs are stretched or compressed in the horizontal direction (relative to the vertical direction) by the ratio of horizontal scale factor to vertical font size.

Character rotation
Specifies the clockwise character rotation in degrees. Valid values are:
X'0000' = 0 degrees
Used for left-to-right writing mode
X'2D00' = 90 degrees
Used for bottom-to-top writing mode
X'5A00' = 180 degrees
Used for right-to-left writing mode
X'8700' = 270 degrees
Used for top-to-bottom writing mode

TrueType fonts provide two sets of metrics (horizontal and vertical) to allow character placement for different writing modes. The metrics for horizontal writing are used when the character rotation is 0 degrees (used for left-to-right writing mode), and a modified version of horizontal metrics is used for a 180 degree character rotation (used for right-to-left writing mode). Likewise, the metrics for vertical writing are used when the character rotation is 270 degrees (used for top-to-bottom writing mode), and a modified version of the vertical metrics is used for 90 degree character rotation (used for bottom-to top writing mode). Figure 3 on page 23 shows the placement of characters based on the character rotation value and the PTOCA inline and baseline direction values.

Font encoding scheme

This selects the font cmaps to use for processing text. The selection is made by specifying two parameters in the triplet:

- Encoding Environment (called Platform ID in TrueType and OpenType font files), which must be set to X'0003' = Microsoft
- Encoding ID, which must be set to X'0001' = Unicode; this encoding selects a Unicode-enabled font

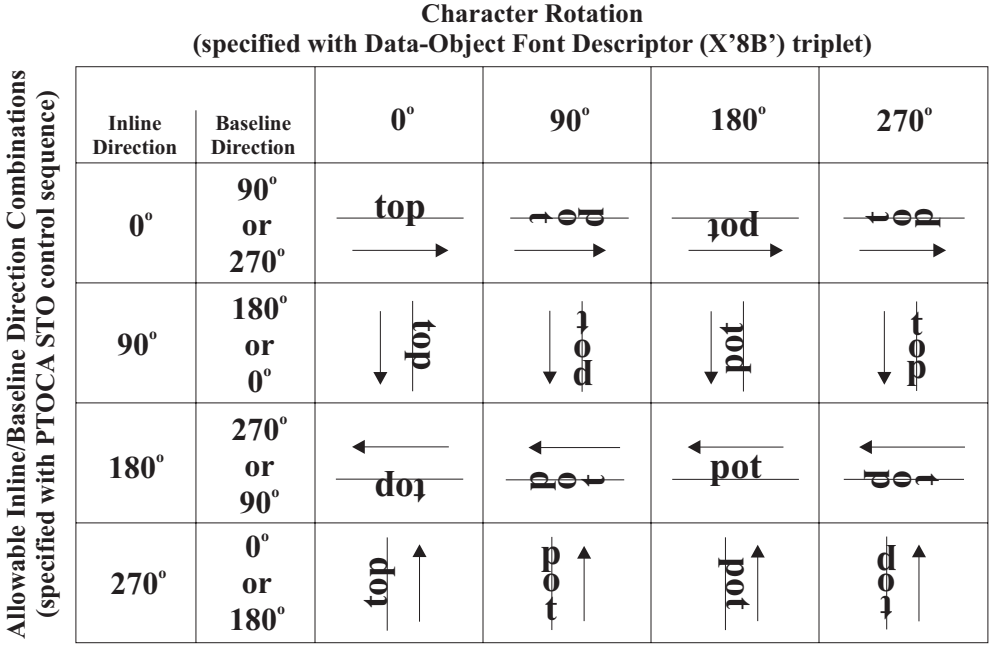


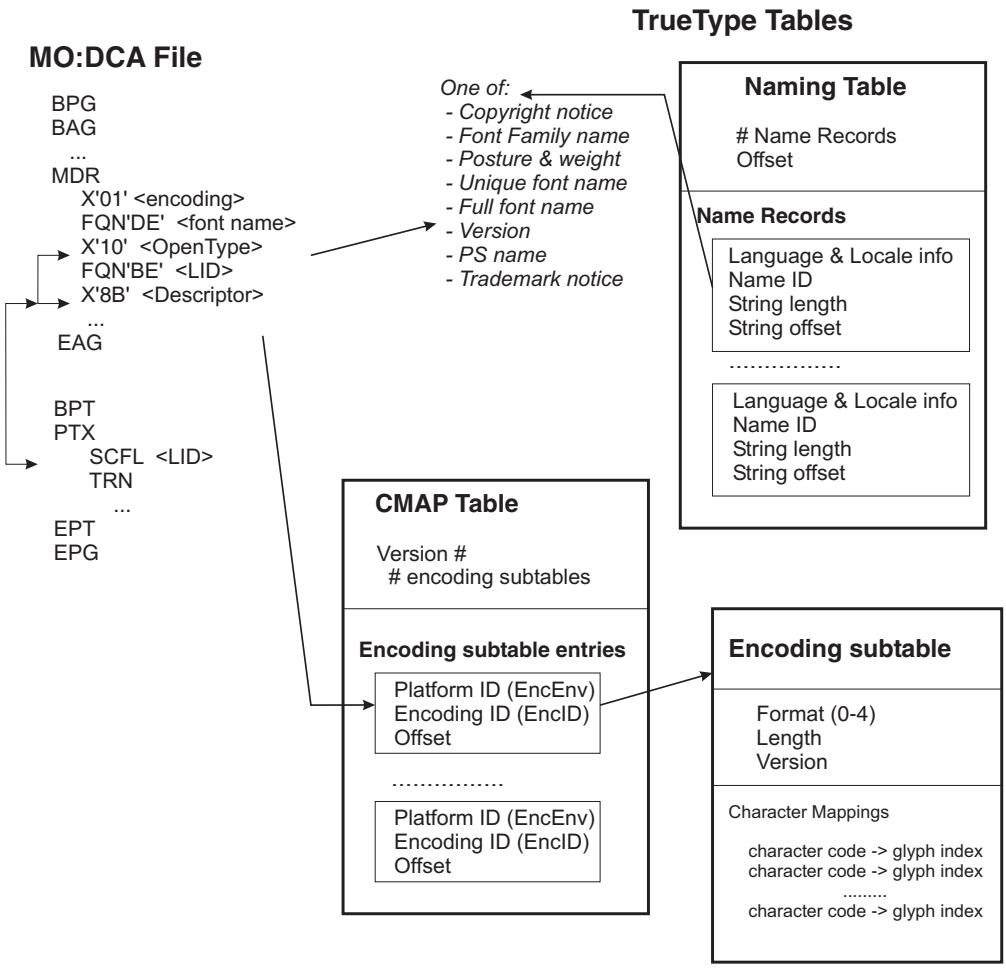
Figure 3. Character placement based on character rotation, inline direction, and baseline direction

MO:DCA font reference: the MDR structured field

The MO:DCA font reference to a TrueType or OpenType font is generated using a Map Data Resource (MDR) structured field instead of an MCF (Map Coded Font) resource. This structured field identifies the TrueType or OpenType font using a full font name, specifies desired attributes of the font, such as point size, and maps the full font name to the local ID that is used to reference the font in the data stream. If the encoding of the user data does not match the Unicode encoding in the font, the MDR may also carry information to identify the user data encoding so that it may be converted to UTF-16BE encoding. For a formal definition of the MDR, see the *Mixed Object Document Content Architecture Reference*.

The MDR uses MO:DCA triplets to specify the various parameters. Figure 4 shows how a TrueType or OpenType font is referenced with a full font name in a MO:DCA data stream.

Figure 4. Referencing a TrueType or OpenType font



The target of the full font name reference in the data stream can be one of the following:

- An FQN type X'01' triplet on the BRS for an inline font container
- An FQN type X'6E' triplet on the BRS for an inline collection container
- A TrueType or OpenType font repeating group entry in the RAT

Because there is no restriction on the language that is used for a full font name reference in the MDR, the targets of the font search should specify the full font name in all of the languages that are supported in the font naming table.

Table 3 on page 25 shows the triplets that can be specified on the MDR structured field for a TrueType or OpenType font reference in the Active Environment Group (AEG) of a page, or the Object Environment Group (OEG) of an AFP GOCA or BCOCA object. Note that when the MDR is specified in the AEG for PTOCA text or in the OEG for AFP GOCA and BCOCA objects, the FQN type X'BE' triplet must carry the true local ID (LID) that is specified for rendering the character string. When the MDR is factored up to the AEG (from the respective OEGs) for AFP GOCA or BCOCA text, the FQN type X'BE' triplet should specify a LID value of

X'FE'.

Table 3. Triplets on the MDR TrueType or OpenType font reference

Parameter	Triplet
Full font name	Fully Qualified Name (X'02') type X'DE' (Data Object External Resource Reference) triplet; mandatory, must be specified once
Local ID (LID) of the font reference in the PTOCA, AFP GOCA, or BCOCA text	Fully Qualified Name (X'02') type X'BE' (Data Object Internal Resource Reference) triplet; mandatory, must be specified once
Characterization of the object being referenced	Object Classification (X'10') triplet; mandatory, must be specified once and must specify object-type OID = X'06072B120004010133' (TrueType or OpenType Font)
Font descriptive information	Data-object Font Descriptor (X'8B') triplet; mandatory, must be specified once
Definition of the encoding scheme used for character strings in the MDR that follow this triplet	Coded Graphic Character Set Global Identifier (X'01') triplet; optional, should be specified once to define the encoding used for the full font name. The CCSID form of this triplet should be used. This triplet must appear before the FQN type X'DE' triplet that specifies the full font name.
Name of the code page that defines the encoding of the presentation-data if that encoding is not UTF-8 or UTF-16; see Chapter 8, “How can TrueType and OpenType fonts be used with legacy character encodings?,” on page 39.	Fully Qualified Name (X'02') type X'85' (Code Page Name Reference) triplet; optional, either this triplet or the Font Coded Graphic Character Set (X'20') triplet may be specified once
CPGID of the code page that defines the encoding of the presentation-data if that encoding is not UTF-8 or UTF-16; see Chapter 8, “How can TrueType and OpenType fonts be used with legacy character encodings?,” on page 39.	Font Coded Graphic Character Set (X'20') triplet; optional, either this triplet or the Fully Qualified Name (X'02') type X'85' (Code Page Name Reference) triplet may be specified once
Indicator that the presentation data is encoded in UTF-8; see Chapter 8, “How can TrueType and OpenType fonts be used with legacy character encodings?,” on page 39.	Encoding Scheme Identifier (X'50') triplet with ESidUD = X'7807'; optional, may be specified once

The following examples show the triplet structure of specific MDR TrueType or OpenType font repeating groups for different user-data encodings:

Table 4. MDR repeating group for UTF-16 data

MDR repeating group mapping a TrueType or OpenType font; user data encoding = font encoding = UTF-16	Coded Graphic Character Set Global Identifier (X'01') triplet
	Fully Qualified Name (X'02') triplet, type X'DE' (Data Object External Resource Reference)
	Fully Qualified Name (X'02') triplet, type X'BE' (Data Object Internal Resource Reference)
	Object Classification (X'10') triplet
	Data-Object Font Descriptor (X'8B') triplet

Table 5. MDR repeating group for UTF-8 data

MDR repeating group mapping a TrueType or OpenType font; user data encoding = UTF-8	Coded Graphic Character Set Global Identifier (X'01') triplet
	Fully Qualified Name (X'02') triplet, type X'DE' (Data Object External Resource Reference)
	Fully Qualified Name (X'02') triplet, type X'BE' (Data Object Internal Resource Reference)
	Object Classification (X'10') triplet
	Data-Object Font Descriptor X'8B') triplet
	Encoding Scheme ID (X'50') triplet

Table 6. MDR repeating group for EBCDIC or ASCII Data with CPGID

MDR repeating group mapping a TrueType or OpenType font; user data encoding = EBCDIC or ASCII, code page identified with CPGID	Coded Graphic Character Set Global Identifier (X'01') triplet
	Fully Qualified Name (X'02') triplet, type X'DE' (Data Object External Resource Reference)
	Fully Qualified Name (X'02') triplet, type X'BE' (Data Object Internal Resource Reference)
	Object Classification (X'10') triplet
	Data-Object Font Descriptor (X'8B') triplet
	Font Coded Graphic Character Set Global Identifier (X'20') triplet

Table 7. MDR repeating group for EBCDIC or ASCII data with code page name

MDR repeating group mapping a TrueType or OpenType font; user data encoding = EBCDIC or ASCII, code page identified with name	Coded Graphic Character Set Global Identifier (X'01') triplet
	Fully Qualified Name (X'02') triplet, type X'DE' (Data Object External Resource Reference)
	Fully Qualified Name (X'02') triplet, type X'BE' (Data Object Internal Resource Reference)
	Object Classification (X'10') triplet
	Data-Object Font Descriptor (X'8B') triplet
	Fully Qualified Name (X'02') triplet, type X'85' (Code Page Name Reference)

Transparency of font linking

The font linking function is transparent to the MO:DCA MDR font reference. That is, the reference to a TrueType or OpenType font does not depend on how many (if any) additional TrueType and OpenType fonts are linked to the font being referenced. *Therefore, the data stream and the application program that generates it do not need to change when additional fonts are linked to, or linked fonts are removed from, the referenced font.*

Note: This assumes that linked fonts are removed because they are not needed. If the data stream uses characters in the linked fonts, it needs to change.

Transparency of font collections

The packaging of the referenced font is transparent to the MO:DCA MDR font reference. That is, the reference to a TrueType or OpenType font uses only the full font name of the font and does not depend on how the font is packaged, either with other fonts in a TrueType collection (TTC) file or standalone in a TrueType font (TTF) or OpenType font (OTF) file. If the font is in a TTC, the reference does not depend on which TTC that is. *Therefore, the data stream and the application program that generates it do not need to change when the packaging of the referenced font is changed.*

Transparency of font capture

The enablement of a font for capture in the printer is transparent to the MO:DCA MDR font reference. That is, the reference to a TrueType or OpenType font uses only a full font name and does not depend on whether the font cannot be captured, whether the font can be captured but needs to be downloaded first, or whether the font is already resident in the printer. *Therefore, the data stream and the application program that generates it do not need to change when the attributes that determine the capturability of the referenced font are changed.*

Preloading TrueType and OpenType fonts

TrueType or OpenType font files can be very large, especially if the fonts support Asian scripts with large character sets. Font files in the 20–40 MB range are not uncommon. System throughput may be affected if such fonts are downloaded to the printer when called for in the Active Environment Group (AEG) of the page. To avoid this overhead at page-print time, TrueType and OpenType fonts can be referenced in the Resource Environment Group (REG) of the document. The REG is

specified before the first page in the document and causes the AFP print server to download any resources that are mapped in the REG and not available in the printer before the first page of the document is processed.

Note: PSF/400 does not support REGs.

How are TrueType and OpenType fonts referenced in AFP line data?

TrueType and OpenType fonts may be used for printing AFP line data fields and records by specifying the MDR structured field in the Active Environment Group (AEG) of the page definition.

The encoding of the line data can be single- or double-byte EBCDIC, single-byte ASCII, UTF-8, or UTF-16. UTF-16 data can be either big-endian or little-endian.

Notes:

1. AFP does not support mixed-byte encodings. Because existing ASCII "double-byte" code pages are actually mixed-byte, you must convert double-byte ASCII line data to a supported encoding before printing it. Infoprint Manager for AIX and Infoprint Manager for Windows provide a configurable transform subsystem that, after initial setup, does this automatically. You can also use a conversion utility like **iconv**.
2. The following restrictions exist on the use of UTF-8 and UTF-16 traditional line data and record format line data:
   a. If there is a byte-order mark (BOM), it must be the first 2 bytes of data (UTF-16) or the first 3 bytes of data (UTF-8) in the first line of line data in the print file. This must be the only place that the BOM appears in the user data.
   b. If there is no BOM, UTF-16 data must be big-endian.
   c. If the print file contains a BOM, you must specify the UDType parameter on the PPFA PAGEDEF command.
   d. The number of data bytes, including the Record ID, but not including carriage controls (CCs) or table reference characters (TRCs), in any record in a UTF-16 data file must be even. If the number of data bytes is odd:
      - For UTF-16LE, processing terminates with an error message.
      - For UTF-16BE, processing continues, but the printer may terminate the page.
   e. If the print file contains a BOM, the raster or outline fonts specified for the file must match the encoding specified on the PPFA PAGEDEF command. For example:
      - UTF-16 data requires Unicode-enabled fonts.
      - If UTF-8 data is used with ASCII fonts, each character must be represented by 1 byte.

      If the encoding of the user data conflicts with the font being used, processing terminates with an error message.
3. When TRCs are used in the line data to select a font, the font mappings in the Active Environment Group (AEG) of the Data Map must either be all FOCA font mappings, specified with Map Coded Font (MCF) structured fields, or all TrueType or OpenType font mappings, specified with MDR structured fields. The reason for this restriction is that TRC 0 (X'00' or X'F0') selects the first font mapped in the AEG; TRC 1 (X'01' or X'F1') selects the second font mapped in the AEG; and so on. Because MCF mappings must precede MDR mappings,

this historically positional selection of fonts mapped in the AEG precludes the use of a mixture of fonts mapped with MCFs and fonts mapped with MDRs.

4. TRCs are not used in record format line data. If included, they are treated as data bytes.

EBCDIC and ASCII encodings are identified by specifying the code page identifier on the MDR. This identifier can be a code page name, which is specified with an FQN X'85' triplet, or a code page global ID, which is specified with a Font Coded Graphic Character Set Identifier (X'20') triplet.

Table 8 shows the triplets that can be specified on the MDR structured field in the AEG of a page definition.

Table 8. Triplets on the MDR TrueType or OpenType font reference in a page definition

Parameter	**Triplet**
Full font name	Fully Qualified Name (X'02') type X'DE' (Data Object External Resource Reference) triplet; mandatory, must be specified once
Local ID (LID) of the font reference in the descriptor used to process the line data	Fully Qualified Name (X'02') type X'BE' (Data Object Internal Resource Reference) triplet; mandatory, must be specified once
Characterization of the font object being referenced	Object Classification (X'10') triplet; mandatory, must be specified once and must specify object-type OID = X'06072B120004010133' (TrueType or OpenType Font)
Font descriptive information	Data-object Font Descriptor (X'8B') triplet; mandatory, must be specified once
Definition of the encoding scheme used for character strings in the MDR that follow this triplet	Coded Graphic Character Set Global Identifier (X'01') triplet; optional, should be specified once to define the encoding used for the full font name. The CCSID form of this triplet should be used.
Name of the code page that defines the encoding of the line data if that encoding is not UTF-8 or UTF-16; see Chapter 8, “How can TrueType and OpenType fonts be used with legacy character encodings?,” on page 39.	Fully Qualified Name (X'02') type X'85' (Code Page Name Reference) triplet; optional, either this triplet or the Font Coded Graphic Character Set (X'20') triplet may be specified once
CPGID of the code page that defines the encoding of the line data if that encoding is not UTF-8 or UTF-16; see Chapter 8, “How can TrueType and OpenType fonts be used with legacy character encodings?,” on page 39.	Font Coded Graphic Character Set (X'20') triplet; optional, either this triplet or the Fully Qualified Name (X'02') type X'85' (Code Page Name Reference) triplet may be specified once
Indicator that the line data is encoded in UTF-8; see Chapter 8, “How can TrueType and OpenType fonts be used with legacy character encodings?,” on page 39.	Encoding Scheme Identifier (X'50') triplet with ESidUD = X'7807'; optional, may be specified once

Chapter 6. How are inline TrueType and OpenType fonts supported?

Like other AFP resources, TrueType and OpenType fonts may be carried in a print-file-level resource group. Such a resource group is also called an *inline resource group* or an *external resource group*. This resource group is optional, and if present, must precede the first document in a print file. Only a single resource group is permitted in a given print file. When a print server attempts to locate a resource, such as a TrueType or OpenType font, that is referenced in the data stream, the inline resource group (if present) is always searched first, that is, it is *always* searched ahead of any resource library.

In general, any resource carried in an inline resource group in an AFP environment must be enveloped with a Begin Resource–End Resource (BRS/ERS) envelope; this includes object container resources carried in a Begin Object Container–End Object Container (BOC/EOC) envelope. Therefore, an inline TrueType or OpenType font or TTC resource must be carried in an object container delimited by BOC/EOC, which in turn is enveloped by BRS/ERS. The actual font data or collection data is partitioned and carried in Object Container Data (OCD) structured fields within the BOC/EOC container. Both font files and collection files may be carried in inline resource groups. Figure 5 shows the general structure of the font or collection resource in its envelopes as carried within a resource group bounded by Begin Resource Group (BRG) and End Resource Group (ERG).

Figure 5. Structure of a resource group

For a formal definition of the BRS/ERS, BOC/EOC and the structure of MO:DCA resource groups and print files, see the *Mixed Object Document Content Architecture Reference*.

FQN triplet types on containers

The MO:DCA Fully Qualified Name X'02' (FQN) triplet is used to specify various types of identifiers on the BRS and BOC envelopes for TrueType and OpenType fonts and collections in inline resource groups. Three parameters of the FQN triplet are of note:

FQNType Specifies the type of identifier in this triplet and how it is used. For example, the identifier type may be a full font name for a base font or a full font name for a linked font.

FQNFmt Specifies the format of the identifier carried in this triplet: either character string (FQNFmt = X'00') or object OID (FQNFmt = X'10').

FQName Specifies the actual identifier, either as a name or as an object OID.

FQN type X'01' triplet

The FQN type X'01' (Replace First GID Name) triplet can be specified on the BRS and BOC structured fields to override the 8-byte token name that is specified on the structured field.

Note: PSF ignores this triplet on the BOC structured field unless it specifies an object OID.

The triplet is optional, but when specified, it replaces the token name. Since the token name is limited to 8 bytes, which with a single-byte encoding maps to 8 characters and with a double–byte encoding like UTF-16 maps to 4 characters, it is not a practical parameter for specifying full font names. Therefore the FQN X'01' triplet should always be used to specify a full font name on a BRS or BOC structured field.

Note: The FQN type X'01' triplet is always specified if the identifier is an object OID for the TrueType or OpenType font or TTC, that is, a token name cannot be used to specify an object OID. The FQN X'01' triplet with object OID may be specified on the BOC, but not on the BRS.

FQN type X'6E' triplet

The FQN type X'6E' (Data-object Font Base Font Identifier) triplet is used to specify the full font name of a specific font in a font collection. It is denoted as *base font* identifier to differentiate it from a *linked font* identifier.

This triplet can only be specified on the BRS and BOC for a TrueType or OpenType Collection, which is identified with the object-type OID for a TTC in the mandatory X'10' triplet on the BRS and BOC.

Note: PSF normally ignores this triplet on the BOC structured field. You should specify it on the BRS structured field.

The object-type OID for a TTC is X'06072B120004010135'; the object-type OID for a TrueType or OpenType font is X'06072B120004010133'. The triplet can be specified multiple times to reflect multiple base fonts in the TTC, and also to reflect the full font name of a given base font in multiple languages. Since there is no restriction on the language that is used for a full font name reference in the MDR, the targets of the font search (the BRS structured fields in an inline resource group and the repeating groups in a RAT) must specify the full font name in all of the languages that are supported in the font naming table.

FQN type X'7E' triplet

The FQN type X'7E' (Data-object Font Linked Font Identifier) triplet is used to specify the full font name of a font that is to be linked to the base font. It is denoted as a *linked font* identifier to differentiate it from a *base font* identifier.

This triplet can be specified multiple times to reflect multiple linked fonts. The order of the triplets determines the order in which the linked fonts are processed. When the triplet is specified on the BRS for a TrueType/OpenType Collection, it applies to the immediately preceding font that was specified with an FQN type

X'6E' triplet on the same BRS. Since this triplet is a reference to a linked font, it need only be specified in one of the languages supported in the linked font's naming table.

The target of the linked font reference can be one of the following:
- An FQN type X'01' triplet on the BRS for an inline font container
- An FQN type X'6E'' triplet on the BRS for an inline collection container
- A TrueType or OpenType font repeating group entry in the RAT

These targets of the font search should specify the full font name in all of the languages that are supported in the linked font's naming table.

Note: PSF normally ignores this triplet on the BOC structured field. You should specify it on the BRS structured field.

Font containers

For a TrueType or OpenType font container, the BRS specifies the following parameters using the indicated triplets:

Table 9. Triplets on the BRS for a font container

Parameter	Triplet
Characterization of the resource container being enveloped by the BRS	Resource Object Type X'21') triplet; mandatory, must be specified once and must specify ObjType = X'92' (Object Container). Note that this is a retired triplet which may only be used on the BRS structured field.
Characterization of the object in the resource container enveloped by BRS/ERS	Object Classification (X'10') triplet; mandatory, must be specified once and must specify object-type OID = X'06072B120004010133' (TrueType or OpenType Font)
Full font name	Fully Qualified Name type X'01' (Replace First GID Name) triplet; may occur more than once to specify the full font name in a language used in the font naming table. The character encoding is UTF-16BE. For example, if the font Naming Table contains two name records for the full font name (Name ID 4), one in English–United States (LCID = X'0409') and one in German–Standard (LCID X'0407'), both encoded in UTF-16, each of these names, encoded in UTF-16BE, is carried in a FQN type X'01' triplet on the BRS.
Full font name of a linked font	Fully Qualified Name type X'7E' (Data-object Font Linked Font Identifier) triplet; optional and may occur more than once to specify the full font names of linked fonts for the base font carried in the container. The character encoding is UTF-16BE. The order in which the FQN type X'7E' triplets are specified determines the order in which the linked fonts are processed.

Table 9. Triplets on the BRS for a font container (continued)

Parameter	Triplet
Definition of the encoding scheme used for character strings in the BRS that follow this triplet	Coded Graphic Character Set Global Identifier (X'01') triplet; optional for TrueType or OpenType font containers since the full font name must be encoded in UTF-16BE.

The BOC may optionally repeat the FQN type X'7E' triplet that is specified on the BRS, but need not. In addition:

- The Resource Object Type (X'21') triplet is not allowed on the BOC.
- The Fully Qualified Name type X'01' (Replace First GID Name) triplet may be specified on the BOC using FQNFmt = X'10' to specify the object OID for the font instead of a full font name; in that case the triplet may only be specified once.

If the BOC specifies an object OID, the print server first attempts to use a printer-resident version of the font that has a matching object OID. If a resident version cannot be found, the print server downloads the font. In most cases, resources in an inline resource group are considered private resources and are not enabled for capture; however, if a particular system has been configured to allow inline resources to be captured, the print server may allow the font to be captured when downloaded.

Font collection containers

For a TTC container, the BRS specifies the following parameters using the indicated triplets:

Table 10. Triplets on the BRS for a font collection container

Parameter	Triplet
Characterization of the resource container being enveloped by the BRS	Resource Object Type (X'21') triplet; mandatory, must be specified once and must specify ObjType = X'92' (Object Container). **Note:** This is a retired triplet that may only be used on the BRS structured field.
Characterization of the object in the resource container enveloped by BRS/ERS	Object Classification (X'10') triplet; mandatory, must be specified once and must specify object-type OID = X'06072B120004010135' (TrueType or OpenType Collection)
Full font name of a base font	Fully Qualified Name type X'6E' (Data-object Font Base Font Identifier) triplet; used to specify the full font name of a base font that is in the collection. For each base font, the triplet may be specified multiple times to specify the full font name in multiple languages. The character encoding is UTF-16BE.

Table 10. Triplets on the BRS for a font collection container (continued)

Parameter	Triplet
Full font name of a linked font	Fully Qualified Name type X'7E' (Data-object Font Linked Font Identifier) triplet; optional and may occur more than once to specify the full font names of linked fonts for the immediately preceding base font that is specified with a FQN type X'6E' triplet on this BRS. The character encoding is UTF-16BE. The order in which the FQN type X'7E' triplets are specified following a given base font name determines the order in which the linked fonts are processed. The sequence of linked fonts specified using this triplet must be repeated after every language instance of the base font full font name.
Definition of the encoding scheme used for character strings in the BRS that follow this triplet	Coded Graphic Character Set Global Identifier (X'01') triplet; optional for TTC containers since the full font name must be encoded in UTF-16BE.
Name of font collection	The TTC must be given a name that is unique within the resource group. This name can be specified using the 8-byte token name in the BRS, or it can be specified using a Fully Qualified Name type X'01' (Replace First GID Name) triplet.

The BOC may optionally repeat the FQN type X'6E' and FQN type X'7E' triplets that are specified on the BRS, but need not. In addition:

- The Resource Object Type (X'21') triplet is not allowed on the BOC.
- The Fully Qualified Name type X'01' (Replace First GID Name) triplet may be specified once using FQNFmt = X'10' to specify the object OID for the collection.

If the BOC specifies an object OID, the print server first attempts to use a printer-resident version of the collection that has a matching object OID. If a resident version cannot be found, the print server downloads the collection. In most cases, resources in an inline resource group are considered private resources and are not enabled for capture; however, if a particular system has been configured to allow inline resources to be captured, the print server may allow the collection to be captured at download.

Chapter 7. How are TrueType and OpenType fonts located in the resource hierarchy?

A TrueType or OpenType font that is referenced in an AFP (MO:DCA-P) data stream may be located in an inline resource group or in a resource library. It may be packaged as a standalone font or in a font collection. It may also have linked fonts. To support this hierarchy, presentation servers process a TrueType or OpenType font reference in an MDR as follows:

1. **The resource group, if present, is searched for a TrueType or OpenType font container or font collection (TTC) container that specifies a matching full font name.**
   - A font container specifies the full font name with a FQN type X'01' triplet on the BRS of the font container.
   - A TTC container specifies the full font name with a Data Object Font Base Font Identifier (X'6E') triplet on the BRS of the collection container.

   The first matching font container or collection container is used. The font container or font collection container may also specify one or more linked fonts for the referenced font.
   - On a font container, linked fonts are specified with Data-object Font Linked Font Identifier (FQN type X'7E') triplets, which carry the full font names of the linked fonts, on the BRS of the font container.
   - On a TTC container, linked fonts are specified with Data-object Font Linked Font Identifier (FQN type X'7E') triplets that immediately follow the Data Object Font Base Font Identifier (X'6E') triplet for the base font on the BRS for the TTC container.

   The full font names for the linked fonts are used in turn to search the resource group for a font container or a TTC container that carries a base font whose full font name matches the full font name of the linked font.
   - If found, such a font is processed as a linked font for the base font. Multiple linked fonts may be specified, and the order in which they are specified on the base font container determines the order in which they are processed. The base font is always processed first, followed by the first-specified linked font, followed by the next-specified linked font, and so on. The last-specified linked font is processed last.
   - If a linked font cannot be found in either an inline font container or an inline collection container, the full font name of the linked font is used to index the Resource Access Table (RAT) to locate the linked font in a resource library.

   Only one level of linking is supported, that is, if a linked font is found and also specifies its own linked fonts, they are ignored.
2. **If a font matching the MDR reference is not found in an inline font container or in an inline collection container, the presentation server accesses the RAT with the full font name to locate the referenced font in a resource library.** In this case, all linked fonts are specified in the RAT repeating group for the referenced font, and the order in which they are specified determines the order in which they are processed. Both inline linked fonts and library-based linked fonts are used, and the inline resource group is *always* searched for linked fonts ahead of the resource library. The resource group search includes TrueType or OpenType font containers, in which case the linked font name is matched against the FQN type X'01' triplet on the font container,

and TTC containers, in which case the linked font name is matched against the FQN type X'6E' triplets on the collection container. Only one level of linking is supported, that is, if a linked font is found and also specifies its own linked fonts, they are ignored.

Note: For both base and linked fonts, the search order within resource libraries varies by platform. For example:

- On z/OS, the user libraries are searched first, followed by the system libraries. In each case the libraries are searched in the order in which they are concatenated.
- On OS/400 V5R3M0, the search order is:
  - The RAT in /QIBM/UserData/OS400/Fonts/TTFonts
  - The RAT in /QIBM/ProdData/OS400/Fonts/TTFonts
  - All fonts in /QIBM/UserData/OS400/Fonts/TTFonts
  - All fonts in /QIBM/ProdData/OS400/Fonts/TTFonts

Chapter 8. How can TrueType and OpenType fonts be used with legacy character encodings?

Much legacy AFP print data is encoded using EBCDIC-based and ASCII-based encoding schemes. Such encodings are not compatible with the Unicode encoding that is supported in TrueType and OpenType fonts. To enable the rendering of such legacy print data with Unicode-enabled TrueType and OpenType fonts, migration support for non-Unicode encodings has been added to the AFP architecture and to AFP systems and is part of the new TrueType and OpenType font support. The supported migration encodings fall into two categories:

- Legacy EBCDIC and ASCII encodings
- UTF-8 encoding

EBCDIC and ASCII legacy data

Many existing legacy AFP applications generate presentation data that is encoded in EBCDIC-based encodings and ASCII-based encodings. In these applications, such encodings are identified by the code page that is specified in the font reference. For example, if the text data in an application is based on the encoding defined by EBCDIC code page 500, then the MO:DCA data stream that carries this text data contains a font reference that includes code page 500 either directly with a code page name reference or with a code page global ID (CPGID) that is part of a Global Resource Identifier (GRID) or indirectly with a coded font reference that then points to code page 500.

Legacy print data that is encoded in this manner can be rendered with TrueType or OpenType fonts by specifying the code page that defines the encoding of the data on the MDR structured field. The code page can be identified either by name (using a Fully Qualified Name type X'85' triplet) or by Code Page Global Identifier (CPGID) using a Font Coded Graphic Character Set Global Identifier (X'20') triplet. In addition, the encoding can optionally be characterized with the ESidCP parameter in an Encoding Scheme Identifier (X'50') triplet on the MDR. This encoding information is transmitted to the printer along with the font reference and the text data to be rendered, and the printer then converts the legacy-encoded data to UTF-16BE encoded data. Once the data is in UTF-16 format, the UTF-16 cmaps in the TrueType or OpenType font can be accessed to find the proper glyphs.

The X’20’ triplet

The Font Coded Graphic Character Set Global Identifier (X'20') triplet is an optional triplet on the MDR. Either this triplet or the FQN type X'85' triplet may be specified to identify the code page; if both are specified, the FQN type X'85' triplet is ignored. This triplet specifies the Code Page Global Identifier (CPGID) and Graphic Character Set Global Identifier (GCSGID) of an IBM code page that defines the encoding in the user data. Note that the encoding in the user data must match the encoding in the referenced code page. If the user data encoding is single-byte, the code page must be a single-byte code page. If the user-data encoding is double-byte, the code page encoding must be double-byte as well.

The FQN type X'85' triplet

The Fully Qualified Name Type X'85' (Code Page Name Reference) triplet is an optional triplet on the MDR. Either this triplet or the Font Coded Graphic Character Set Global Identifier (X'20') triplet may be specified to identify the code page; if both are specified, the FQN type X'85' triplet is ignored. This triplet specifies the name of an IBM code page that defines the encoding in the user data. In AFP environments, the name consists of 8 characters and follows the naming conventions for AFP code pages defined in *IBM Infoprint Fonts: Font Summary*. An example of a code page name is T1V10500. Note that the encoding in the user data must match the encoding in the referenced code page. If the user data encoding is single-byte, the code page must be a single-byte code page. If the user-data encoding is double-byte, the code page encoding must be double-byte as well.

UTF-8 data

Another type of user-data encoding that is prevalent and that also is not UTF-16 based is UTF-8. Since there are no legacy code pages that support the UTF-8 encoding directly, the above scheme of code page identification does not work for this encoding. Instead, to identify text data that is encoded in UTF-8, the ESidUD parameter in the Encoding Scheme (X'50'') triplet is used on the MDR. A value of ESidUD = X'7807' in this triplet indicates that the user data to be rendered with the referenced font is encoded in UTF-8.

As in the EBCDIC or ASCII scenario, this encoding information is transmitted to the printer along with the font reference and the text data to be rendered, and the printer then converts the UTF-8 encoded data to UTF-16BE encoded data using the algorithm published in the Unicode Standard. Once the data is in UTF-16 format, the UTF-16 cmaps in the font can be accessed to find the proper glyphs.

The Encoding Scheme Identifier (X'50') triplet

The Encoding Scheme Identifier (X'50') triplet has been used in the past to characterize the encoding scheme associated with a code page in a referenced font. This parameter was called ESid. The triplet is being extended to now support an additional parameter to reflect the encoding scheme for user data that is to be rendered with the referenced font. This new parameter is called ESidUD, and the old parameter has been renamed ESidCP.

Not all defined combinations of ESidCP and ESidUD make sense when this triplet is specified on an MDR for a TrueType or OpenType font reference. The ESidCP is only used to characterize the code page encoding when the data is encoded in accordance with a code page identifier that is specified on the MDR, and in that case the ESidUD is ignored and the triplet is optional. The ESidUD is only used to identify user data that is encoded in UTF-8, and in that case the ESidCP is ignored and the triplet is required. Table 11 shows the valid combinations of ESidCP and ESidUD values that are allowed when the X'50' triplet is specified on an MDR reference to a TrueType or OpenType font.

Table 11. Valid combinations of ESidUD and ESidCP in the extended X'50' Triplet on the MDR

ESidUD	ESidCP
Not specified (ignored)	X'2100' (PC-Data SBCS [ASCII])
Not specified (ignored)	X'6100' (EBCDIC SBCS)
Not specified (ignored)	X'6200' (EBCDIC DBCS)

Table 11. Valid combinations of ESidUD and ESidCP in the extended X'50' Triplet on the MDR (continued)

ESidUD	ESidCP
X'7807' (UTF-8)	Ignored

Encoding triplets on the MDR

Table 12 summarizes the use of triplets on the MDR to specify the encoding of the presentation data.

Table 12. User data encoding and MDR triplets

User Data Encoding	MDR Triplets
EBCDIC or ASCII	Fully Qualified Name (X'02') type X'85' (Code Page Name Reference) triplet **or** Font Coded Graphic Character Set (X'20') triplet
	Optional: Encoding Scheme Identifier (X'50') triplet with ESidCP = X'2100', X'6100', or X'6200'; ESidUD is ignored
UTF-8	Encoding Scheme Identifier (X'50') triplet with ESidUD = X'7807'; ESidCP is ignored

Code point conversions

Code point conversions, either from legacy EBCDIC or ASCII data to UTF-16BE, or from UTF-8 to UTF-16BE, are performed in the presentation device, for example, the printer or viewer.

Chapter 9. What AFP products have TrueType and OpenType font support?

The new TrueType and OpenType font support in AFP has been implemented as a complete end-to-end solution that includes, or will include in the future, the following products and components:

- Extensions to the AFP architecture
- AFP generating applications:
  - AFP Printer Driver for Windows
  - AFP Conversion and Indexing Facility (ACIF)
  - OS/400 DDS keyword support
  - Page Printer Formatting Aid (PPFA)
  - Vendor formatting applications that generate AFP print files
- AFP print servers:
  - PSF for z/OS
  - PSF/400
  - Infoprint Manager for AIX
  - Infoprint Manager for Windows
- AFP Viewer Plug-in
- OS/400 Host Print Transform (HPT)
- All AFCCU-based IPDS™ printers with microcode level 11.7.113 or later; including:
  - Infoprint 4000
  - Infoprint 4100
- Font Installer for AFP systems, a Java-based utility that is shipped as an optional feature of Infoprint Fonts for Multiplatforms
- WorldType® Fonts for AFP Print Servers and WorldType Fonts for AFP Clients, two sets of IBM OpenType fonts that are shipped as optional features of Infoprint Fonts for Multiplatforms
- Optional fonts shipped with OS/400

System overview

Figure 6 on page 44 shows how TrueType and OpenType fonts are supported by the various components of an AFP system.

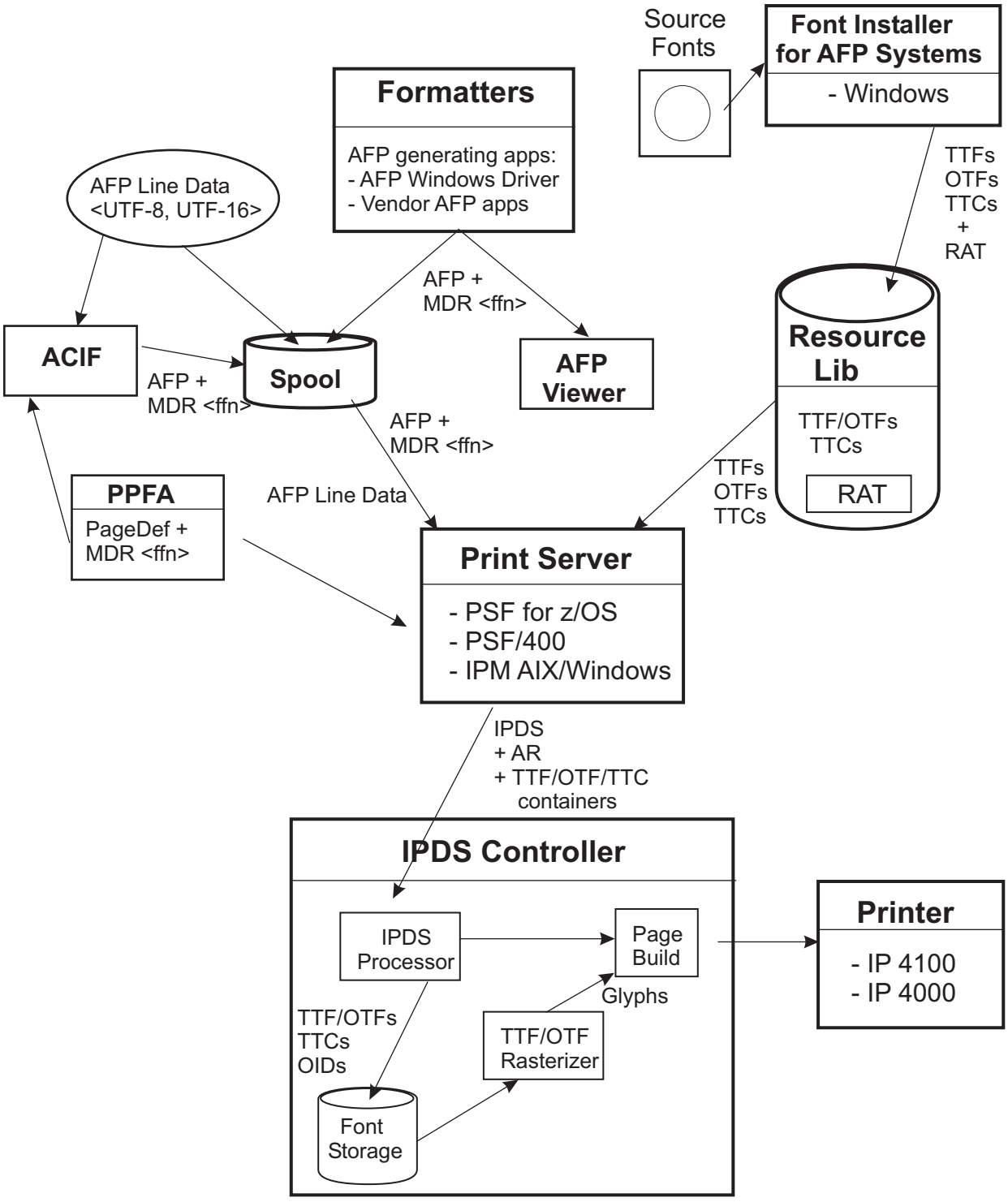


Figure 6. AFP system with TrueType and OpenType font support

AFP enablers

Applications that generate AFP include:
- AFP Printer Driver for Windows
- AFP Conversion and Indexing Facility (ACIF)
- OS/400 DDS
- Page Printer Formatting Aid (PPFA)
- Vendor formatting applications that generate AFP print files

AFP Printer Driver for Windows

The AFP Printer Driver for Windows will support TrueType and OpenType font references in the AFP text that it generates on the following platforms:
- Microsoft Windows NT®
- Microsoft Windows 2000
- Microsoft Windows XP

The character encoding will be UTF-16BE. This eliminates the need to substitute FOCA fonts for the TrueType/OpenType fonts used by Windows applications and enhances the fidelity of the AFP text generated by the Driver.

AFP Conversion and Indexing Facility (ACIF)

The ACIF application supports TrueType and OpenType font resources on the following platforms:
- AIX
- z/OS
- Microsoft Windows NT
- Microsoft Windows 2000
- Microsoft Windows XP

These resources can be collected by ACIF just as FOCA font resources are collected. The TrueType and OpenType fonts must be installed in a library that contains a Resource Access Table built by an application like the Font Installer for AFP Systems.

On z/OS, two new parameters are used to specify the user libraries and system libraries that contain TrueType/OpenType fonts:

USERPATH
: Specifies user libraries.

FONTPATH
: Specifies system libraries.

On AIX and Windows, ACIF searches the same libraries as Infoprint Manager. For more information, refer to *ACIF: User's Guide*.

OS/400 DDS

OS/400 data description specifications (DDS) use the FONTNAME keyword to support TrueType and OpenType fonts. For more information, refer to *DDS Reference: Printer Files* in the iSeries Information Center at **http://www.ibm.com/eserver/iseries/infocenter**.

Page Printer Formatting Aid (PPFA)

PPFA supports the use of TrueType and OpenType fonts on the following platforms:
- AIX

- OS/400
- VM
- VSE
- z/OS
- Microsoft Windows NT
- Microsoft Windows 2000
- Microsoft Windows XP

Support is similar to support of FOCA fonts. That is, fonts and their attributes are defined using a font definition command, then associated with presentation text.

- TrueType and OpenType fonts are defined using the font definition command DOFONT, specifying a local name, the full font name, and other attributes needed to find and render the presentation text. (FOCA fonts are defined using the FONT command.)
- PPFA references defined fonts in either of these ways:
  - The FONT subcommand of the PRINTLINE, LAYOUT, XLAYOUT, FIELD, or FIELD BARCODE presentation command references a TrueType, OpenType, or FOCA font with the local name.
  - An EXTREF command in the PAGEFORMAT creates a reference to a TrueType or OpenType font that is used in data objects, for example, GOCA objects.

For more information, refer to the *PPFA User's Guide*.

Print servers

AFP print servers include:
- PSF for z/OS
- PSF/400
- Infoprint Manager for AIX
- Infoprint Manager for Windows

PSF for z/OS

PSF for z/OS supports MO:DCA references to TrueType and OpenType fonts that reside in various resource repositories from printer-resident storage to the resource group in the print file to file system resource libraries. Fonts in file system resource libraries must be installed by the Font Installer for AFP Systems. For these fonts, the term *library* means the path or set of paths that contain the TrueType/OpenType font resources and are referred to as a path library as opposed to a Partitioned Data Set (PDS) library.

File system

TrueType and OpenType fonts may be installed in a user path library or in the default level system path library.

User path library: The user path library is identified to PSF with a USERPATH parameter on the OUTPUT JCL statement; this parameter on the OUTPUT statement is optional.

The TrueType/OpenType font repository must be a zFS or HFS. The user path library is used to locate the font object after PSF has tried locating it in the printer and inline in the print file. If PSF cannot locate the font object in the printer, inline, and in the user path library, PSF looks for it in the system path library.

System path library: The default level system path library is identified to PSF as follows:

- With DD statements in the PSF startup procedure for the desired path libraries; since the fonts reside in file systems, these DD statements have PATH parameters and not DSN parameters
- With the FONTPATH parameter on the PRINTDEV statement in the PSF startup procedure; the FONTPATH parameter references the proper path library DD statement (also in the PSF startup procedure).

The TrueType and OpenType font repository must be a zFS or HFS. The system path library is where PSF tries to find the font if PSF cannot find it in the printer, inline in the print file, or in the user path library. PSF searches the system path libraries for the TrueType or OpenType font in the order that the paths are specified in the indicated system path DD statement

Font Installer for AFP Systems

TrueType and OpenType fonts that are to be made available to PSF for z/OS in a resource library must be installed using the Font Installer for AFP Systems utility. This program is packaged and distributed as an optional feature of Infoprint Fonts for Multiplatforms. In order to enable the library to z/OS, it must be File System (FS) mounted to the z/OS environment as the font path library.

PSF/400

PSF/400 supports references to TrueType and OpenType that reside in various resource repositories from printer-resident storage to the resource group in the print file to file system resource libraries. The search order is:

1. Printer-resident storage.
2. Print file level resource group.
3. System resource libraries. For details, see "File system" on page 48.

Support for TrueType and OpenType fonts is provided in the following program temporary fixes (PTFs):

- SI13017
- SI13022
- SI13018
- SI13019
- SI13043
- SI13020
- SI13021
- SI13688

Differences between using FOCA fonts and TrueType and OpenType fonts on OS/400

Table 13 shows the differences between TrueType and OpenType fonts and FOCA fonts in how they are specified, managed, and installed.

Table 13. Differences between FOCA fonts and TrueType and OpenType fonts on OS/400

FOCA fonts	TrueType and OpenType fonts
FOCA fonts are OS/400 objects.	TrueType and OpenType fonts are stream files in the IFS.
DDS keyword FNTCHRSET is used to select a FOCA font.	DDS keyword FONTNAME is used to select a TrueType or OpenType font.

Table 13. Differences between FOCA fonts and TrueType and OpenType fonts on OS/400 (continued)

FOCA fonts	TrueType and OpenType fonts
PSF locates FOCA fonts using a defined search algorithm. The library list associated with the job is searched for a matching font.	TrueType fonts reside in one of two locations: • /QIBM/ProdData/OS400/Fonts/TTFonts (for IBM-supplied fonts) • /QIBM/UserData/OS400/Fonts/TTFonts (for user fonts)
FOCA fonts are created and managed using the font resource commands CHGCDEFNT, CHGFNTRSC, CRTFNTRSC, DLTFNTRSC, DSPCDEFNT, DSPFNTRSCA, and WRKFNTRSC.	TrueType and OpenType fonts have no OS/400 management commands. Fonts are installed using the Font Installer for AFP Systems; the generated RAT resides in the /QIBM file system.
FOCA fonts are specified by using a FNTRSC object name such as C0D0GT10.	TrueType and OpenType fonts are specified using a full font name such as **Times New Roman**.

Differences in TrueType and OpenType font processing between V5R2 and V5R3

A Resource Access Table (RAT) is not used to map full font names to file names in V5R2M0. Instead, the system maintains index tables that map the full font names specified in the MO:DCA MDR to the actual font files stored in the IFS. The index table is built for all TrueType and OpenType fonts contained in the following directories:

- QIBM/ProdData/OS400/Fonts/TTFonts
- QIBM/UserData/OS400/Fonts/TTFonts

Since there is no support for the RAT in the V5R2M0 implementation, linked fonts and font capturing are not supported in V5R2M0.

In V5R3M0, a RAT is included with the IBM-supplied fonts in /QIBM/ProdData/OS400/Fonts/TTFonts. The RAT enables these fonts to be downloaded and captured by printers. It also links the Chinese extension font to a number of the Simplified Chinese fonts (see "Optional fonts for OS/400" on page 50).

For fonts in the /QIBM/UserData/OS400/Fonts/TTFonts directory, the base V5R3M0 implementation functions exactly the same as V5R2M0. Customers who are using TrueType fonts today and do not wish to take advantage of the new function provided with a RAT do not need to change anything when they migrate from V5R2M0 to V5R3M0. However, customers can choose to step up to the full functionality of the TrueType and OpenType support, including linked fonts and font capture, simply by using the Font Installer for AFP Systems to install the fonts. This program generates a RAT when TrueType or OpenType fonts are installed, and this enables the new functionality.

File system

TrueType and OpenType fonts reside in one of two IFS directories on the iSeries:

- /QIBM/ProdData/OS400/Fonts/TTFonts for IBM fonts supplied with option 43
- /QIBM/UserData/OS400/Fonts/TTFonts for all other fonts

When searching for fonts, the UserData path is searched first, followed by the ProdData path.

On V5R3 systems, the search order for full font names follows:

1. Search the RAT, if it exists, in UserData.
2. Search the RAT, if it exists, in ProdData.
3. Search all the TrueType and OpenType font files in UserData.
4. Search all the TrueType and OpenType font files in ProdData.

This search order allows the RAT (if present) to take precedence over the V5R2M0 search method, but allows the old method to function if a RAT is not present.

Font Installer for AFP Systems

The Font Installer for AFP Systems utility can be used to install TrueType and OpenType fonts on a V5R3M0 system. This program is packaged and distributed as an optional feature of Infoprint Fonts for Multiplatforms. The V5R3M0 system software detects if a RAT is present, and if so, uses it to access fonts in the library.

Infoprint Manager for AIX and Infoprint Manager for Windows

Infoprint Manager for AIX and Infoprint Manager for Windows will support MO:DCA references to TrueType and OpenType fonts that reside in various resource repositories from printer-resident storage to the resource group in the print file to file system directories and folders. The search order is:

1. Print file level resource group
2. File system directories and folders
3. Printer-resident storage

File system

TrueType and OpenType fonts can be installed in AIX directories and in Windows folders in the standard file system. A Resource Access Table is required to access TrueType and OpenType fonts in a file system directory. Directories and folders are searched in the same order used for current FOCA fonts; for details consult *Infoprint Manager for AIX: Procedures* and *Infoprint Manager for Windows: Procedures*.

Font Installer for AFP Systems

TrueType and OpenType fonts that are to be made available to Infoprint Manager in an AIX directory or Windows folder must be installed using the Font Installer for AFP Systems. This program is packaged and distributed as an optional feature of Infoprint Fonts for Multiplatforms.

AFP Viewer Plug-in

The AFP Viewer Plug-in will support TrueType and OpenType font references on an MDR structured field in an AFP print file. The referenced fonts are rasterized using the Agfa Universal Font Scaling Technology (UFST) rasterizer. This function will be supported on the following platforms:

- Microsoft Windows 95
- Microsoft Windows NT
- Microsoft Windows 2000
- Microsoft Windows XP

The AFP Viewer Plug-in will be supported with the following browsers:

- Netscape Navigator 4.*x* and greater
- Microsoft Internet Explorer 4.*x* and greater

The AFP Viewer Plug-in uses the TrueType and OpenType fonts that are installed in folders on the Windows system and does not use a Resource Access Table to

locate these fonts. Therefore, it is not necessary to install the fonts for this product using the Font Installer for AFP Systems utility.

IPDS printers

TrueType and OpenType fonts are supported on all AFCCU-based IPDS printers with microcode level 11.7.113 or later. This includes the following printers:

- Infoprint 4000
- Infoprint 4100

WorldType Fonts for AFP Print Servers and for AFP Clients

WorldType Fonts for AFP Print Servers and WorldType Fonts for AFP Clients are optional features of IBM Infoprint Fonts for Multiplatforms. They consist of a rich set of TrueType and OpenType fonts for installation on IBM print servers or on a client workstation. Both features contain the same fonts.

Optional fonts for OS/400

Option 43, Additional Fonts, for OS/400 includes the following OpenType and TrueType fonts:

- Monotype Sans WT
- Monotype Sans WT J
- Monotype Sans WT K
- Monotype Sans WT ME
- Monotype Sans WT SC (see note)
- Monotype Sans WT TC
- Monotype Sans Duospace WT
- Monotype Sans Duospace WT J
- Monotype Sans Duospace WT K
- Monotype Sans Duospace WT ME
- Monotype Sans Duospace WT SC (see note)
- Monotype Sans Duospace WT TC
- Monotype Sans Duospace Ext B (see note)
- Times New Roman WT
- Times New Roman WT J
- Times New Roman WT K
- Times New Roman WT ME
- Times New Roman WT SC
- Times New Roman WT TC
- Thorndale™ Duospace WT
- Thorndale Duospace WT J
- Thorndale Duospace WT K
- Thorndale Duospace WT ME
- Thorndale Duospace WT SC
- Thorndale Duospace WT TC

Note: Option 43 includes a resource access table that links Monospace Sans Duospace Ext B to the base fonts Monotype Sans WT SC and Monotype Sans Duospace WT SC. The characters of Monospace Sans Duospace Ext B are thus available to documents that specify either of these base fonts.

Chapter 10. What about Unicode complex text support?

Unicode-encoded text is considered to be *complex* if it has the following formatting characteristics:

- Bidirectional rendering
- Contextual shaping
- Combined characters (for example, ligatures that are mandatory and have no equivalent Unicode code point)
- Specialized word break and justification rules

The major languages that use complex text are Arabic, Hindi, and Thai, but there are more. Complex text cannot be rendered in the traditional one-code-point-to-one glyph fashion; it requires a layout engine that examines runs of code points and maps these to runs of glyph indexes and their positions. The number of glyphs rendered may not be the same as the number of code points in the original text. Complex text support is not part of the current AFP support for TrueType and OpenType fonts and Unicode.

Text that does not require such formatting is considered to be plain text. Latin languages, such as French, German, Italian, Spanish, use ligatures, for example, German umlauts and French accents on vowels, but Unicode provides code points for the combined characters. Therefore, this is not considered to be complex text. Another way to state this is that the current support assumes that incoming Unicode has been normalized to pre-composed characters.

Note: Font kerning is an issue that is separate from combining characters. Unicode text with kerned characters is not considered to be complex text. AFP does not support kerning for TrueType or OpenType fonts.

Basis for support in AFP

Rendering Unicode complex text is no longer one-code-point-to-one-glyph rendering. It is based on two major components:

- The use of OpenType fonts
- The invocation of a layout engine that examines runs of code points and maps these to runs of glyph indices and their positions

OpenType fonts add not only support for PostScript outline technology, but also OpenType layout tables to support Unicode complex text. These tables contain script-specific information on glyph substitution, glyph positioning, justification, and baseline positioning, all of which are used by a layout engine to render complex text.

The AFP support for TrueType fonts includes support for OpenType fonts as well, so that it forms the basis for OS/400 support and any future support for Unicode complex text.

Unicode complex text support in OS/400

OS/400 V5R3 provides a prototypical implementation of AFP support of Unicode complex text. OS/400 data description specifications (DDS) use the UNISCRIPT keyword to identify fields of Unicode text for complex text layout. In addition to marking the Unicode text for complex layout, this keyword provides a number of parameters to control the layout by:

- Specifying the base direction of the field for bidirectional reordering
- Specifying an alternate inline position for right-to-left text
- Requesting normalization of Unicode text to composed forms

For more information, refer to *DDS Reference: Printer Files* in the iSeries Information Center at **http://www.ibm.com/eserver/iseries/infocenter**.

Current IPDS printers do not support the PTOCA data stream control used to mark text for complex layout. The Text Fidelity control allows users to specify how the printer should process an unsupported text control. The alternatives are:

- Stop and report a NACK.
- Skip the control sequence and continue processing the code points in the traditional one-code point to one-glyph manner.

To achieve complex layout, you must use the Host Print Transform (HPT) function to render the document. HPT performs complex text layout by interfacing with the set of layout APIs provided in OS/400 option 39, International Components for Unicode (ICU).[2] HPT generates output as raster page images, which can be sent to image-capable printers, faxes and viewers.

To correctly render the text of a given script, two related requirements must be met:

1. ICU must include a layout engine for the given script.
2. The font must include the OpenType tables used by this layout engine.

For more information on the International Components for Unicode, refer to http://oss.software.ibm.com/icu.

2. In V5R3, option 39 provides version 2.6 of ICU.

Appendix A. Using the Font Installer

The Font Installer for AFP systems allows you to manage IBM WorldType fonts and other OpenType and TrueType fonts in Microsoft Unicode format. You can:

- Install fonts and font collections in a resource library on your own computer, on a mapped network drive, or at a remote location
- Add characters to a font by linking other font files to it
- Verify the installation of fonts and links
- Uninstall fonts

The Font Installer runs on Windows 2000, but it allows you to manage fonts on any AFP system in the network.

Note: Performance on a remote system can be slow. You may prefer to perform major edits on your local system, then FTP the edited font library to the remote system.

To start the Font Installer, select **Start —> Programs —> IBM AFP Resource Installers —> Font Installer for AFP Systems**.

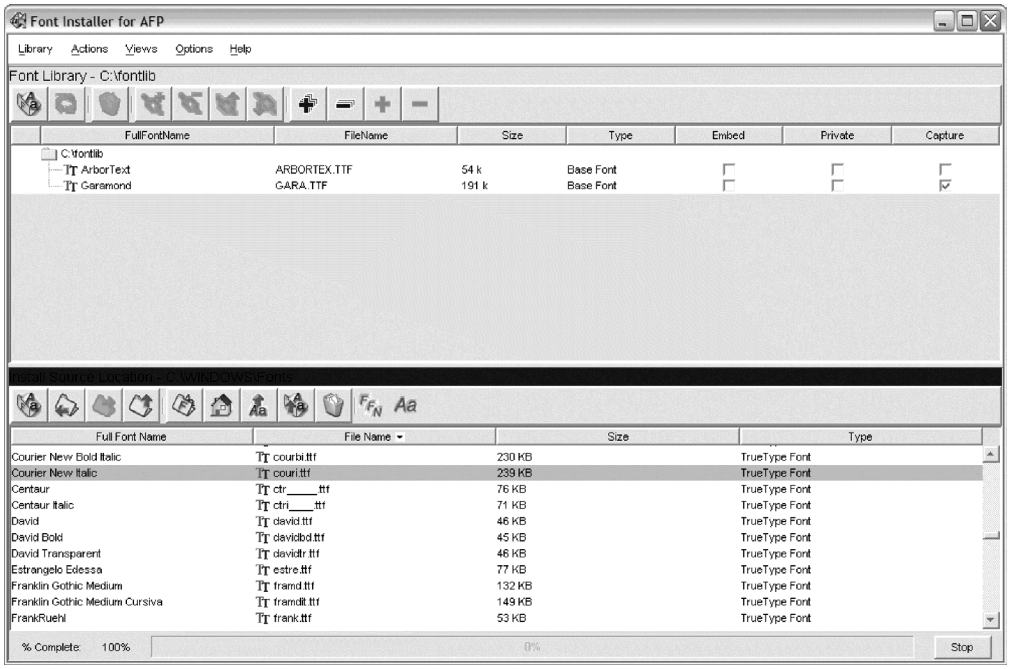


Figure 7. Font Installer for AFP Systems main window

The main window of the Font Installer has two panes.

- In the top pane, you can select the resource library where you want to install fonts or work with fonts that you have already installed.
- In the bottom pane, you can select an install source location.

Note: If you select an FTP (remote) location, Font Installer connects by default to the home directory of the current user. You can specify a different directory for the initial connection. After connecting, you can navigate to a different directory.

Use the icons or the menu bar to work with fonts. Use the Font Installer online help to guide you.

Appendix B. Is the font Unicode-enabled?

To determine if a font you want to install is Unicode-enabled:

1. Start the Font Installer for AFP Systems (see Appendix A, "Using the Font Installer," on page 53).
2. In the bottom pane of the main window, open the folder or directory where the font file is located and select the font.
3. Select **Actions –> Font Properties** to open the Font Properties dialog.

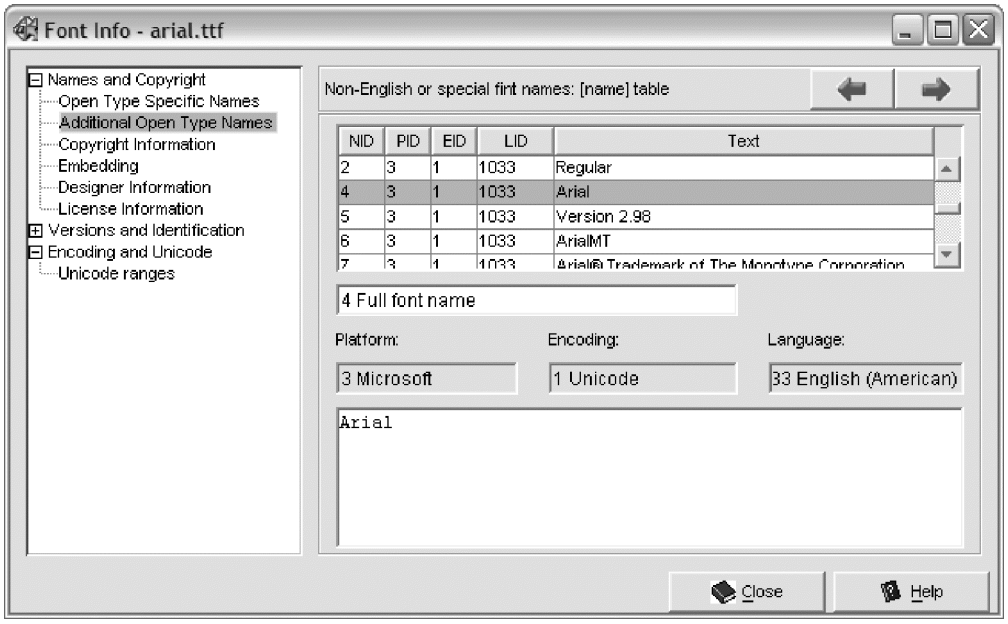


Figure 8. Font Installer for AFP Systems Font Properties dialog: Additional OpenType Names view

4. In the Additional OpenType Names view, scroll through the table until you find a record with NID 4, the Full Font Name record. Select it.
5. The fields below the table expand the columns of the selected record.
   - If the platform ID (PID) in this record is 3 (Microsoft), and the encoding ID (EID) is 1 (Unicode), go to step 6 on page 56.
   - Otherwise, look for another Full Font Name record. A single font may have several full font names. If you find one with a PID of 3 of and an EID of 1, go to step 6 on page 56.

     Note: The table records are not sorted by NID, so you may find another record 4 after several higher NIDs.
   - If you cannot find a Full Font Name record with PID of 3 of and an EID of 1, stop. The font is not Unicode-enabled. You cannot install it with the Font Installer.

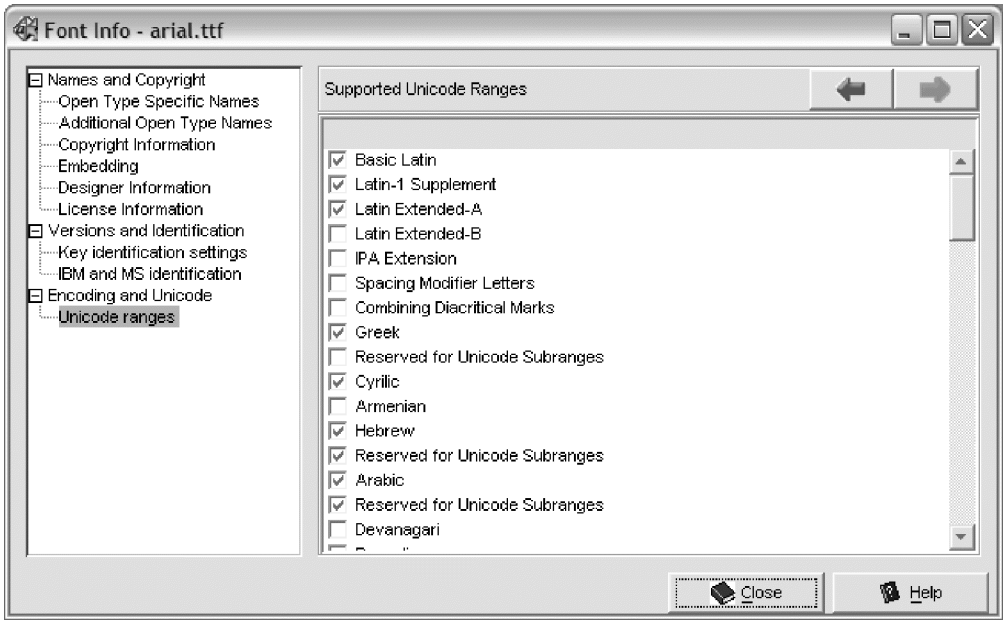


Figure 9. Font Installer for AFP Systems Font Properties dialog: Unicode Ranges view

6. In the Unicode Ranges view, look at the list of supported Unicode ranges. If any are checked, the font has Unicode cmaps. It is Unicode-enabled.

Appendix C. Related publications

For the most current information, please go to the IBM Printing Systems website at **http://www.ibm.com/printers**.

Some products provide publications in PDF format on a CD-ROM supplied with the product. You can download PDF versions of these publications or order printed copies of these publications from IBM at the following website:

http://www.elink.ibmlink.ibm.com/public/applications/publications/cgibin/pbi.cgi?CTY=US

Notes:

1. There is a fee for the printed manuals.
2. You can use the form number specified for a publication to download a PDF version or order a printed version.

Data Stream and Object Content Architectures publication library

- *Mixed Object Document Content Architecture Reference*, SC31-6802. 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 to be performed on the content. The term "mixed" in Mixed Object Document Content Architecture refers both to the mixture of data objects and the mixture of document constructs that comprise the document's components. A MO:DCA document can contain a mixture of bar code, graphics, image, and presentation text data objects. An Object Content Architecture (OCA) has been established for each data object type to define its respective syntax and semantics.
- *Font Object Content Architecture Reference*, S544-3285. This publication describes the Font Object Content Architecture (FOCA). FOCA is a resource architecture for describing the structure and content of fonts referenced by presentation data objects in the document.
- *Bar Code Object Content Architecture Reference*, S544-3766. This publication describes the Bar Code Object Content Architecture (BCOCA). BCOCA is 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.
- *AFP Programming Guide and Line Data Reference*, S544-3884. This publication describes presentation of line-mode data using the PAGEDEF print control object and other MO:DCA objects. It is for programmers who write applications that print line-mode data across the IBM system environments.
- *AFP Graphic Object Content Architecture Reference*, S544-5498. This publication contains the architecture definition of the Graphics Object Content Architecture (GOCA) for AFP. This is the version of the data architecture that is used in AFP environments 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.
- *Presentation Text Object Content Architecture Reference*, SC31-6803. This publication describes the Presentation Text Object Content Architecture (PTOCA). PTOCA is 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 Reference*, SC31-6805. This book defines the Image Object Content Architecture (IOCA). IOCA is a data architecture for describing resolution-independent image objects. The architecture specifies recording formats, data compression, color encoding, and grayscale encoding.
- *Intelligent Printer Data Stream Reference*, S544-3417. This publication describes the Intelligent Printer Data Stream™ (IPDS) architecture. The IPDS architecture defines the data stream used by print server programs and device drivers to manage all-points-addressable page printing on a full spectrum of devices from low-end workstation and local area network-attached (LAN-attached) printers to high-speed, high-volume page printers for production jobs, shared printing, and mailroom applications. The same object content architectures carried in a MO:DCA data stream can be carried in an IPDS data stream to be interpreted and presented by microcode executing in printer hardware. The IPDS architecture defines bidirectional command protocols for query, resource management, and error recovery. The IPDS architecture also provides interfaces for document finishing operations provided by preprocessing and postprocessing devices attached to IPDS printers.

Character Data Representation Architecture publication library

- *Character Data Representation Architecture: Overview*, GC09-2207. This publication provides a quick overview of the architecture, the problems it was designed to solve, and the benefits of its implementation. The various components of the architecture are highlighted.
- *Character Data Representation Architecture: Reference and Registry*, SC09-2190. This manual provides detailed information on the architecture including the coded graphic character identification mechanism and the application programming interfaces. In addition, a CD-ROM included with the publication provides many code page and character set listings and character conversion tables.

ACIF publication library

- *ACIF: User's Guide*, S544-5285. This publication tells you how to use ACIF to convert line-data applications for printing with PSF for OS/390® and z/OS, PSF/MVS, PSF/VM, PSF/VSE, Infoprint Manager for AIX, and Infoprint Manager for Windows.

PPFA publication library

- *PPFA User's Guide*, S544-5284. This publication explains how to create form definitions and page definitions for use in printing with Print Services Facility on the AIX, MVS™, VM, VSE, OS/400, and Windows platforms.
- *PPFA Quick Reference*, G544-3701. This reference manual contains the syntax for all commands, subcommands, and parameters for Page Printer Formatting Aid/370.

PSF for z/OS publication library

- *PSF for z/OS: Introduction*, G544-5625. This publication provides an overview of the functions of PSF for z/OS and how it can benefit your installation. PSF for z/OS drives IPDS (AFP) printers, manages AFP resources, and handles printer errors.
- *PSF for z/OS: Customization*, S544-5622. This publication tells the system programmer how to install, configure, tune, and maintain PSF for and z/OS.

- *PSF for z/OS: User's Guide*, S544-5630. This publication describes how to print jobs using PSF for z/OS. It describes the Job Control Language (JCL) to use and how to use AFP resources such as form definitions, page definitions, color map tables, and fonts.
- *PSF for z/OS: Download for z/OS*, S544-5624. This publication describes Download for z/OS, a feature of PSF for z/OS that transmits output data sets on the JES spool to PSF for AIX, Infoprint Manager for AIX, Infoprint Manager for Windows, and Content Manager OnDemand for printing or archiving. This publication describes how to configure the MVS, OS/390, or z/OS system, how to submit jobs, and how to diagnose errors.
- *PSF for z/OS: Messages and Codes*, G544-5627. This publication provides error messages, return codes, and abend reason codes for PSF for z/OS.
- *PSF for z/OS: Diagnosis*, G544-5623. This publication describes how to use the trace facility and contains other information that can help diagnose errors in PSF for z/OS.
- *PSF for z/OS: Security Guide*, S544-3291. This publication fulfills the U.S. government requirement for a "Security Feature User's Guide" that describes how PSF is implemented in a B1-secure environment.

The Infoprint publication library

For the most current information, see the IBM Printing Systems website at **http://www.ibm.com/printers**.

Infoprint Manager common publication library

These publications are common to both Infoprint Manager for AIX and Infoprint Manager for Windows:

- *IBM Infoprint Manager: Update Guide for PTF U483536 and PTF UR54088*, S544-5895. This publication contains information describing fixes and product enhancements available with these service updates for both Infoprint Manager for AIX Version 4 Release 1 and Infoprint Manager for Windows Version 2 Release 1. These service updates become available to customers in December 2003. Earlier versions of some of this information have been published previously in the current product library.
- *Infoprint Manager: SAP R/3 Planning and Configuration Guide*, S544-5902. For print administrators who need to create and configure both the SAP R/3 environment. IBM recommends that users possess a detailed knowledge of a SAP R/3 environment, as well as administering and maintaining either AIX or Windows operating systems. This publication describes how a business enterprise that uses SAP R/3 can use either Infoprint Manager for AIX or Infoprint Manager for Windows to manage its print environment.
- *IBM Infoprint Manager: Reference*, S544-5475. For print administrators, operators, and application programmers with the need to perform command-line functions. This publication describes the commands, utilities, transforms, attributes, and attribute values associated with Infoprint.

Infoprint Manager for AIX publication library

Infoprint Manager for AIX includes publications in PDF format on a CD-ROM supplied with the product. You can download PDF versions of these publications or order printed copies of these publications from IBM at the following website:

http://www.elink.ibmlink.ibm.com/public/applications/publications/cgibin/pbi.cgi?CTY=US

Notes:

1. There is a fee for the printed manuals.
2. You can use the form number specified for a publication to download a PDF version or order a printed version.

The publications for Infoprint Manager for AIX are:

- *IBM Infoprint Manager for AIX: Installation Instructions*, G544-5891. This publication provides updated migration and installation information from the instructions available in the *IBM Infoprint Manager for AIX: Getting Started* manual.
- *IBM Infoprint Manager for AIX: Introduction and Planning Guide*, G544-5833. This publication gives an overview of Infoprint Manager for AIX, introduces its concepts, and helps you prepare to use it. It addresses migration from previous versions of the product. Finally, this publication explains the differences between the AIX and Windows versions of Infoprint.
- *IBM Infoprint Manager for AIX: Getting Started*, G544-5817. This publication helps you install and configure IBM Infoprint Manager for AIX. It focuses primarily on server and client system software configuration.
- *Guidelines for Installing Infoprint Manager for AIX 4.1 in an HACMP Configuration*, G544-5818. This publication helps you install and configure HACMP with IBM Infoprint Manager for AIX. It also tells you how to use HACMP with IBM Infoprint Manager for AIX.
- *IBM Infoprint Manager for AIX: Configuring and Using Infoprint Fax*, G544-5915. This publication describes the primary tasks necessary for installing, configuring, and using the Infoprint Fax feature to both send and receive faxes from an Infoprint Manager for AIX server.
- *IBM Infoprint Manager for AIX: Impositioning Assist*, G544-5910. This publication describes how commercial print shops can use Infoprint Manager for AIX to make frequent layout changes to signature-based documents with relatively static content. As a prerequisite, this solution requires the Infoprint Manager for AIX Print-On-Demand feature, an IBM Infoprint 4100 Advanced Function Printing System, and a third-party impositioning product, such as ScenicSoft Preps.
- *IBM Infoprint Manager for AIX: Procedures*, G544-5815. This publication describes the primary tasks necessary for setting up an Infoprint environment, submitting print jobs, scheduling jobs, and managing the print environment.
- *PSF Direct Network Configuration Guide for System/370*, S544-5486. For system administrators or network specialists responsible for configuring a system for Print Services Facility (PSF) Direct. PSF Direct is a function of IBM Infoprint Manager for AIX that allows a PSF program (PSF for z/OS, PSF/390, PSF/400, PSF/MVS, PSF/VM, or PSF/VSE) to print remotely, using the SNA LU 6.2 protocol, on printers supported by Infoprint. The PSF program sends the print data stream directly to the Infoprint printer.
- *IBM PSF for AIX: AFP Upload Configuration Guide Using SNA Server/6000*, S544-5422. For system administrators, this publication describes how to configure AFP Upload on both AIX (client) and MVS (server) to send files to the JES spool on the MVS server, using a SNA LU 6.2 protocol.
- *IBM PSF for AIX: AFP Upload Configuration Guide Using TCP/IP*, S544-5423. For system administrators, this publication describes how to configure AFP Upload on both AIX (client) and MVS (server) to send files to the JES spool on the MVS server, using a TCP/IP protocol.
- *Infoprint Manager for AIX: Using Generalized Fax*, S544-5893. This publication describes the primary tasks necessary for configuring and using the Generalized

Fax support that is included with Infoprint Manager for AIX. This includes sending and receiving faxes from an Infoprint Manager Server.

Infoprint Manager for Windows publication library

Infoprint Manager for Windows includes publications in PDF format on a CD-ROM supplied with the product. You can download PDF versions of these publications or order printed copies of these publications from IBM at the following website:

http://www.elink.ibmlink.ibm.com/public/applications/publications/cgibin/pbi.cgi?CTY=US

Notes:

1. There is a fee for the printed manuals.
2. You can use the form number specified for a publication to download a PDF version or order a printed version.

The publications for Infoprint Manager for Windows are:

- *IBM Infoprint Manager for Windows: Introduction and Planning Guide*, G544-5716. This publication gives an overview of Infoprint Manager for Windows, introduces its concepts, and helps you prepare to use it. It describes features of this product and compares its functions to those of IBM Print Services Facility for OS/2 (PSF for OS/2). It addresses migration from PSF for OS/2 at a high level. Finally, this publication explains the differences between the AIX and Windows versions of Infoprint.
- *IBM Infoprint Manager for Windows: Getting Started*, G544-5717. This publication helps you install and configure Infoprint Manager for Windows. It focuses primarily on server and client system software configuration, but it also includes information for the IBM 4159 Model 001 S/390 Parallel Channel Emulator Card.
- *IBM Infoprint Manager for Windows: Procedures*, G544-5814. This publication contains all the Infoprint Manager for Windows procedures.

The following publications for Infoprint Manager for Windows do not have a form number and must be obtained from the Web at `www.ibm.com/printers/ipmwinlib`:

- *Infoprint Manager for Windows NT and Windows 2000: Configuring PSF Direct for an iSeries System* . This publication describes how Print Services Facility (PSF Direct) can be configured on Systems Network Architecture (SNA) networks in an iSeries operating system.
- *Infoprint Manager for Windows NT and Windows 2000: Configuring PSF Direct for the 3174 Communications Controller*. This publication describes how Print Services Facility (PSF Direct) can be configured on Systems Network Architecture (SNA) networks in a 3174 Token-Ring gateway configuration.
- *Infoprint Manager for Windows NT and Windows 2000: Configuring PSF Direct for the 3172 Communications Controller*. This publication describes how Print Services Facility (PSF Direct) can be configured on Systems Network Architecture (SNA) networks in a 3172 Token-Ring gateway configuration.
- *Infoprint Manager for Windows NT and Windows 2000: Configuring PSF Direct for the 37xx Communications Controller*. This publication describes how Print Services Facility (PSF Direct) can be configured on Systems Network Architecture (SNA) networks in either a local 37xx Token-Ring gateway configuration or remote 37xx Token-Ring gateway configuration.

Infoprint Fonts for Multiplatforms publication library

- *IBM Infoprint Fonts: Font Summary*, G544-5846. This publication summarizes the two Infoprint Font products: Infoprint Fonts for z/OS (5648-E76) and Infoprint Fonts for Multiplatforms (5648-E77). It describes font concepts, and it contains font summary tables for Expanded Core Fonts, DBCS Core Fonts, and Simulation Fonts. It also contains a table listing the code pages.
- *IBM Infoprint Fonts: Introduction to Type Transformer and Utilities for Windows*, G544-5853. Type Transformer, a part of IBM Infoprint Fonts for Multiplatforms, is a program that transforms Type 1 format and CID-keyed format outline fonts into fonts that can be used with IBM's advanced function printers. This book introduces the single-byte and double-byte parts of the Type Transformer, and it also introduces the utilities that come with Type Transformer: AFP Font Editor, DUVRMARK, and the CID to EPS transform.

iSeries publication library

You can find iSeries publications in the iSeries Information Center at **http://www.ibm.com/eserver/iseries/infocenter**.

- *DDS Reference: Printer Files*, in the iSeries Information Center. This publication provides the information for coding the data description specifications (DDS) for printer files that can be described externally.
- *iSeries: Guide to Output*, S544-5913. This publication describes how to create and print AFP documents on OS/400 systems.

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 rights 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) 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.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation
Licensing
2–31 Roppongi 3–chome, Minato-ku
Tokyo 106, Japan

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
Department 11PA Building 002S
PO Box 1900
Boulder CO 80301 USA

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee. The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurement may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

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.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information is for planning purposes only. The information herein is subject to change before the products described become available.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

For online versions of this book, we authorize you to:

- Copy, modify, and print the documentation contained on the media, for use within your enterprise, provided you reproduce the copyright notice, all warning statements, and other required statements on each copy or partial copy.
- Transfer the original unaltered copy of the documentation when you transfer the related IBM product (which may be either machines you own, or programs, if the program's license terms permit a transfer). You must, at the same time, destroy all other copies of the documentation.

You are responsible for payment of any taxes, including personal property taxes, resulting from this authorization.

Your failure to comply with the terms above terminates this authorization. Upon termination, you must destroy your machine readable documentation.

Trademarks

The following terms, used in this publication, are trademarks or registered trademarks of the IBM Corporation in the United States or other countries or both:

- Advanced Function Presentation
- Advanced Function Printing
- AFCCU
- AFP
- AIX®
- Bar Code Object Content Architecture
- BCOCA
- IBM®
- InfoColor®
- Infoprint®
- Intelligent Printer Data Stream
- IPDS
- iSeries
- Mixed Object Document Content Architecture
- MO:DCA
- MVS
- OS/2®
- OS/390®
- OS/400®
- Print Services Facility
- RS/6000®
- S/370
- S/390®
- System/370
- z/OS®

Adobe Caslon and Adobe Garamond are trademarks of Adobe Systems Incorporated which may be registered in certain jurisdictions.

Agfa is a trademark of Agfa Corporation or its affiliates, which may be registered in certain jurisdictions.

Apple® and TrueType are registered trademarks of Apple Computer, Inc.

Arial, Centaur, Courier New, Gill Sans, Monotype Sans Duospace, and Times New Roman are trademarks of The Monotype Corporation and may be registered in certain jurisdictions.

Frutiger, Helvetica, Optima, and Palatino are all trademarks of Linotype-Hell AG and/or its subsidiaries 1981, all rights reserved.

Futura is a registered trademark of Fundicion Tipográfica Neufville.

ITC, ITC Garamond, and ITC Zapf Dingbats are registered trademarks of International Typeface Corporation.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Lucida is a registered trademark of Bigelow & Holmes.

Microsoft, OpenType, Windows, and Windows NT are trademarks of Microsoft Corporation in the United States, other countries, or both.

Monotype, Thorndale, and WorldType are trademarks of Agfa Monotype Corporation.

Unicode is a registered trademark of Unicode Incorporated.

Univers is a registered trademark of Haas Typefoundry Ltd.

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

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

Glossary

This glossary contains terms that apply to the AFP architecture and systems that implement this architecture.

If you do not find the term that you are looking for, please refer to the *IBM Dictionary of Computing*, ZC20-1699, or the *IBM Dictionary of Printing*, G544-3973.

The following definitions are provided as supporting information only, and are not intended to be used as a substitute for the semantics described in the body of this reference.

B

base font. A font that is mapped in the RAT.

basic multilingual plane (BMP). In the Unicode encoding space, Plane 0. The BMP is addressed with two-byte UTF-16 code points in the range U+0000 to U+FFFF. It contains all non-surrogate characters and a surrogate block for addressing the supplemental planes.

BMP. Basic multilingual plane.

C

CCSID. Coded Character Set Identifier.

character. A member of a set of elements that is used for the representation, organization, or control of data. Characters can be letters, digits, punctuation marks, or other symbols represented in the form of a spatial arrangement of adjacent or connected strokes or in the form of other physical conditions in data media. In architecture, a character can be either a *graphic character* or a *control character*.

character identifier. The identifier that represents a character, regardless of its style; for example, all uppercase A's have the same character identifier. See also *graphic character identifier*.

character map (cmap). In TrueType fonts, a mapping of code points to glyph indices, which are used to index the actual character shape information. A cmap may contain multiple subtables, each defining a platform-specific encoding.

cmap. Character map.

Coded Character Set Identifier (CCSID). A 16-bit number identifying a specific set of encoding scheme identifier, character set identifiers, code page identifiers and other relevant information that uniquely identifies the coded graphic character representation used. See also *Coded Graphic Character Set Global Identifier (CGCSGID)*.

coded graphic character set. A set of graphic characters with their assigned code points.

Coded Graphic Character Set Global Identifier (CGCSGID). A four-byte binary or a ten-digit decimal identifier consisting of the concatenation of a GCSGID and a CPGID. The CGCSGID identifies the code point assignments in the code page for a specific graphic character set, from among all the graphic characters that are assigned in the code page. See also *Coded Character Set Identifier (CCSID)*.

code page. A resource object containing descriptive information, graphic character identifiers, and code points corresponding to a coded graphic character set. See also *coded graphic character set*. A set of assignments, each of which assigns a code point to a character. Each code page has a unique name or identifier. Within a given code page, a code point is assigned to one character. More than one character set can be assigned code points from the same code page. See also *code point*.

Code Page Global Identifier (CPGID). A unique code page identifier that can be expressed as either a two-byte binary or a five-digit decimal value.

code point. A unique bit pattern that can serve as an element of a code page or a site in a code table, to which a character can be assigned. The element is associated with a binary value. The assignment of a character to an element of a code page determines the binary value that will be used to represent each occurrence of the character in a character string. Code points are one or more bytes long. See also *code page*.

complex text. Unicode-encoded text that cannot be rendered in the traditional one-code-point-to-one glyph fashion; for example, bidirectional text as in Arabic or combined characters as in Hindi.

control character. A character whose occurrence in a particular context specifies a control function. A control character starts, changes, or stops any operation that affects recording, processing, transmitting, or interpreting data. Examples are line-feed, shift-in, and shift-out, and, in printing, carriage return, font change, and end of transmission. In architecture, a control character can be recorded for use in a subsequent action, and it can have a graphic representation.

CPGID. Code Page Global Identifier.

D

data-object font. A complete non-FOCA font resource that is analogous to a coded font. Examples of data-object fonts are TrueType fonts and OpenType fonts.

DBCS. Double-byte character set.

display order. The order in which characters are rendered. Especially in bidirectional languages, it may not be the same as logical order.

double-byte character set (DBCS). A character set that can contain up to 65536 characters.

double-byte coded font. A coded font in which the code points are two bytes long.

E

EBCDIC. Extended Binary-Coded Decimal Interchange Code.

EID. Encoding identifier.

EncEnv. Platform identifier.

EncID. Encoding identifier.

encoding identifier (EID or EncID). A code that identifies the encoding of a string in the font naming table.

encoding scheme. A set of specific definitions that describe the philosophy used to represent character data. The number of bits, the number of bytes, the allowable ranges of bytes, the maximum number of characters, and the meanings assigned to some generic and specific bit patterns, are some examples of specifications to be found in such a definition.

Encoding Scheme Identifier (ESID). A 16-bit number assigned to uniquely identify a particular encoding scheme specification. See also *encoding scheme*.

ESID. Encoding Scheme Identifier.

Extended Binary-Coded Decimal Interchange Code (EBCDIC). A coded character set that consists of eight-bit coded characters.

F

FGID. Font Global Identifier.

FOCA. Font Object Content Architecture.

font. A set of graphic characters that have a characteristic design, or a font designer's concept of how the graphic characters should appear. The characteristic design specifies the characteristics of its graphic characters. Examples of characteristics are shape, graphic pattern, style, size, weight, and increment. Examples of fonts are fully described fonts, symbol sets, and their internal printer representations. See also *coded font* and *symbol set*.

font character set. A FOCA resource containing descriptive information, font metrics, and the digital representation of character shapes for a specified graphic character set.

Font Global Identifier (FGID). A unique font identifier that can be expressed as either a two-byte binary or a five-digit decimal value. The FGID is used to identify a type style and the following characteristics or parameters: posture, weight, and width.

font height (FH). Synonymous with *vertical font size*.

font inline sequence. The clockwise rotation of the inline direction relative to a character pattern.

font local identifier. A binary identifier that is mapped by the environment to a named resource to identify a coded font. See also *local identifier*.

font metrics. Measurement information that defines individual character values such as height, width, and space, as well as overall font values such as averages and maximums. Font metrics can be expressed in specific fixed units, such as pels, or in relative units that are independent of both the resolution and the size of the font.

font naming table. A table in the font object that defines many font properties, including the names used to identify the font in various contexts.

font object. A resource object that contains some or all of the description of a font.

Font Object Content Architecture (FOCA). An architected collection of constructs used to describe fonts and to interchange those font descriptions.

font referencing. A method used to identify or characterize a font. Examples of processes that use font referencing are document editing, formatting, and presentation.

font width (FW). Synonymous with *horizontal font size*.

full font name. The name that the Windows operating systems uses to identify a font. It is in the naming table of the font file.

FW. Font width.

G

GB18030. A legal standard of the People's republic of China that requires support of Coded Character Set ID

1386, plus 6582 Unicode Extension-A, plus 1,948 additional non-Han characters (Mongolian, Uygur, Tibetan, and Yi).

GCGID. Graphic Character Global Identifier.

GCSGID. Graphic Character Set Global Identifier.

GID. Global identifier.

Global Resource Identifier (GRID). An eight-byte identifier that identifies a coded font resource. A GRID contains the following fields in the order shown:

1. GCSGID of a minimum set of graphic characters required for presentation. It can be a character set that is associated with the code page, or with the font character set, or with both.
2. CPGID of the associated code page
3. FGID of the associated font character set
4. Font width in 1440ths of an inch.

glyph. A member of a set of symbols that represent data. Glyphs can be letters, digits, punctuation marks, or other symbols. Synonymous with *graphic character*. See also *character*.

GOCA. Graphics Object Content Architecture.

graphic character. A member of a set of symbols that represent data. Graphic characters can be letters, digits, punctuation marks, or other symbols. Synonymous with *glyph*. See also *character*.

Graphic Character Global Identifier (GCGID). An alphanumeric character string used to identify a specific graphic character. A GCGID can be from four bytes to eight bytes long.

graphic character identifier. The unique name for a graphic character in a font or in a graphic character set. See also *character identifier*.

Graphic Character Set Global Identifier (GCSGID). A unique graphic character set identifier that can be expressed as either a two-byte binary or a five-digit decimal value.

Graphics Object Content Architecture (GOCA). An architected collection of constructs used to interchange and present graphics data.

GRID. Global Resource Identifier.

I

Intelligent Printer Data Stream (IPDS). An architected host-to-printer data stream that contains both data and controls defining how the data is to be presented.

IPDS. Intelligent Printer Data Stream.

italics. A typeface with characters that slant upward to the right. In FOCA, italics is the common name for the defined inclined typeface posture attribute or parameter.

K

Kanji. A graphic character set for symbols used in Japanese ideographic alphabets.

kerning. The design of graphic characters so that their character boxes overlap, resulting in the reduction of space between characters. This allows characters to be designed for cursive languages, ligatures, and proportionally-spaced fonts. An example of kerning is the printing of adjacent graphic characters so they overlap on the left or right side.

L

language. A set of symbols, conventions, and rules that is used for conveying information.

LCID. Language identifier.

language identifier (LCID or LID). A code that identifies the language of a record in the font naming table.

LID. (1) Language identifier. (2) Local identifier.

ligature. A single glyph representing two or more characters. Examples of characters that can be presented as ligatures are *ff* and *ffi*.

linked font. A font that is processed as an extension of a base font.

local identifier (LID). An identifier that is mapped by the environment to a named resource.

logical order. The order in which characters are typed on a keyboard. Especially in bidirectional languages, it may not be the same as display order. Synonymous with *memory order*, *storage order*.

logical unit (L-unit). A unit of linear measurement expressed with a unit base and units per unit-base value. For example, in MO:DCA and IPDS architectures, the following logical units are used:

- 1 logical unit = 1/1440 inch (unit base = 10 inches, units per unit base = 14400)
- 1 logical unit = 1/240 inch (unit base = 10 inches, units per unit base = 2400)

lowercase. Pertaining to small letters as distinguished from capital letters. Examples of small letters are *a*, *b*, and *g*. Contrast with *uppercase*.

L-unit. Logical unit.

M

memory order. Synonym for *logical order*.

Mixed Object Document Content Architecture (MO:DCA). An architected, device-independent data stream for interchanging documents.

MO:DCA. Mixed Object Document Content Architecture.

MO:DCA-P. The subset of the MO:DCA architecture that defines presentation documents.

monospaced font. A font with graphic characters having a uniform character increment. The distance between reference points of adjacent graphic characters is constant in the escapement direction. The blank space between the graphic characters can vary. Synonymous with *uniformly spaced font*. Contrast with *proportionally spaced font* and *typographic font*.

N

name identifier (NID). The index to the font naming table.

naming table. Font naming table.

NID. Name identifier.

O

object. A collection of structured fields. The first structured field provides a begin-object function and the last structured field provides an end-object function. The object can contain one or more other structured fields whose content consists of one or more data elements of a particular data type. An object can be assigned a name, which can be used to reference the object. Examples of objects are image, graphics, text, page segment, and document index objects.

object identifier (OID). A notation for assigning globally-unambiguous names. The notation is defined in international standard ISO/IEC 8824:1990(E).

OID. Object identifier.

OpenType font (OTF). An extension of the TrueType font format that adds:
- Support for PostScript outlines
- Better support for international character sets
- Broader support for advanced typographic control

OTF. OpenType font.

P

PID. Platform identifier.

plane. In Unicode, a group of 65,536 code points. See *basic multilingual plane (BMP)*, *supplemental plane*.

platform identifier (PID). A code in the font naming table that identifies the platform that supports the font.

point. A unit of measure used mainly for measuring typographical material. There are seventy-two points to an inch.

posture. Inclination of a letter with respect to a vertical axis. Examples of inclination are upright and inclined. An example of upright is Roman. An example of inclined is italics.

presentation text object. An object that contains presentation text data. See also *object*.

Presentation Text Object Content Architecture (PTOCA). An architected collection of constructs used to interchange and present presentation text data.

programming request for price quotation (PRPQ). A custom feature of an IBM program product.

proportion. Relationship of the width of a letter to its height.

proportional spacing. The spacing of characters in a printed line so that each character is allotted a space based on the character's width.

proportionally spaced font. A font with graphic characters that have varying character increments. Proportional spacing can be used to provide the appearance of even spacing between presented characters and to eliminate excess blank space around narrow characters. An example of a narrow character is the letter *i*. Synonymous with *typographic font*. Contrast with *monospaced font* and *uniformly spaced font*.

PRPQ. Programming request for price quotation.

PTOCA. Presentation Text Object Content Architecture.

R

RAT. Resource access table.

request for price quotation (RPQ). A custom feature of an IBM product.

resource. An object that is referenced by a data stream or by another object to provide data or information. Resource objects can be stored in libraries. In MO:DCA, resource objects can be contained within a resource group. Examples of resources are fonts, overlays, and page segments.

resource access table (RAT). A table in a resource library that maps resource names specified in the

MO:DCA data stream to information used to find and process the resource on a given system.

resource library. A library, folder, or directory that contains resources.

RPQ. Request for price quotation.

S

sans serif. A type style characterized by strokes that end with no flaring or crossing of lines at the stroke-ends. Contrast with *serif*.

SBCS. Single-byte character set.

serif. A short line angling from or crossing the free end of a stroke. Examples are horizontal lines at the tops and bottoms of vertical strokes on capital letters, for example, *I* and *H*, and the decorative strokes at the ends of the horizontal members of a capital *E*. Contrast with *sans serif*.

shape technology. A method used to encode character shapes digitally using a specified algorithm.

single-byte character set (SBCS). A character set that can contain up to 256 characters.

single-byte coded font. A coded font in which the code points are one byte long.

storage order. Synonym for *logical order*.

supplemental plane. Any of Unicode planes 1–16. Supplemental planes are addressed with UTF-16 surrogates.

surrogates. Pairs of Unicode code points that allow for the encoding of as many as 1 million additional characters without any use of escape codes.

T

TrueType collection (TTC). Multiple TrueType or OpenType fonts that are contained in a single file structure. When such fonts share many glyph sets, significant file space can be saved by packaging them in a TTC.

TTC. TrueType collection.

TrueType font (TTF). A font format based on scalable outline technology with flexible hinting. Glyph shapes are based on quadratic curves. The font is described with a set of tables contained in a TrueType Font File.

TTF. TrueType font.

typeface. All characters of a single type family or style, weight class, width class, and posture, regardless of size. For example, Sonoran Serif, Bold, Normal, Italics, in any point size.

type family. All characters of a single design, regardless of attributes such as width, weight, posture, and size. Examples are Courier and Gothic.

type structure. Attributes of characters other than type family or typeface. Examples are solid shape, hollow shape, and overstruck.

type style. The form of characters within the same font, for example, Courier or Gothic.

type weight. A parameter indicating the degree of boldness of a typeface. A character's stroke thickness determines its type weight. Examples are light, medium, and bold. Synonymous with *weight class*.

type width. A parameter indicating a relative change from the font's normal width-to-height ratio. Examples are normal, condensed, and expanded. Synonymous with *width class*.

typographic font. A font with graphic characters that have varying character increments. Proportional spacing can be used to provide the appearance of even spacing between presented characters and to eliminate excess blank space around narrow characters. An example of a narrow character is the letter *i*. Synonymous with *proportionally spaced font*. Contrast with *monospaced font* and *uniformly spaced font*.

U

Unicode. A character encoding standard for information processing that includes all major scripts of the world. Unicode defines a consistent way of encoding multilingual text. Unicode specifies a numeric value, a name, and other attributes—such as directionality—for each of its characters; for example, the name for $ is "dollar sign" and its numeric value is X'0024'. This Unicode value is called a *Unicode code point* and is represented as U+0024. Unicode provides for three encoding forms: UTF-8, UTF-16, and UTF-32.

Unicode-enabled font. A TrueType or OpenType font that contains a Microsoft Unicode subtable identified by platform ID = 3 (Microsoft), and platform-specific encoding ID= 1 (Unicode, UTF-16), and that specifies a full font name (Name ID 4) using the same encoding in the naming table.

Unicode Migration Fonts RPQ. A custom group of FOCA-based fonts that support UTF-16 encoding in user presentation data.

uniformly spaced font. A font with graphic characters having a uniform character increment. The distance between reference points of adjacent graphic characters is constant in the escapement direction. The blank space between the graphic characters can vary. Synonymous with *monospaced font*. Contrast with *proportionally spaced font* and *typographic font*.

uppercase. Pertaining to capital letters. Examples of capital letters are *A*, *B*, and *C*. Contrast with *lowercase*.

UTF-8. A byte-oriented form that is designed for ease of use in traditional ASCII environments. Each UTF-8 code point contains from one to four bytes. All Unicode code points can be encoded in UTF-8 and all 7-bit ASCII characters can be encoded in one byte.

UTF-16. The default Unicode encoding. A fixed, two-byte Unicode encoding form that can contain surrogates and identifies the byte order of each UTF-16 code point via a Byte Order Mark in the first 2 bytes of the data.

UTF-16BE. UTF-16 that uses big endian byte order; this is the byte order for all multi-byte data within AFP data streams. The Byte Order Mark is not necessary when the data is externally identified as UTF-16BE (or UTF-16LE).

UTF-16LE. UTF-16 that uses little endian byte order.

UTF-32. A fixed, four-byte Unicode encoding form in which each UTF-32 code point is precisely identical to the Unicode code point.

UTF-32BE. UTF-32 serialized as bytes in most significant byte first order (big endian). UTF-32BE is structurally the same as UCS-4.

UTF-32LE. UTF-32 serialized as bytes in least significant byte first order (little endian).

V

vertical font size. A characteristic value, perpendicular to the character baseline, that represents the size of all graphic characters in a font. Synonymous with *font height*. In a font character set, nominal vertical font size is a font-designer defined value corresponding to the nominal distance between adjacent baselines when character rotation is zero degrees and no external leading is used. This distance represents the baseline-to-baseline increment that includes the font's maximum baseline extent and the designer's recommendation for internal leading. The font designer can also define a minimum and a maximum vertical font size to represent the limits of scaling. In font referencing, the specified vertical font size is the desired size of the font when the characters are presented. If this size is different from the nominal vertical font size specified in a font character set, the character shapes and character metrics might need to be scaled prior to presentation.

W

weight class. A parameter indicating the degree of boldness of a typeface. A character's stroke thickness determines its weight class. Examples are light, medium, and bold. Synonymous with *type weight*.

width class. A parameter indicating a relative change from the font's normal width-to-height ratio. Examples are normal, condensed, and expanded. Synonymous with *type width*.

Index

T

U

W

Readers' Comments — We'd Like to Hear from You

Printing Systems Division
Using OpenType Fonts in an AFP System

Publication No. G544-5876-02

Overall, how satisfied are you with the information in this book?

	Very Satisfied	Satisfied	Neutral	Dissatisfied	Very Dissatisfied
Overall satisfaction	☐	☐	☐	☐	☐

How satisfied are you that the information in this book is:

	Very Satisfied	Satisfied	Neutral	Dissatisfied	Very Dissatisfied
Accurate	☐	☐	☐	☐	☐
Complete	☐	☐	☐	☐	☐
Easy to find	☐	☐	☐	☐	☐
Easy to understand	☐	☐	☐	☐	☐
Well organized	☐	☐	☐	☐	☐
Applicable to your tasks	☐	☐	☐	☐	☐

Please tell us how we can improve this book:

Thank you for your responses. May we contact you? ☐ Yes ☐ No

When you send comments to IBM, you grant IBM a nonexclusive right to use or distribute your comments in any way it believes appropriate without incurring any obligation to you.

Name

Address

Company or Organization

Phone No.

Readers' Comments — We'd Like to Hear from You
G544-5876-02

Cut or Fold
Along Line

Fold and Tape | **Please do not staple** | Fold and Tape

NO POSTAGE
NECESSARY
IF MAILED IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST-CLASS MAIL PERMIT NO. 40 ARMONK, NEW YORK

POSTAGE WILL BE PAID BY ADDRESSEE

IBM Corporation
IBM Printing Systems Division
Department H7FE, Building 004N
Information Development
P.O. Box 1900
Boulder, CO USA 80301-9817

Fold and Tape | **Please do not staple** | Fold and Tape

G544-5876-02

Cut or Fold
Along Line

Program Number:

Printed in USA

G544-5876-02