Advanced Function Presentation Consortium
Data Stream and Object Architectures

Metadata Object Content Architecture Reference

AFPC-0013-01

AFPConsortium
Advanced Function Presentation

Note

Before using this information, read the information in "Notices" on page 17.

First Edition (July 2014)

This edition applies to the Metadata Object Content Architecture. It is the first edition, and is produced by the AFP Consortium™ (AFPC™). This edition remains current until a new edition is published.

Internet

Visit our home page, where this publication and others are available:
`http://www.afpcinc.org`

Preface

This book defines the Metadata Object Content Architecture (MOCA).

This book is a reference, not a tutorial, and does not describe any product implementation of the architecture.

Who Should Read This Book

This book is for system programmers and other developers who need such information to develop or adapt a product or program to interoperate with other products in an Advanced Function Presentation™ (AFP™) environment.

AFP Consortium (AFPC)

The Advanced Function Presentation (AFP) architectures began as the strategic, general purpose document and information presentation architecture for the IBM® Corporation. The first specifications and products go back to 1984. Although all of the components of the architecture have grown over the years, the major concepts of object-driven structures, print integrity, resource management, and support for high print speeds were built in from the start.

In the early twenty-first century, IBM saw the need to enable applications to create color output that is independent from the device used for printing and to preserve color consistency, quality, and fidelity of the printed material. This need resulted in the formation, in October 2004, of the AFP Color Consortium™ (AFPCC™). The goal was to extend the object architectures with support for full-color devices including support for comprehensive color management. The idea of doing this via a consortium consisting of the primary AFP architecture users was to build synergism with partners from across the relevant industries, such as hardware manufacturers that produce printers as well as software vendors of composition, work flow, viewer, and transform tools. Quickly more than 30 members came together in regular meetings and work group sessions to create the AFP Color Management Object Content Architecture™ (CMOCA™). A major milestone was reached by the AFP Color Consortium with the initial official release of the CMOCA specification in May 2006.

Since the cooperation between the members of the AFP Color Consortium turned out to be very effective and valuable, it was decided to broaden the scope of the consortium efforts and IBM soon announced its plans to open up the complete scope of the AFP architecture to the consortium. In June 2007, IBM's role as founding member of the consortium was transferred to the InfoPrint® Solutions Company, an IBM/Ricoh® joint venture. In February 2009, the consortium was incorporated under a new set of bylaws with tiered membership and shared governance resulting in the creation of a formal open standards body called the AFP Consortium (AFPC). Ownership of and responsibility for the AFP architectures was transferred at that time to the AFP Consortium.

How to Use This Book

This document is divided into five chapters:

- Chapter 1, “A Presentation Architecture Perspective”, on page 1 introduces the AFPC presentation architectures and describes the role of data streams and data objects.
- Chapter 2, “Introduction to MOCA”, on page 7 introduces the goals and scope of the Metadata Object Content Architecture.
- Chapter 3, “Metadata Concepts”, on page 9 discusses metadata concepts as they relate to AFP.
- Chapter 4, “Metadata Object (MO)”, on page 11 specifies the MO syntax and semantics.
- Chapter 5, “MO Header Attributes”, on page 15 registers and describes the supported MO types, formats, and compression.

The “Glossary” on page 19 defines some of the terms used in this book.

How to Read the Syntax Diagrams

The basic data types used in the Metadata Object Content Architecture (MOCA) are:

UBIN	Unsigned binary
UTF16	UTF-16BE characters
UNDF	Undefined type

The following notation conventions apply to the MO data structures.

- Each byte contains eight bits.
- Bytes of an MO data structure are numbered beginning with byte 0. For example, a two-byte field followed by a one-byte field would be numbered as follows:

Bytes 0–1	Field 1
Byte 2	Field 2

- Field values are expressed in hexadecimal or binary notation:

  X'7FFF' = +32,767

  B'0001' = 1

- Some bits or bytes are labeled *reserved*. The content of reserved fields is not checked by MO receivers. However, MO generators should set reserved fields to the specified value.

Related Publications

Several other publications can help you understand the architecture concepts described in this book. AFP Consortium publications and a few other AFP publications are available on the AFP Consortium website, `http://www.afpcinc.org`.

Table 1. AFP Consortium Architecture References

AFP Architecture Publication	Order Number
AFP Programming Guide and Line Data Reference	S544-3884
Bar Code Object Content Architecture™ Reference	AFPC-0005
Color Management Object Content Architecture Reference	AFPC-0006
Font Object Content Architecture Reference	S544-3285
Graphics Object Content Architecture for Advanced Function Presentation Reference	AFPC-0008
Image Object Content Architecture Reference	AFPC-0003
Intelligent Printer Data Stream™ Reference	AFPC-0001
Metadata Object Content Architecture Reference	AFPC-0013
Mixed Object Document Content Architecture™ (MO:DCA™) Reference	AFPC-0004
Presentation Text Object Content Architecture Reference	SC31-6803

Table 2. Additional AFP Consortium Documentation

AFPC Publication	Order Number
AFP Color Management Architecture™ (ACMA™)	AFPC-0015
AFPC Company Abbreviation Registry	AFPC-0012
AFPC Font Typeface Registry	AFPC-0016
BCOCA™ Frequently Asked Questions	AFPC-0011
MO:DCA-L: The OS/2 PM Metafile (.met) Format	AFPC-0014
Presentation Object Subsets for AFP	AFPC-0002
Recommended IPDS™ Values for Object Container Versions	AFPC-0017

Table 3. Other Documentation

Publication
XMP (Extensible Metadata Platform) Specification, September 2005
Efficient XML Interchange (EXI) Format 1.0
RFC 1952 - GZIP file format specification version 4.3

Contents

Figures

Tables

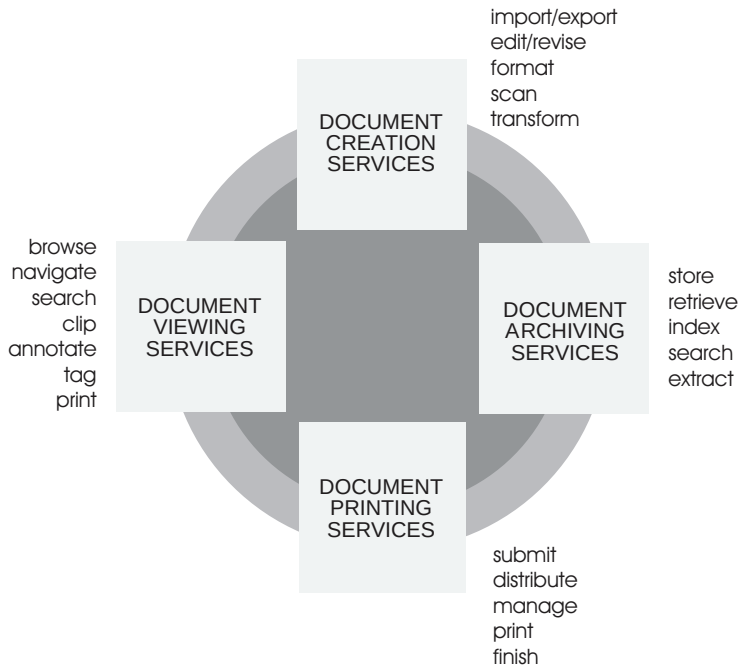
Chapter 1. A Presentation Architecture Perspective

This chapter provides a brief overview of Presentation Architecture.

The Presentation Environment

Figure 1 shows today's presentation environment.

Figure 1. Presentation Environment. The environment is a coordinated set of services architected to meet the presentation needs of today's applications.



The ability to create, store, retrieve, view, and print data in presentation formats friendly to people is a key requirement in almost every application of computers and information processing. This requirement is becoming increasingly difficult to meet because of the number of applications, servers, and devices that must interoperate to satisfy today's presentation needs.

The solution is a presentation architecture base that is both robust and open ended, and easily adapted to accommodate the growing needs of the open system environment. AFP presentation architectures provide that base by defining interchange formats for data streams and objects that enable applications, services, and devices to communicate with one another to perform presentation functions. These presentation functions might be part of an integrated system solution or they might be totally separated from one another in time and space. AFP presentation architectures provide structures that support object-oriented models and client/server environments.

AFP presentation architectures define interchange formats that are system independent and are independent of any particular format used for physically transmitting or storing data. Where appropriate, AFP presentation architectures use industry and international standards, such as the ITU-TSS (formerly known as CCITT) facsimile standards for compressed image data.

Architecture Components

AFP presentation architectures provide the means for representing documents in a data format that is independent of the methods used to capture or create them. Documents can contain combinations of text, image, graphics, and bar code objects in device-independent and resolution-independent formats. Documents can contain fonts, overlays, and other resource objects required at presentation time to present the data properly. Finally, documents can contain resource objects, such as a document index and tagging elements supporting the search and navigation of document data, for a variety of application purposes.

The presentation architecture components are divided into two major categories: *data streams* and *objects*.

Data Streams

A *data stream* is a continuous ordered stream of data elements and objects conforming to a given format. Application programs can generate data streams destined for a presentation service, archive library, presentation device, or another application program. The strategic presentation data stream architectures are:

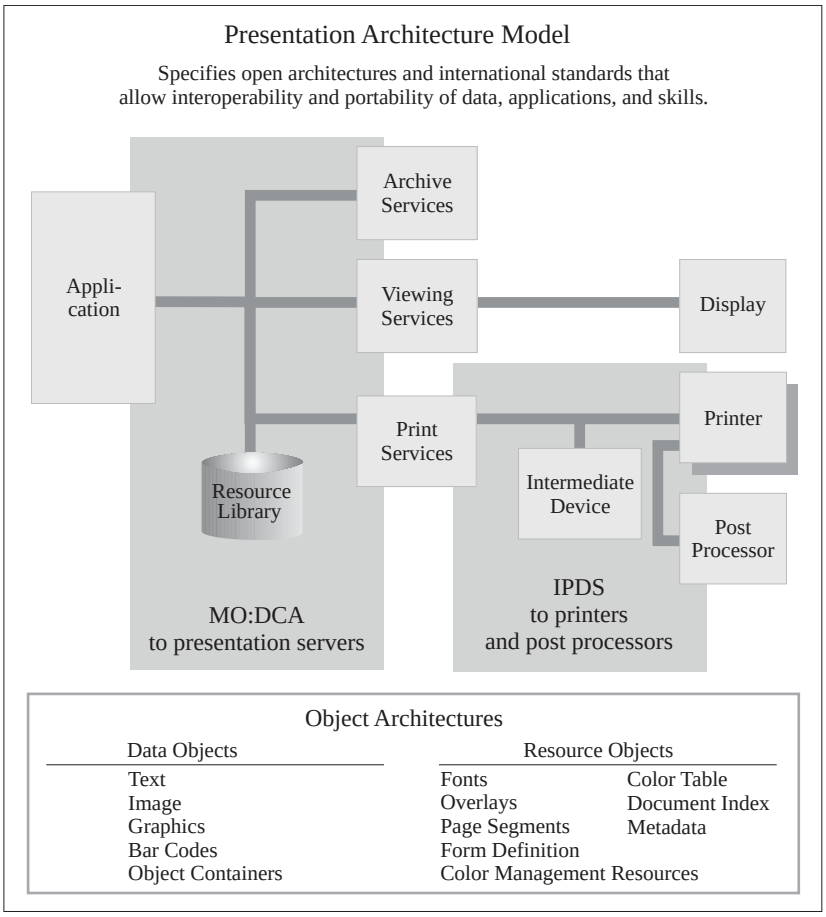
- *Mixed Object Document Content Architecture (MO:DCA)*
- *Intelligent Printer Data Stream (IPDS) Architecture*.

The MO:DCA architecture defines the data stream used by applications to describe documents and object envelopes for interchange with other applications and application services. Documents defined in the MO:DCA format can be archived in a database, then later retrieved, viewed, annotated, and printed in local or distributed systems environments. Presentation fidelity is accommodated by including resource objects in the documents that reference them.

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 pre-processing and post-processing devices attached to IPDS printers.

Figure 2 shows a system model relating MO:DCA and IPDS data streams to the presentation environment previously described. Also shown in the model are the object content architectures that apply to all levels of presentation processing in a system.

Figure 2. Presentation Model. This diagram shows the major components in a presentation system and their use of data stream and object architectures.



Objects

Documents can be made up of different kinds of data, such as text, graphics, image, and bar code. *Object content architectures* describe the structure and content of each type of data format that can exist in a document or appear in a data stream. Objects can be either *data objects* or *resource objects*.

A data object contains a single type of presentation data, that is, presentation text, vector graphics, raster image, or bar codes, and all of the controls required to present the data.

A resource object is a collection of presentation instructions and data. These objects are referenced by name in the presentation data stream and can be stored in system libraries so that multiple applications and the print server can use them.

All object content architectures (OCAs) are totally self-describing and independently defined. When multiple objects are composed on a page, they exist as peer objects that can be individually positioned and manipulated to meet the needs of the presentation application.

The AFPC-defined object content architectures are:

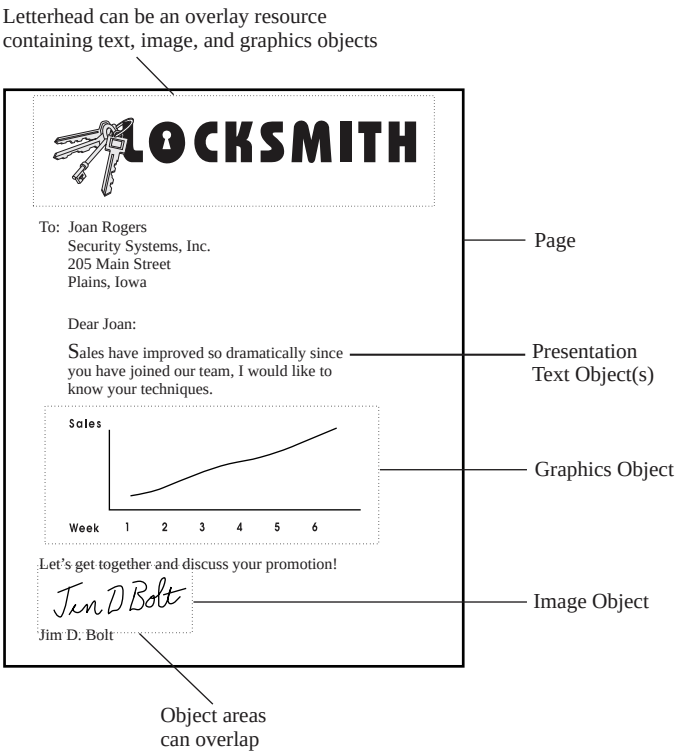
- *Presentation Text Object Content Architecture (PTOCA):* A data architecture for describing text objects that have been formatted for all-points-addressable presentations. Specifications of fonts, text color, and other visual attributes are included in the architecture definition.
- *Image Object Content Architecture (IOCA):* A data architecture for describing resolution-independent image objects captured from a number of different sources. Specifications of recording formats, data compression, color, and grayscale encoding are included in the architecture definition.
- *Graphics Object Content Architecture for Advanced Function Presentation (AFP GOCA):* A version of GOCA that is used in Advanced Function Presentation (AFP) environments. GOCA is a data architecture for describing vector graphics 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.
- *Bar Code Object Content Architecture (BCOCA):* A data architecture for describing bar code objects, using a number of different symbologies. Specification of the data to be encoded and the symbology attributes to be used are included in the architecture definition.
- *Font Object Content Architecture (FOCA):* A resource architecture for describing the structure and content of fonts referenced by presentation data objects in the document.
- *Color Management Object Content Architecture (CMOCA):* A resource architecture used to carry the color management information required to render presentation data.
- *Metadata Object Content Architecture (MOCA):* A resource architecture used to carry metadata in an AFP environment.

The MO:DCA and IPDS architectures also support data objects that are not defined by object content architectures. Examples of such objects are Tag Image File Format (TIFF), Encapsulated PostScript® (EPS), and Portable Document Format (PDF). Such objects can be carried in a MO:DCA envelope called an *object container*, or they can be referenced without being enveloped in MO:DCA structures.

In addition to object content architectures, the MO:DCA architecture defines envelope architectures for objects of common value in the presentation environment. Examples of these are *Form Definition* resource objects for managing the production of pages on the physical media, *overlay* resource objects that accommodate electronic storage of forms data, and *index* resource objects that support indexing and tagging of pages in a document.

Figure 3 on page 5 shows an example of an all-points-addressable page composed of multiple presentation objects.

Figure 3. Presentation Page. This is an example of a mixed-object page that can be composed in a device-independent MO:DCA format and can be printed on an IPDS printer.



Chapter 2. Introduction to MOCA

The Metadata Object Content Architecture (MOCA) defines an AFP object that carries metadata in an AFP environment. This object is called a Metadata Object (MO). The long range objective of MOCA is to improve the utility, manageability, and archive integrity of AFP documents.

The initial MOCA specification is defined to support the requirements of AFP Archive. This places limits on the number of MO types and the level of penetration into the MO:DCA component hierarchy where metadata may be associated. Only descriptive metadata, with no presentation or operational semantic, is defined.

MOCA may be extended, in the future, to include additional MO types, associations and methods, including the possibility of operational metadata that reaches the presentation device.

Chapter 3. Metadata Concepts

Metadata is descriptive information that is associated with and augments other data. Some examples of metadata for a print file are the Date, Author, Description, or Copyright information. The primary purpose of the Metadata Object Content Architecture is to provide a framework for carrying and referencing metadata in MO:DCA environments. There are many predefined metadata specifications. MOCA will accommodate selected industry defined metadata formats while also providing a placeholder for AFP specific metadata, to be defined by the AFPC to address targeted needs should they arise in the future.

A Metadata Object (MO) carries metadata in the following format:

- **XMP**™, which is the Adobe Extensible Metadata Platform.

In the MO:DCA environment, MOs will be carried within an object container (BOC/EOC).

Metadata, in general, has no intended presentation semantics.

Chapter 4. Metadata Object (MO)

An MO consists of a header followed by MO data.

MO Syntax

This is the syntax of an MO.

Data contained in fixed-length fields that are encoded as UTF-16BE is left-aligned. If the number of bytes used by the characters in these fields is smaller than the field length, the remaining bytes will be padded with “@” (X'0040').

Table 4. MO Syntax

Length in Bytes	Offset	Type	Name	Range	Meaning	M/O
4	0-3	UBIN	MOLength	X'00000032' - X'FFFFFFFF'	MO length, including this MOLength field	M
MO header starts here						
2	4-5	UBIN	HeaderLength	X'002E' - end of header	MO header length, including this HeaderLength field	M
6	6-11	UTF16	MOType	DES (X'0044 0045 0053')	Descriptive	M
8	12-19	UTF16	MOFormat	XMP (X'0058 004D 0050 0040')	Extensible Metadata Platform (XMP)	M
20	20-39	UTF16	MOCompression	NONE (X'004E 004F 004E 0045 0040 0040 0040 0040 0040 0040')	Uncompressed	M
				GZIP (X'0047 005A 0049 0050 0040 0040 0040 0040 0040 0040')	“Gzip” text compression	
				EXI (X'0045 0058 0049 0040 0040 0040 0040 0040 0040 0040')	Efficient XML Interchange (EXI) compression	
8	40-47			X'0000000000000000'	Reserved - should be set to zero	M
2	48-49	UBIN	MONameLength	X'0000' - X'00FA', even values only	Length, in bytes, of the MOName field that follows	M
0-250	50-end of name	UTF16	MOName	Any valid UTF-16BE characters (thus an even number of bytes)	A human-readable MO name in UTF-16BE	O
	End of name-end of header	UNDF			Reserved for future use; receivers should accept but ignore; generators should not specify	O
MO header ends here						
			MOData	Any	MO Data	O

M/O: Mandatory or Optional field

MO Semantics

MOLength The length of the complete MO, including the MOLength parameter. MOLength, in bytes, may be 50 (X'00000032') to X'FFFFFFFF'. If an invalid value is found in this field, the optional exception is EC-0100.

HeaderLength The length of the MO header, including the HeaderLength parameter. HeaderLength, in bytes, may be any value greater than or equal to 46 (X'002E'). If an invalid value is found in this field, the optional exception is EC-0200.

MOType One MOType is defined in the Metadata Object Content Architecture. The defined MOType is DES. See “MOType” on page 15. If an invalid or unsupported value is found in this field, the optional exception is EC-0220.

MOFormat One MOFormat is defined in the Metadata Object Content Architecture. The defined MOFormat is XMP. See “MOFormat” on page 15. If an invalid or unsupported value is found in this field, the optional exception is EC-0230.

MOCompression MOCompression is defined in the Metadata Object Content Architecture to indicate the type of compression applied to MO data. See “MOCompression” on page 15. If an invalid or unsupported value is found in this field, the optional exception is EC-0240.

MONameLength The length of the MOName field. MONameLength, in bytes, may be any even value from 0 (X'0000') to 250 (X'00FA'). If an invalid value is found in this field, the optional exception is EC-0250.

MOName A user-defined string containing an optional human-readable MO name. This field can contain up to 250 bytes; therefore, if the UTF-16BE string contains no surrogates, the MO name can contain up to 125 characters. If the environment containing the MO has a method of referencing an MO by name, this field is to be used as the name. If an invalid value is found in this field, the optional exception is EC-0210.

MOData MO data. The format of the metadata is determined by the value of the MOFormat parameter. If an invalid value is found in this field, the optional exception is EC-0300.

MO Exception Conditions

Both recognition and reporting of exception conditions is optional. Exception conditions have a format of EC-xxxx.

The exception conditions are as follows:

EC-0100 Invalid Length Value

The specified MOLength is invalid.

EC-0200 Invalid Field Value

The specified HeaderLength is invalid.

EC-0210 Invalid Field Value

The specified MOName is not valid UTF-16BE.

EC-0220 Invalid or Unsupported Field Value

The specified MOType is invalid or unsupported.

EC-0230 Invalid or Unsupported Field Value

The specified MOFormat is invalid or unsupported.

EC-0240 Invalid or Unsupported Field Value

The specified MOCompression is invalid or unsupported.

EC-0250 Invalid Field Value

The specified MONameLength is invalid.

EC-0300 Invalid MOData

The specified MOData does not meet the specification associated with the indicated MOFormat.

Chapter 5. MO Header Attributes

Attribute fields for MO type, format, and compression are carried in the MO header. Each of these fields is described below.

MOType

The following MO types are defined:
- DES

Each value for MOType is described in more detail below.

DES

MOType = DES (X'004400450053')

Description

MOType DES refers to descriptive metadata used to label, tag, or otherwise describe elements of a print file. Common descriptive metadata are attributes such as Title, Date, and Author. Descriptive metadata is distinct from the primary content of a document and does not affect the architected rendering of data.

MOFormat

The following MO formats are defined:
- XMP

Each value for MOFormat is described in more detail below.

XMP

MOFormat = XMP (X'0058004D00500040')

Description

MOFormat XMP refers to a metadata object using the Extensible Metadata Platform (XMP) data model, serialization, and core properties.

See the *XMP Specification* dated September 2005 for a complete specification of XMP.

MOCompression

The following MO compression formats are defined:
- NONE
- GZIP
- EXI

Each value for MOCompression is described in more detail below.

NONE

MOCompression = NONE (X'004E004F004E00450040004000400040004000400040')

Description

The MO data is uncompressed.

GZIP

MOCompression = GZIP (X'0047005A00490050004000400040004000400040')

Description

The MO data is compressed as text using GZIP.

See *RFC 1952 - GZIP file format specification version 4.3*.

EXI

MOCompression = EXI (X'00450058004900400040004000400040004000400040')

Description

The MO data is compressed as XML using Efficient XML Interchange.

See *Efficient XML Interchange (EXI) Format 1.0*.

Notices

The AFP Consortium or consortium member companies might 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.

The following statement does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: AFP Consortium 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 might not apply to you.

This publication 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. The AFP Consortium might make improvements and/or changes in the architecture described in this publication at any time without notice.

Any references in this publication to 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 architecture and use of those Web sites is at your own risk.

The AFP Consortium may use or distribute any information you supply in any way it believes appropriate without incurring any obligation to you.

This information contains examples of data and reports used in daily business operations. To illustrate them in a complete manner, some examples include the names of individuals, companies, brands, or products. These names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

Trademarks

These terms are trademarks or registered trademarks of Ricoh Co., Ltd., in the United States, other countries, or both:

ACMA
Advanced Function Presentation
AFP
AFPCC
AFP Color Consortium
AFP Color Management Architecture
Bar Code Object Content Architecture
BCOCA
CMOCA
Color Management Object Content Architecture
InfoPrint
Intelligent Printer Data Stream
IPDS
Mixed Object Document Content Architecture
MO:DCA
Ricoh

IBM is a trademark of the International Business Machines Corporation in the United States, other countries, or both.

PostScript and XMP are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States and/or other countries.

AFPC and AFP Consortium are trademarks of the AFP Consortium.

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

Glossary

This glossary contains terms that apply to MOCA and also a few terms that apply to other related presentation architectures.

If you do not find the term that you are looking for, please refer to the *IBM Dictionary of Computing* (document number ZC20-1699), the *InfoPrint Dictionary of Printing*, or the glossary in the *IPDS Reference* (document number AFPC-0001).

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.

A

Advanced Function Presentation (AFP). An open architecture for the management of presentable information that is developed by the AFP Consortium (AFPC). AFP comprises a number of data stream and data object architectures:

- Mixed Object Document Content Architecture (MO:DCA); formerly referred to as AFPDS
- Intelligent Printer Data Stream (IPDS) Architecture
- AFP Line Data Architecture
- Bar Code Object Content Architecture (BCOCA)
- Color Management Object Content Architecture (CMOCA)
- Font Object Content Architecture (FOCA)
- Graphics Object Content Architecture for AFP (AFP GOCA)
- Image Object Content Architecture (IOCA)
- Metadata Object Content Architecture (MOCA)
- Presentation Text Object Content Architecture (PTOCA)

AFP. See *Advanced Function Presentation*.

AFP Consortium (AFPC). A formal open standards body that develops and maintains AFP architecture. Information about the consortium can be found at `http://www.afpcinc.org`.

AFP data stream. A presentation data stream that is processed in AFP environments. The MO:DCA architecture defines the strategic AFP interchange data stream. The IPDS architecture defines the strategic AFP printer data stream.

AFPDS. A term formerly used to identify the composed-page MO:DCA-based data stream interchanged in AFP environments. See also *MO:DCA* and *AFP data stream*.

application. (1) The use to which an information system is put. (2) A collection of software components used to perform specific types of work on a computer.

architected. Identifies data that is defined and controlled by an architecture. Contrast with *unarchitected*.

B

bar code. An array of elements, such as bars, spaces, and two-dimensional modules that together represent data elements or characters in a particular symbology. The elements are arranged in a predetermined pattern following unambiguous rules defined by the symbology.

Bar Code Object Content Architecture (BCOCA). An architected collection of constructs used to interchange and present bar code data.

BCOCA. See *Bar Code Object Content Architecture*.

C

CMOCA. See *Color Management Object Content Architecture*.

Color Management Object Content Architecture (CMOCA). An architected collection of constructs used for the interchange and presentation of the color management information required to render a print file, document, group of pages or sheets, page, overlay, or data object with color fidelity.

compression algorithm. An algorithm used to compress data. Compression of data can decrease the volume of data.

construct. An architected set of data such as a structured field or a triplet.

controlling environment. The environment in which an object is embedded, for example, the IPDS and MO:DCA data streams.

D

data stream. A continuous stream of data that has a defined format. An example of a defined format is a structured field.

deprecated. An architected construct is marked as “deprecated” to indicate that it should no longer be used because it has been superseded by a newer construct.

Use or support of a deprecated construct is permitted but no longer recommended. Constructs are deprecated rather than immediately removed to provide backward compatibility.

device dependent. Dependent upon one or more device characteristics.

device independent. Not dependent upon device characteristics.

document. (1) A machine-readable collection of one or more objects that represents a composition, a work, or a collection of data. (2) A publication or other written material.

document component. An architected part of a document data stream. Examples of document components are documents, pages, page groups, indexes, resource groups, objects, and process elements.

document content architecture. A family of architectures that define the syntax and semantics of the document component. See also *document component* and *structured field*.

document element. A self-identifying, variable-length, bounded record, that can have a content portion that provides control information, data, or both. An application or device does not have to understand control information or data to parse a data stream when all the records in the data stream are document elements. See also *structured field*.

E

Efficient XML Interchange (EXI). A format that allows XML documents to be encoded as binary data, rather than as plain text.

exception. An invalid or unsupported data-stream construct.

exception action. Action taken when an exception is detected.

exception condition. The condition that exists when a product finds an invalid or unsupported construct.

exchange. The predictable interpretation of shared information by a family of system processes in an environment where the characteristics of each process must be known to all other processes. Contrast with *interchange*.

EXI. See *Efficient XML Interchange*.

Extensible Markup Language (XML). A set of rules for encoding documents in a format that is both human-readable and machine-readable.

Extensible Metadata Platform (XMP). An ISO standard, originally created by Adobe Systems Incorporated, for the creation, processing, and interchange of standardized and custom metadata for all kinds of resources.

F

FOCA. See *Font Object Content Architecture*.

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

G

GOCA. See *Graphics Object Content Architecture*.

graphics data. Data containing lines, arcs, markers, and other constructs that describe a picture.

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

GZIP. A widely-used,free software compression algorithm.

H

hexadecimal. A number system with a base of sixteen. The decimal digits 0 through 9 and characters A through F are used to represent hexadecimal digits. The hexadecimal digits A through F correspond to the decimal numbers 10 through 15, respectively. An example of a hexadecimal number is X'1B', that is equal to the decimal number 27.

host. In the IPDS architecture, a computer that drives a printer.

I

image. An electronic representation of a picture produced by means of sensing light, sound, electron radiation, or other emanations coming from the picture or reflected by the picture. An image can also be generated directly by software without reference to an existing picture.

Image Object Content Architecture (IOCA). An architected collection of constructs used to interchange and present images.

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.

interchange. The predictable interpretation of shared information in an environment where the characteristics of each process need not be known to all other processes. Contrast with *exchange*.

International Organization for Standardization (ISO). An organization of national standards bodies from various countries established to promote development of standards to facilitate international exchange of goods and services, and develop cooperation in intellectual, scientific, technological, and economic activity.

interoperability. The capability to communicate, execute programs, or transfer data among various functional units in a way that requires the user to have little or no knowledge of the unique characteristics of those units.

IOCA. See *Image Object Content Architecture*.

IPDS. See *Intelligent Printer Data Stream*.

ISO. See *International Organization for Standardization*.

M

M/O. A table heading for architecture syntax. The entries under this heading indicate whether the construct is mandatory (M) or optional (O).

meaning. A table heading for architecture syntax. The entries under this heading convey the meaning or purpose of a construct. A meaning entry can be a long name, a description, or a brief statement of function.

metadata. Descriptive information that is associated with and augments other data.

metadata object. In AFP, the resource object that carries metadata.

Metadata Object Content Architecture (MOCA). A resource object architecture to carry metadata that serves to provide context or additional information about an AFP object or other AFP data.

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

MO:DCA. See *Mixed Object Document Content Architecture*.

MOCA. See *Metadata Object Content Architecture*.

N

name. A table heading for architecture syntax. The entries under this heading are short names that give a general indication of the contents of the construct.

O

object. (1) 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, that can be used to reference the object. Examples of objects are metadata, presentation text, font, graphics, and image objects. (2) Something that a user works with to perform a task.

offset. A table heading for architecture syntax. The entries under this heading indicate the numeric displacement into a construct. The offset is measured in bytes and starts with byte zero. Individual bits can be expressed as displacements within bytes.

P

parameter. (1) A variable that is given a constant value for a specified application. (2) A variable used in conjunction with a command to affect its result.

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

PTOCA. See *Presentation Text Object Content Architecture*.

R

range. A table heading for architecture syntax. The entries under this heading give numeric ranges applicable to a construct. The ranges can be expressed in binary, decimal, or hexadecimal. The range can consist of a single value.

reserved. Having no assigned meaning and put aside for future use. The content of reserved fields is not used by receivers, and should be set by generators to a specified value, if given, or to binary zeros. A reserved field or value can be assigned a meaning by an architecture at any time.

retired. Set aside for a particular purpose, and not available for any other purpose. Retired fields and values are specified for compatibility with existing products and identify one of the following:

- Fields or values that have been used by a product in a manner not compliant with the architected definition
- Fields or values that have been removed from an architecture

S

semantics. The meaning of the parameters of a construct. See also *syntax*.

structured field. A self-identifying, variable-length, bounded record, that can have a content portion that

provides control information, data, or both. See also *document element*.

surrogate pair. A sequence of two Unicode code points that allow for the encoding of as many as 1 million additional characters without any use of escape codes.

syntax. The rules governing the structure of a construct. See also *semantics*.

T

triplet. A three-part self-defining variable-length parameter consisting of a length byte, an identifier byte, and parameter-value bytes.

U

UBIN. A data type for architecture syntax, indicating one or more bytes to be interpreted as an unsigned binary number.

unarchitected. Identifies data that is neither defined nor controlled by an architecture. Contrast with *architected*.

UNDF. A data type for architecture syntax, indicating one or more bytes that are undefined by the architecture.

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+nnnn. Unicode provides for three encoding forms (UTF-8, UTF-16, and UTF-32), described as follows:

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

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).

X

XML. See *Extensible Markup Language*.

XMP. See *Extensible Metadata Platform*.

Index

A

E

G

M

T

X

Advanced Function Presentation Consortium

Metadata Object Content Architecture Reference

AFPC-0013-01

AFPConsortium

Advanced Function Presentation