[Note that this file is a concatenation of more than one RFC.]

Internet Engineering Task Force (IETF)
Request for Comments: 8010
Obsoletes: 2910, 3382
Category: Standards Track
ISSN: 2070-1721

M. Sweet
Apple Inc.
I. McDonald
High North, Inc.

January 2017

Internet Printing Protocol/1.1: Encoding and Transport

Abstract

The Internet Printing Protocol (IPP) is an application-level protocol for distributed printing using Internet tools and technologies. This document defines the rules for encoding IPP operations, attributes, and values into the Internet MIME media type called "application/ipp". It also defines the rules for transporting a message body whose Content-Type is "application/ipp" over HTTP and/or HTTPS. The IPP data model and operation semantics are described in "Internet Printing Protocol/1.1: Model and Semantics" (RFC 8011).

This document obsoletes RFCs 2910 and 3382.

Status of This Memo

This is an Internet Standards Track document.

This document is a product of the Internet Engineering Task Force (IETF). It represents the consensus of the IETF community. It has received public review and has been approved for publication by the Internet Engineering Steering Group (IESG). Further information on Internet Standards is available in Section 2 of RFC 7841.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at http://www.rfc-editor.org/info/rfc8010.

Table of Contents

1. Introduction

This document contains the rules for encoding IPP operations and describes two layers: the transport layer and the operation layer.

The transport layer consists of an HTTP request and response. All IPP implementations support HTTP/1.1, the relevant parts of which are described in the following RFCs:

- Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing [RFC7230]
- Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content [RFC7231]
- Hypertext Transfer Protocol (HTTP/1.1): Conditional Requests [RFC7232]
- Hypertext Transfer Protocol (HTTP/1.1): Caching [RFC7234]
- Hypertext Transfer Protocol (HTTP/1.1): Authentication [RFC7235]
- The 'Basic' HTTP Authentication Scheme [RFC7617]
- HTTP Digest Access Authentication [RFC7616]

IPP implementations can support HTTP/2, which is described in the following RFCs:

- Hypertext Transfer Protocol Version 2 (HTTP/2) [RFC7540]
- HPACK - Header Compression for HTTP/2 [RFC7541]

This document specifies the HTTP headers that an IPP implementation supports.

The operation layer consists of a message body in an HTTP request or response. The "Internet Printing Protocol/1.1: Model and Semantics" document [RFC8011] and subsequent extensions (collectively known as the IPP Model) define the semantics of such a message body and the supported values. This document specifies the encoding of an IPP request and response message.

2. Conventions Used in This Document

2.1. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

2.2. Printing Terminology

Client: Initiator of outgoing IPP session requests and sender of outgoing IPP operation requests (Hypertext Transfer Protocol -- HTTP/1.1 [RFC7230] User Agent).

Document: An object created and managed by a Printer that contains description, processing, and status information. A Document object may have attached data and is bound to a single Job.

'ipp' URI: An IPP URI as defined in [RFC3510].

'ipps' URI: An IPPS URI as defined in [RFC7472].

Job: An object created and managed by a Printer that contains description, processing, and status information. The Job also contains zero or more Document objects.

Logical Device: A print server, software service, or gateway that processes Jobs and either forwards or stores the processed Job or uses one or more Physical Devices to render output.

Model: The semantics of operations, attributes, values, and status-codes used in the Internet Printing Protocol as defined in the Internet Printing Protocol/1.1: Model and Semantics document [RFC8011] and subsequent extensions.

Output Device: A single Logical or Physical Device.

Physical Device: A hardware implementation of an endpoint device, e.g., a marking engine, a fax modem, etc.

Printer: Listener for incoming IPP session requests and receiver of incoming IPP operation requests (Hypertext Transfer Protocol -- HTTP/1.1 [RFC7230] Server) that represents one or more Physical Devices or a Logical Device.

2.3. Abbreviations

ABNF: Augmented Backus-Naur Form [RFC5234]

ASCII: American Standard Code for Information Interchange [RFC20]

HTTP: Hypertext Transfer Protocol [RFC7230]

HTTPS: HTTP over TLS [RFC2818]

IANA: Internet Assigned Numbers Authority

IEEE: Institute of Electrical and Electronics Engineers

IESG: Internet Engineering Steering Group

IPP: Internet Printing Protocol (this document and [PWG5100.12])

ISTO: IEEE Industry Standards and Technology Organization

LPD: Line Printer Daemon Protocol [RFC1179]

PWG: IEEE-ISTO Printer Working Group

RFC: Request for Comments

TCP: Transmission Control Protocol [RFC793]

TLS: Transport Layer Security [RFC5246]

URI: Uniform Resource Identifier [RFC3986]

URL: Uniform Resource Locator [RFC3986]

UTF-8: Unicode Transformation Format - 8-bit [RFC3629]

3. Encoding of the Operation Layer

The operation layer is the message body part of the HTTP request or response and it MUST contain a single IPP operation request or IPP operation response. Each request or response consists of a sequence of values and attribute groups. Attribute groups consist of a sequence of attributes each of which is a name and value. Names and values are ultimately sequences of octets.

The encoding consists of octets as the most primitive type. There are several types built from octets, but three important types are integers, character strings, and octet strings, on which most other

data types are built. Every character string in this encoding MUST be a sequence of characters where the characters are associated with some charset [RFC2978] and some natural language. A character string MUST be in "reading order" with the first character in the value (according to reading order) being the first character in the encoding. A character string whose associated charset is US-ASCII and whose associated natural language is US English is henceforth called a US-ASCII-STRING. A character string whose associated charset and natural language are specified in a request or response as described in the Model is henceforth called a LOCALIZED-STRING. An octet string MUST be in "Model order" with the first octet in the value (according to the Model order) being the first octet in the encoding. Every integer in this encoding MUST be encoded as a signed integer using two's-complement binary encoding with big-endian format (also known as "network order" and "most significant byte first"). The number of octets for an integer MUST be 1, 2, or 4, depending on usage in the protocol. A one-octet integer, henceforth called a SIGNED-BYTE, is used for the version-number and tag fields. A two-byte integer, henceforth called a SIGNED-SHORT, is used for the operation-id, status-code, and length fields. A four-byte integer, henceforth called a SIGNED-INTEGER, is used for value fields and the request-id.

The following two sections present the encoding of the operation layer in two ways:

- informally through pictures and description
- formally through Augmented Backus-Naur Form (ABNF), as specified by RFC 5234 [RFC5234]

An operation request or response MUST use the encoding described in these two sections.

3.1. Picture of the Encoding

3.1.1. Request and Response

An operation request or response is encoded as follows:

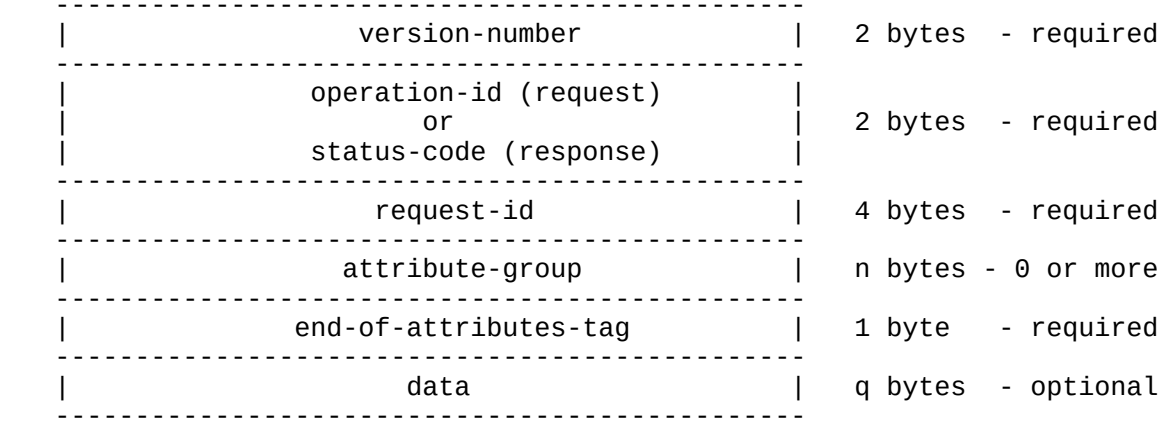


Figure 1: IPP Message Format

The first three fields in the above diagram contain the value of attributes described in Section 4.1.1 of the Model and Semantics document [RFC8011].

The fourth field is the "attribute-group" field, and it occurs 0 or more times. Each "attribute-group" field represents a single group of attributes, such as an Operation Attributes group or a Job Attributes group (see the Model). The Model specifies the required attribute groups and their order for each operation request and response.

The "end-of-attributes-tag" field is always present, even when the "data" is not present. The Model specifies whether the "data" field is present for each operation request and response.

3.1.2. Attribute Group

Each "attribute-group" field is encoded as follows:

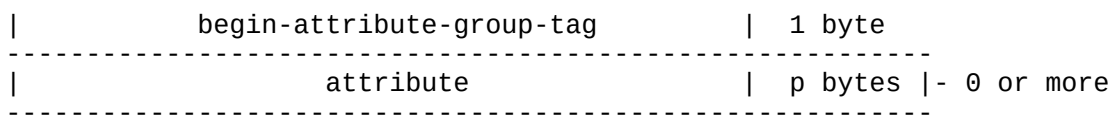


Figure 2: Attribute Group Encoding

An "attribute-group" field contains zero or more "attribute" fields.

Note that the values of the "begin-attribute-group-tag" field and the "end-of-attributes-tag" field are called "delimiter-tags".

3.1.3. Attribute

An "attribute" field is encoded as follows:

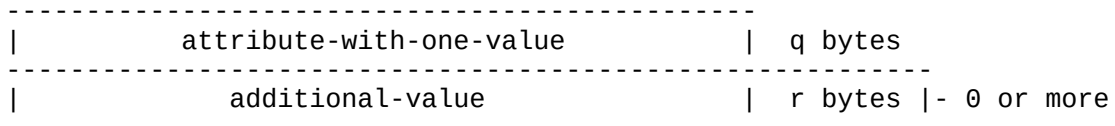


Figure 3: Attribute Encoding

When an attribute is single valued (e.g., "copies" with a value of 10) or multi-valued with one value (e.g., "sides-supported" with just the value 'one-sided'), it is encoded with just an "attribute-with-one-value" field. When an attribute is multi-valued with n values (e.g., "sides-supported" with the values 'one-sided' and 'two-sided-long-edge'), it is encoded with an "attribute-with-one-value" field followed by n-1 "additional-value" fields.

3.1.4. Attribute-with-one-value

Each "attribute-with-one-value" field is encoded as follows:

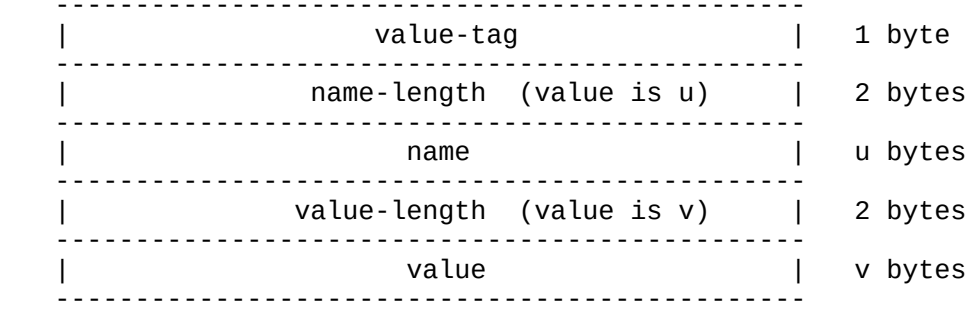


Figure 4: Single Value Attribute Encoding

An "attribute-with-one-value" field is encoded with five subfields:

- The "value-tag" field specifies the attribute syntax, e.g., 0x44 for the attribute syntax ’keyword’.
- The "name-length" field specifies the length of the "name" field in bytes, e.g., u in the above diagram or 15 for the name "sides-supported".
- The "name" field contains the textual name of the attribute, e.g., "sides-supported".
- The "value-length" field specifies the length of the "value" field in bytes, e.g., v in the above diagram or 9 for the (keyword) value ’one-sided’.
- The "value" field contains the value of the attribute, e.g., the textual value ’one-sided’.

3.1.5. Additional-value

Each "additional-value" field is encoded as follows:

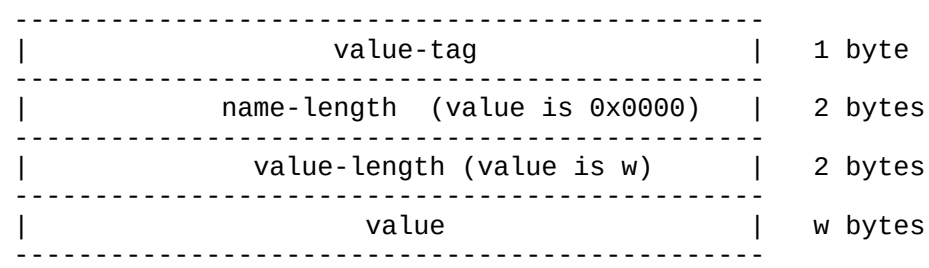


Figure 5: Additional Attribute Value Encoding

An "additional-value" is encoded with four subfields:

- The "value-tag" field specifies the attribute syntax, e.g., 0x44 for the attribute syntax ’keyword’.

- The "name-length" field has the value of 0 in order to signify that it is an "additional-value". The value of the "name-length" field distinguishes an "additional-value" field ("name-length" is 0) from an "attribute-with-one-value" field ("name-length" is not 0).

- The "value-length" field specifies the length of the "value" field in bytes, e.g., w in the above diagram or 19 for the (keyword) value ’two-sided-long-edge’.

- The "value" field contains the value of the attribute, e.g., the textual value ’two-sided-long-edge’.

3.1.6. Collection Attribute

Collection attributes create a named group containing related "member" attributes. The "attribute-with-one-value" field for a collection attribute is encoded as follows:

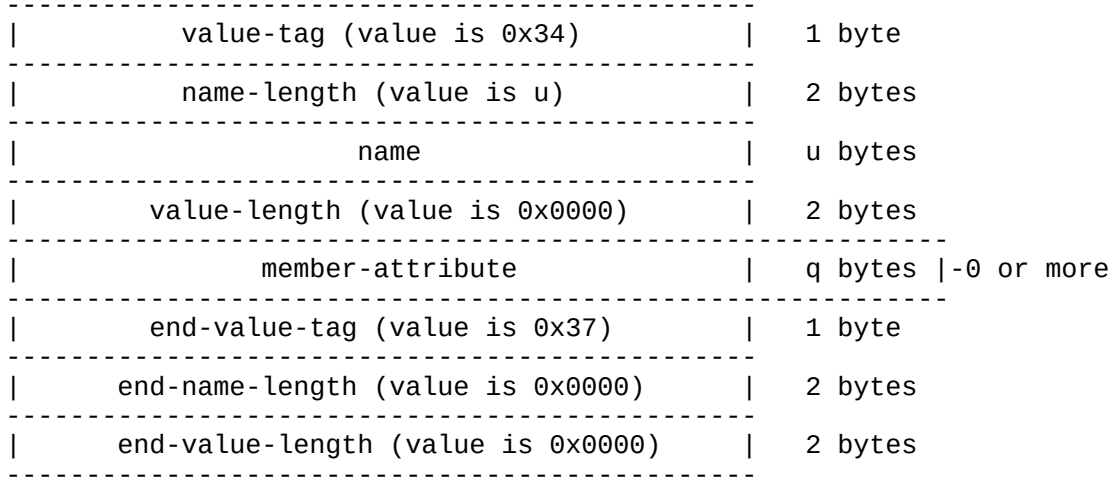


Figure 6: Collection Attribute Encoding

Collection attribute is encoded with eight subfields:

- The "value-tag" field specifies the start attribute syntax: 0x34 for the attribute syntax 'begCollection'.
- The "name-length" field specifies the length of the "name" field in bytes, e.g., u in the above diagram or 9 for the name "media-col". Additional collection attribute values use a name length of 0x0000.
- The "name" field contains the textual name of the attribute, e.g., "media-col".
- The "value-length" field specifies a length of 0x0000.
- The "member-attribute" field contains member attributes encoded as defined in Section 3.1.7.
- The "end-value-tag" field specifies the end attribute syntax: 0x37 for the attribute syntax 'endCollection'.
- The "end-name-length" field specifies a length of 0x0000.

o The "end-value-length" field specifies a length of 0x0000.

3.1.7. Member Attributes

Each "member-attribute" field is encoded as follows:

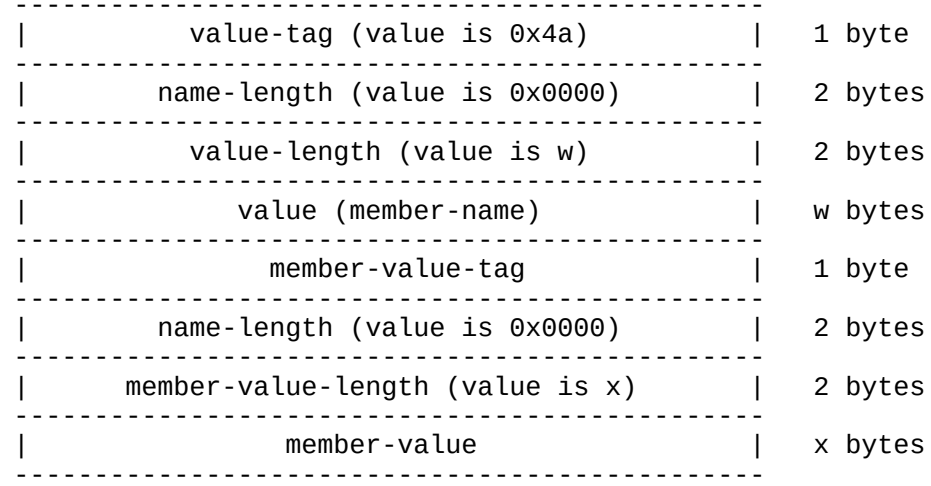


Figure 7: Member Attribute Encoding

A "member-attribute" is encoded with eight subfields:

o The "value-tag" field specifies 0x4a for the attribute syntax 'memberAttrName'.

o The "name-length" field has the value of 0 in order to signify that it is a "member-attribute" contained in the collection.

o The "value-length" field specifies the length of the "value" field in bytes, e.g., w in the above diagram or 10 for the member attribute name 'media-type'. Additional member attribute values are specified using a value length of 0.

o The "value" field contains the name of the member attribute, e.g., the textual value 'media-type'.

o The "member-value-tag" field specifies the attribute syntax for the member attribute, e.g., 0x44 for the attribute syntax 'keyword'.

- The second "name-length" field has the value of 0 in order to signify that it is a "member-attribute" contained in the collection.
- The "member-value-length" field specifies the length of the member attribute value, e.g., x in the above diagram or 10 for the value 'stationery'.
- The "member-value" field contains the value of the attribute, e.g., the textual value 'stationery'.

3.1.8. Alternative Picture of the Encoding of a Request or a Response

From the standpoint of a parser that performs an action based on a "tag" value, the encoding consists of:

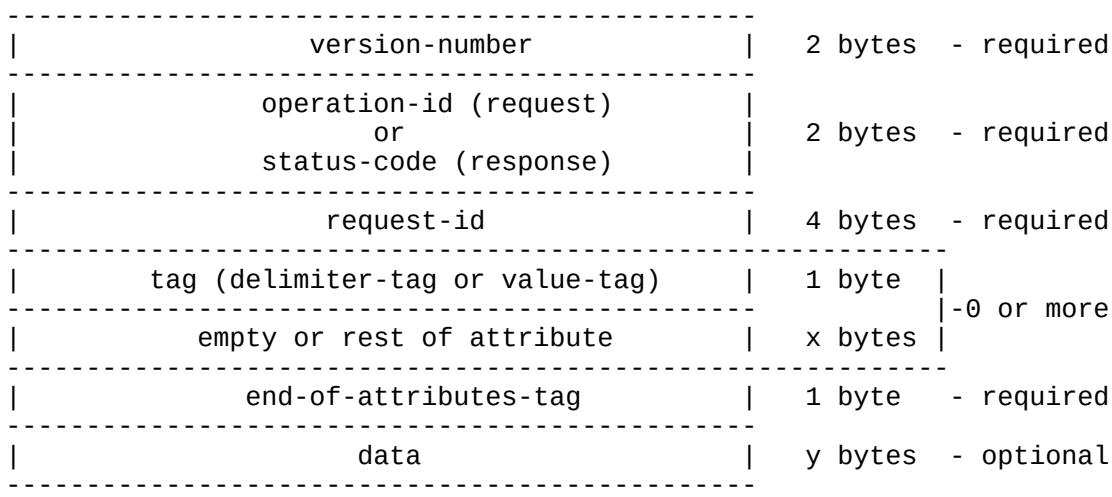


Figure 8: Encoding Based on Value Tags

The following shows what fields the parser would expect after each type of "tag":

- "begin-attribute-group-tag": expect zero or more "attribute" fields
- "value-tag": expect the remainder of an "attribute-with-one-value" or an "additional-value"
- "end-of-attributes-tag": expect that "attribute" fields are complete and there is optional "data"

3.2. Syntax of Encoding

The ABNF [RFC5234] syntax for an IPP message is shown in Figure 9.

```
ipp-message  = ipp-request / ipp-response
ipp-request  = version-number operation-id request-id
               *attribute-group end-of-attributes-tag data
ipp-response = version-number status-code request-id
               *attribute-group end-of-attributes-tag  data

version-number       = major-version-number minor-version-number
major-version-number = SIGNED-BYTE
minor-version-number = SIGNED-BYTE

operation-id = SIGNED-SHORT     ; mapping from model
status-code  = SIGNED-SHORT     ; mapping from model
request-id   = SIGNED-INTEGER   ; whose value is > 0

attribute-group          = begin-attribute-group-tag *attribute
attribute                = attribute-with-one-value *additional-value
attribute-with-one-value = value-tag name-length name
                           value-length value
additional-value         = value-tag zero-name-length
                           value-length value

name-length  = SIGNED-SHORT     ; number of octets of 'name'
name         = LALPHA *( LALPHA / DIGIT / "-" / "_" / "." )
value-length = SIGNED-SHORT     ; number of octets of 'value'
value        = OCTET-STRING
data         = OCTET-STRING

zero-name-length          = %x00.00           ; name-length of 0
value-tag                 = %x10-ff           ; see Section 3.5.2
begin-attribute-group-tag = %x00-02 / %x04-0f ; see Section 3.5.1
end-of-attributes-tag     = %x03              ; tag of 3
                                              ; see Section 3.5.1

SIGNED-BYTE    = BYTE
SIGNED-SHORT   = 2BYTE
SIGNED-INTEGER = 4BYTE
DIGIT          = %x30-39        ; "0" to "9"
LALPHA         = %x61-7A        ; "a" to "z"
BYTE           = %x00-ff
OCTET-STRING   = *BYTE
```

Figure 9: ABNF of IPP Message Format

Figure 10 defines additional terms that are referenced in this document and provides an alternate grouping of the delimiter tags.

```
delimiter-tag = begin-attribute-group-tag /   ; see Section 3.5.1
          end-of-attributes-tag
begin-attribute-group-tag = %x00 / operation-attributes-tag /
   job-attributes-tag / printer-attributes-tag /
   unsupported-attributes-tag / future-group-tags
operation-attributes-tag   = %x01             ; tag of 1
job-attributes-tag         = %x02             ; tag of 2
end-of-attributes-tag      = %x03             ; tag of 3
printer-attributes-tag     = %x04             ; tag of 4
unsupported-attributes-tag = %x05             ; tag of 5
future-group-tags          = %x06-0f          ; future extensions
```

Figure 10: ABNF for Attribute Group Tags

3.3. Attribute-group

Each "attribute-group" field MUST be encoded with the "begin-attribute-group-tag" field followed by zero or more "attribute" sub-fields.

Table 1 maps the Model group name to value of the "begin-attribute-group-tag" field:

Model Document Group	"begin-attribute-group-tag" field values
Operation Attributes	"operations-attributes-tag"
Job Template Attributes	"job-attributes-tag"
Job Object Attributes	"job-attributes-tag"
Unsupported Attributes	"unsupported-attributes-tag"
Requested Attributes	(Get-Job-Attributes) "job-attributes-tag"
Requested Attributes	(Get-Printer-Attributes)"printer-attributes-tag"
Document Content	in a special position at the end of the message as described in Section 3.1.1.

Table 1: Group Values

For each operation request and response, the Model prescribes the required and optional attribute groups, along with their order. Within each attribute group, the Model prescribes the required and optional attributes, along with their order.

When the Model requires an attribute group in a request or response and the attribute group contains zero attributes, a request or response SHOULD encode the attribute group with the "begin-attribute-group-tag" field followed by zero "attribute" fields. For example, if the Client requests a single unsupported attribute with the Get-Printer-Attributes operation, the Printer MUST return no "attribute" fields, and it SHOULD return a "begin-attribute-group-tag" field for the Printer Attributes group. The Unsupported Attributes group is not such an example. According to the Model, the Unsupported Attributes group SHOULD be present only if the Unsupported Attributes group contains at least one attribute.

A receiver of a request MUST be able to process the following as equivalent empty attribute groups:

a. A "begin-attribute-group-tag" field with zero following "attribute" fields.

b. A missing, but expected, "begin-attribute-group-tag" field.

When the Model requires a sequence of an unknown number of attribute groups, each of the same type, the encoding MUST contain one "begin-attribute-group-tag" field for each attribute group, even when an "attribute-group" field contains zero "attribute" sub-fields. For example, the Get-Jobs operation may return zero attributes for some Jobs and not others. The "begin-attribute-group-tag" field followed by zero "attribute" fields tells the recipient that there is a Job in queue for which no information is available except that it is in the queue.

3.4. Required Parameters

Some operation elements are called parameters in the Model. They MUST be encoded in a special position and they MUST NOT appear as operation attributes. These parameters are described in the subsections below.

3.4.1. "version-number"

The "version-number" field consists of a major and minor version-number, each of which is represented by a SIGNED-BYTE. The major version-number is the first byte of the encoding and the minor version-number is the second byte of the encoding. The protocol described in [RFC8011] has a major version-number of 1 (0x01) and a minor version-number of 1 (0x01). The ABNF for these two bytes is %x01.01.

Note: See Section 9 for more information on the "version-number" field and IPP version numbers.

3.4.2. "operation-id"

The "operation-id" field contains an operation-id value as defined in the Model. The value is encoded as a SIGNED-SHORT and is located in the third and fourth bytes of the encoding of an operation request.

3.4.3. "status-code"

The "status-code" field contains a status-code value as defined in the Model. The value is encoded as a SIGNED-SHORT and is located in the third and fourth bytes of the encoding of an operation response.

If an IPP status-code is returned, then the HTTP status-code MUST be 200 (OK). With any other HTTP status-code value, the HTTP response MUST NOT contain an IPP message body, and thus no IPP status-code is returned.

3.4.4. "request-id"

The "request-id" field contains the request-id value as defined in the Model. The value is encoded as a SIGNED-INTEGER and is located in the fifth through eighth bytes of the encoding.

3.5. Tags

There are two kinds of tags:

- delimiter tags: delimit major sections of the protocol, namely attribute groups and data
- value tags: specify the type of each attribute value

Tags are part of the IANA IPP registry [IANA-IPP]

3.5.1. "delimiter-tag" Values

Table 2 specifies the values for the delimiter tags defined in this document. These tags are registered, along with tags defined in other documents, in the "Attribute Group Tags" registry.

Tag Value (Hex)	Meaning
0x00	Reserved
0x01	"operation-attributes-tag"
0x02	"job-attributes-tag"
0x03	"end-of-attributes-tag"
0x04	"printer-attributes-tag"
0x05	"unsupported-attributes-tag"

Table 2: "delimiter-tag" Values

When a "begin-attribute-group-tag" field occurs in the protocol, it means that zero or more following attributes up to the next group tag are attributes belonging to the attribute group specified by the value of the "begin-attribute-group-tag". For example, if the value of "begin-attribute-group-tag" is 0x01, the following attributes are members of the Operations Attributes group.

The "end-of-attributes-tag" (value 0x03) MUST occur exactly once in an operation and MUST be the last "delimiter-tag". If the operation has a document-data group, the Document data in that group follows the "end-of-attributes-tag".

The order and presence of "attribute-group" fields (whose beginning is marked by the "begin-attribute-group-tag" subfield) for each operation request and each operation response MUST be that defined in the Model.

A Printer MUST treat a "delimiter-tag" (values from 0x00 through 0x0f) differently from a "value-tag" (values from 0x10 through 0xff) so that the Printer knows there is an entire attribute group as opposed to a single value.

3.5.2. "value-tag" Values

The remaining tables show values for the "value-tag" field, which is the first octet of an attribute. The "value-tag" field specifies the type of the value of the attribute.

Table 3 specifies the "out-of-band" values for the "value-tag" field defined in this document. These tags are registered, along with tags defined in other documents, in the "Out-of-Band Attribute Value Tags" registry.

Tag Value (Hex)	Meaning
0x10	unsupported
0x12	unknown
0x13	no-value

Table 3: Out-of-Band Values

Table 4 specifies the integer values defined in this document for the "value-tag" field; they are registered in the "Attribute Syntaxes" registry.

Tag Value (Hex)	Meaning
0x20	Unassigned integer data type (see IANA IPP registry)
0x21	integer
0x22	boolean
0x23	enum
0x24-0x2f	Unassigned integer data types (see IANA IPP registry)

Table 4: Integer Tags

Table 5 specifies the octetString values defined in this document for the "value-tag" field; they are registered in the "Attribute Syntaxes" registry.

Tag Value (Hex)	Meaning
0x30	octetString with an unspecified format
0x31	dateTime
0x32	resolution
0x33	rangeOfInteger
0x34	begCollection
0x35	textWithLanguage
0x36	nameWithLanguage
0x37	endCollection
0x38-0x3f	Unassigned octetString data types (see IANA IPP registry)

Table 5: octetString Tags

Table 6 specifies the character-string values defined in this document for the "value-tag" field; they are registered in the "Attribute Syntaxes" registry.

Tag Value (Hex)	Meaning
0x40	Unassigned character-string data type (see IANA IPP registry)
0x41	textWithoutLanguage
0x42	nameWithoutLanguage
0x43	Unassigned character-string data type (see IANA IPP registry)
0x44	keyword
0x45	uri
0x46	uriScheme
0x47	charset
0x48	naturalLanguage
0x49	mimeMediaType
0x4a	memberAttrName
0x4b-0x5f	Unassigned character-string data types (see IANA IPP registry)

Table 6: String Tags

Note: An attribute value always has a type, which is explicitly specified by its tag; one such tag value is "nameWithoutLanguage". An attribute’s name has an implicit type, which is keyword.

The values 0x60-0xff are reserved for future type definitions in Standards Track documents.

The tag 0x7f is reserved for extending types beyond the 255 values available with a single byte. A tag value of 0x7f MUST signify that the first four bytes of the value field are interpreted as the tag value. Note this future extension doesn’t affect parsers that are unaware of this special tag. The tag is like any other unknown tag, and the value length specifies the length of a value, which contains a value that the parser treats atomically. Values from 0x00000000 to 0x3fffffff are reserved for definition in future Standards Track documents. The values 0x40000000 to 0x7fffffff are reserved for vendor extensions.

3.6. "name-length"

The "name-length" field consists of a SIGNED-SHORT and specifies the number of octets in the immediately following "name" field. The value of this field excludes the two bytes of the "name-length" field. For example, if the "name" field contains 'sides', the value of this field is 5.

If a "name-length" field has a value of zero, the following "name" field is empty and the following value is treated as an additional value for the attribute encoded in the nearest preceding "attribute-with-one-value" field. Within an attribute group, if two or more attributes have the same name, the attribute group is malformed (see [RFC8011]). The zero-length name is the only mechanism for multi-valued attributes.

3.7. (Attribute) "name"

The "name" field contains the name of an attribute. The Model specifies such names.

3.8. "value-length"

The "value-length" field consists of a SIGNED-SHORT, which specifies the number of octets in the immediately following "value" field. The value of this field excludes the two bytes of the "value-length" field. For example, if the "value" field contains the keyword (string) value 'one-sided', the value of this field is 9.

For any of the types represented by binary signed integers, the sender MUST encode the value in exactly four octets.

For any of the types represented by binary signed bytes, e.g., the boolean type, the sender MUST encode the value in exactly one octet.

For any of the types represented by character strings, the sender MUST encode the value with all the characters of the string and without any padding characters.

For "out-of-band" values for the "value-tag" field defined in this document, such as 'unsupported', the "value-length" MUST be 0 and the "value" empty; the "value" has no meaning when the "value-tag" has one of these "out-of-band" values. For future "out-of-band" "value-tag" fields, the same rule holds unless the definition explicitly states that the "value-length" MAY be non-zero and the "value" non-empty

3.9. (Attribute) "value"

The syntax types (specified by the "value-tag" field) and most of the details of the representation of attribute values are defined in the Model. Table 7 augments the information in the Model and defines the syntax types from the Model in terms of the five basic types defined in Section 3. The five types are US-ASCII-STRING, LOCALIZED-STRING, SIGNED-INTEGER, SIGNED-SHORT, SIGNED-BYTE, and OCTET-STRING.

Syntax of Attribute Value	Encoding
textWithoutLanguage, nameWithoutLanguage	LOCALIZED-STRING
textWithLanguage	OCTET-STRING consisting of four fields: a SIGNED-SHORT, which is the number of octets in the following field; a value of type natural-language; a SIGNED-SHORT, which is the number of octets in the following field; and a value of type textWithoutLanguage. The length of a textWithLanguage value MUST be 4 + the value of field a + the value of field c.
nameWithLanguage	OCTET-STRING consisting of four fields: a SIGNED-SHORT, which is the number of octets in the following field; a value of type natural-language; a SIGNED-SHORT, which is the number of octets in the following field; and a value of type nameWithoutLanguage. The length of a nameWithLanguage value MUST be 4 + the value of field a + the value of field c.
charset, naturalLanguage, mimeMediaType, keyword, uri, and uriScheme	US-ASCII-STRING
boolean	SIGNED-BYTE where 0x00 is ’false’ and 0x01 is ’true’
integer and enum	a SIGNED-INTEGER

dateTime	OCTET-STRING consisting of eleven octets whose contents are defined by "DateAndTime" in RFC 2579 [RFC2579]
resolution	OCTET-STRING consisting of nine octets of two SIGNED-INTEGERs followed by a SIGNED-BYTE. The first SIGNED-INTEGER contains the value of cross-feed direction resolution. The second SIGNED-INTEGER contains the value of feed direction resolution. The SIGNED-BYTE contains the units value.
rangeOfInteger	Eight octets consisting of two SIGNED-INTEGERs. The first SIGNED-INTEGER contains the lower bound and the second SIGNED-INTEGER contains the upper bound.
1setOf X	Encoding according to the rules for an attribute with more than one value. Each value X is encoded according to the rules for encoding its type.
octetString	OCTET-STRING
collection	Encoding as defined in Section 3.1.6.

Table 7: Attribute Value Encoding

The attribute syntax type of the value determines its encoding and the value of its "value-tag".

3.10. Data

The "data" field MUST include any data required by the operation.

4. Encoding of Transport Layer

HTTP/1.1 [RFC7230] is the REQUIRED transport layer for this protocol. HTTP/2 [RFC7540] is an OPTIONAL transport layer for this protocol.

The operation layer has been designed with the assumption that the transport layer contains the following information:

- the target URI for the operation; and
- the total length of the data in the operation layer, either as a single length or as a sequence of chunks each with a length.

Printer implementations MUST support HTTP over the IANA-assigned well-known port 631 (the IPP default port), although a Printer implementation can support HTTP over some other port as well.

Each HTTP operation MUST use the POST method where the request-target is the object target of the operation and where the "Content-Type" of the message body in each request and response MUST be "application/ipp". The message body MUST contain the operation layer and MUST have the syntax described in Section 3.2, "Syntax of Encoding". A Client implementation MUST adhere to the rules for a Client described for HTTP [RFC7230]. A Printer (server) implementation MUST adhere to the rules for an origin server described for HTTP [RFC7230].

An IPP server sends a response for each request that it receives. If an IPP server detects an error, it MAY send a response before it has read the entire request. If the HTTP layer of the IPP server completes processing the HTTP headers successfully, it MAY send an intermediate response, such as "100 Continue", with no IPP data before sending the IPP response. A Client MUST expect such a variety of responses from an IPP server. For further information on HTTP, consult the HTTP documents [RFC7230].

An HTTP/1.1 server MUST support chunking for IPP requests, and an IPP Client MUST support chunking for IPP responses according to HTTP/1.1 [RFC7230].

4.1. Printer URI, Job URI, and Job ID

All Printer and Job objects are identified by a Uniform Resource Identifier (URI) [RFC3986] so that they can be persistently and unambiguously referenced. Jobs can also be identified by a combination of Printer URI and Job ID.

Some operation elements are encoded twice, once as the request-target on the HTTP request-line and a second time as a REQUIRED operation attribute in the application/ipp entity. These attributes are the target for the operation and are called "printer-uri" and "job-uri".

Note: The target URI is included twice in an operation referencing the same IPP object, but the two URIs can be different. For example, the HTTP request-target can be relative while the IPP request URI is absolute.

HTTP allows Clients to generate and send a relative URI rather than an absolute URI. A relative URI identifies a resource with the scope of the HTTP server but does not include scheme, host, or port. The following statements characterize how URIs are used in the mapping of IPP onto HTTP:

1. Although potentially redundant, a Client MUST supply the target of the operation both as an operation attribute and as a URI at the HTTP layer. The rationale for this decision is to maintain a consistent set of rules for mapping "application/ipp" to possibly many communication layers, even where URIs are not used as the addressing mechanism in the transport layer.

2. Even though these two URIs might not be literally identical (one being relative and the other being absolute), they MUST both reference the same IPP object.

3. The URI in the HTTP layer is either relative or absolute and is used by the HTTP server to route the HTTP request to the correct resource relative to that HTTP server.

4. Once the HTTP server resource begins to process the HTTP request, it can get the reference to the appropriate IPP Printer object from either the HTTP URI (using to the context of the HTTP server for relative URIs) or from the URI within the operation request; the choice is up to the implementation.

5. HTTP URIs can be relative or absolute, but the target URI in the IPP operation attribute MUST be an absolute URI.

5. IPP URI Schemes

The IPP URI schemes are 'ipp' [RFC3510] and 'ipps' [RFC7472]. Clients and Printers MUST support the ipp-URI value in the following IPP attributes:

- Job attributes:
  - job-uri
  - job-printer-uri
- Printer attributes:
  - printer-uri-supported
- Operation attributes:
  - job-uri
  - printer-uri

Each of the above attributes identifies a Printer or Job. The ipp-URI and ipps-URI are intended as the value of the attributes in this list. All of these attributes have a syntax type of 'uri', but there are attributes with a syntax type of 'uri' that do not use the 'ipp' scheme, e.g., "job-more-info".

If a Printer registers its URI with a directory service, the Printer MUST register an ipp-URI or ipps-URI.

When a Client sends a request, it MUST convert a target ipp-URI to a target http-URL (or ipps-URI to a target https-URI) for the HTTP layer according to the following steps:

1. change the 'ipp' scheme to 'http' or 'ipps' scheme to 'https'; and
2. add an explicit port 631 if the ipp-URL or ipps-URL does not contain an explicit port. Note that port 631 is the IANA-assigned well-known port for the 'ipp' and 'ipps' schemes.

The Client MUST use the target http-URL or https-URL in both the HTTP request-line and HTTP headers, as specified by HTTP [RFC7230]. However, the Client MUST use the target ipp-URI or ipps-URI for the value of the "printer-uri" or "job-uri" operation attribute within the application/ipp body of the request. The server MUST use the

ipp-URI or ipps-URI for the value of the "printer-uri", "job-uri", or "printer-uri-supported" attributes within the application/ipp body of the response.

For example, when an IPP Client sends a request directly, i.e., no proxy, to an ipp-URI "ipp://printer.example.com/ipp/print/myqueue", it opens a TCP connection to port 631 (the IPP implicit port) on the host "printer.example.com" and sends the following data:

```
POST /ipp/print/myqueue HTTP/1.1
Host: printer.example.com:631
Content-type: application/ipp
Transfer-Encoding: chunked
...
"printer-uri" 'ipp://printer.example.com/ipp/print/myqueue'
       (encoded in application/ipp message body)
...
```

Figure 11: Direct IPP Request

As another example, when an IPP Client sends the same request as above via a proxy "myproxy.example.com", it opens a TCP connection to the proxy port 8080 on the proxy host "myproxy.example.com" and sends the following data:

```
POST http://printer.example.com:631/ipp/print/myqueue HTTP/1.1
Host: printer.example.com:631
Content-type: application/ipp
Transfer-Encoding: chunked
...
"printer-uri" 'ipp://printer.example.com/ipp/print/myqueue'
       (encoded in application/ipp message body)
...
```

Figure 12: Proxied IPP Request

The proxy then connects to the IPP origin server with headers that are the same as the "no-proxy" example above.

6. IANA Considerations

The IANA-PRINTER-MIB [RFC3805] has been updated to reference this document; the current version is available from <http://www.iana.org>.

See the IANA Considerations in the document "Internet Printing Protocol/1.1: Model and Semantics" [RFC8011] for information on IANA considerations for IPP extensions. IANA has updated the existing

'application/ipp' media type registration (whose contents are defined in Section 3 "Encoding of the Operation Layer") with the following information.

Type name: application

Subtype name: ipp

Required parameters: N/A

Optional parameters: N/A

Encoding considerations: IPP requests/responses MAY contain long lines and ALWAYS contain binary data (for example, attribute value lengths).

Security considerations: IPP requests/responses do not introduce any security risks not already inherent in the underlying transport protocols. Protocol mixed-version interworking rules in [RFC8011] as well as protocol-encoding rules in this document are complete and unambiguous. See also the security considerations in this document and [RFC8011].

Interoperability considerations: IPP requests (generated by Clients) and responses (generated by servers) MUST comply with all conformance requirements imposed by the normative specifications [RFC8011] and this document. Protocol-encoding rules specified in RFC 8010 are comprehensive so that interoperability between conforming implementations is guaranteed (although support for specific optional features is not ensured). Both the "charset" and "natural-language" of all IPP attribute values that are a LOCALIZED-STRING are explicit within IPP requests/responses (without recourse to any external information in HTTP, SMTP, or other message transport headers).

Published specifications: RFCs 8010 and 8011

Applications that use this media type: Internet Printing Protocol (IPP) print clients and print servers that communicate using HTTP/HTTPS or other transport protocols. Messages of type "application/ipp" are self-contained and transport independent, including "charset" and "natural-language" context for any LOCALIZED-STRING value.

Fragment identifier considerations: N/A

Additional information:

Deprecated alias names for this type: N/A
Magic number(s): N/A
File extension(s): N/A
Macintosh file type code(s): N/A

Person & email address to contact for further information:

ISTO PWG IPP Workgroup <ipp@pwg.org>

Intended usage: COMMON

Restrictions on usage: N/A

Author: ISTO PWG IPP Workgroup <ipp@pwg.org>

Change controller: ISTO PWG IPP Workgroup <ipp@pwg.org>

Provisional registration? (standards tree only): No

7. Internationalization Considerations

See the section on "Internationalization Considerations" in the document "Internet Printing Protocol/1.1: Model and Semantics" [RFC8011] for information on internationalization. This document adds no additional issues.

8. Security Considerations

The IPP Model and Semantics document [RFC8011] discusses high-level security requirements (Client Authentication, Server Authentication, and Operation Privacy). Client Authentication is the mechanism by which the Client proves its identity to the server in a secure manner. Server Authentication is the mechanism by which the server proves its identity to the Client in a secure manner. Operation Privacy is defined as a mechanism for protecting operations from eavesdropping.

Message Integrity is addressed in the document "Internet Printing Protocol (IPP) over HTTPS Transport Binding and the 'ipps' URI Scheme" [RFC7472].

8.1. Security Conformance Requirements

This section defines the security requirements for IPP Clients and IPP objects.

8.1.1. Digest Authentication

IPP Clients and Printers SHOULD support Digest Authentication [RFC7616]. Use of the Message Integrity feature (qop="auth-int") is OPTIONAL.

Note: Previous versions of this specification required support for the MD5 algorithms; however, [RFC7616] makes SHA2-256 mandatory to implement and deprecates MD5, only allowing its use for backwards compatibility reasons. IPP implementations that support Digest Authentication MUST support SHA2-256 and SHOULD support MD5 for backwards compatibility.

Note: The reason that IPP Clients and Printers SHOULD (rather than MUST) support Digest Authentication is that there is a certain class of Output Devices where it does not make sense. Specifically, a low-end device with limited ROM space and low paper throughput may not need Client Authentication. This class of device typically requires firmware designers to make trade-offs between protocols and functionality to arrive at the lowest-cost solution possible. Factored into the designer’s decisions is not just the size of the code, but also the testing, maintenance, usefulness, and time-to-market impact for each feature delivered to the customer. Forcing such low-end devices to provide security in order to claim IPP/1.1 conformance would not make business sense. Print devices that have high-volume throughput and have available ROM space will typically provide support for Client Authentication that safeguards the device from unauthorized access because these devices are prone to a high loss of consumables and paper if unauthorized access occurs.

8.1.2. Transport Layer Security (TLS)

IPP Clients and Printers SHOULD support Transport Layer Security (TLS) [RFC5246] [RFC7525] for Server Authentication and Operation Privacy. IPP Printers MAY also support TLS for Client Authentication. IPP Clients and Printers MAY support Basic Authentication [RFC7617] for User Authentication if the channel is secure, e.g., IPP over HTTPS [RFC7472]. IPP Clients and Printers SHOULD NOT support Basic Authentication over insecure channels.

The IPP Model and Semantics document [RFC8011] defines two Printer attributes ("uri-authentication-supported" and "uri-security-supported") that the Client can use to discover the security policy of a Printer. That document also outlines IPP-specific security considerations and is the primary reference for security implications with regard to the IPP itself.

Note: Because previous versions of this specification did not require TLS support, this version cannot require it for IPP/1.1. However, since printing often involves a great deal of sensitive or private information (medical reports, performance reviews, banking information, etc.) and network monitoring is pervasive ([RFC7258]), implementors are strongly encouraged to include TLS support.

Note: Because IPP Printers typically use self-signed X.509 certificates, IPP Clients SHOULD support Trust On First Use (defined in [RFC7435]) in addition to traditional X.509 certificate validation.

8.2. Using IPP with TLS

IPP uses the "Upgrading to TLS Within HTTP/1.1" mechanism [RFC2817] for 'ipp' URIs. The Client requests a secure TLS connection by using the HTTP "Upgrade" header while the server agrees in the HTTP response. The switch to TLS occurs either because the server grants the Client's request to upgrade to TLS or a server asks to switch to TLS in its response. Secure communication begins with a server's response to switch to TLS.

IPP uses the "HTTPS: HTTP over TLS" mechanism [RFC2818] for 'ipps' URIs. The Client and server negotiate a secure TLS connection immediately and unconditionally.

9. Interoperability with Other IPP Versions

It is beyond the scope of this specification to mandate conformance with versions of IPP other than 1.1. IPP was deliberately designed, however, to make supporting other versions easy. IPP objects (Printers, Jobs, etc.) SHOULD:

- understand any valid request whose major "version-number" is greater than 0; and
- respond appropriately with a response containing the same "version-number" parameter value used by the Client in the request (if the Client-supplied "version-number" is supported) or the highest "version-number" supported by the Printer (if the Client-supplied "version-number" is not supported).

IPP Clients SHOULD:

- understand any valid response whose major "version-number" is greater than 0.

9.1. The "version-number" Parameter

The following are rules regarding the "version-number" parameter (see Section 3.3):

1. Clients MUST send requests containing a "version-number" parameter with the highest supported value, e.g., '1.1', '2.0', etc., and SHOULD try supplying alternate version numbers if they receive a 'server-error-version-not-supported' error return in a response. For example, if a Client sends an IPP/2.0 request that is rejected with the 'server-error-version-not-supported' error and an IPP/1.1 "version-number", it SHOULD retry by sending an IPP/1.1 request.

2. IPP objects (Printers, Jobs, etc.) MUST accept requests containing a "version-number" parameter with a '1.1' value (or reject the request for reasons other than 'server-error-version-not-supported').

3. IPP objects SHOULD either accept requests whose major version is greater than 0 or reject such requests with the 'server-error-version-not-supported' status-code. See Section 4.1.8 of [RFC8011].

4. In any case, security MUST NOT be compromised when a Client supplies a lower "version-number" parameter in a request. For example, if an IPP/2.0 conforming Printer accepts version '1.1' requests and is configured to enforce Digest Authentication, it MUST do the same for a version '1.1' request.

9.2. Security and URI Schemes

The following are rules regarding security, the "version-number" parameter, and the URI scheme supplied in target attributes and responses:

1. When a Client supplies a request, the "printer-uri" or "job-uri" target operation attribute MUST have the same scheme as that indicated in one of the values of the "printer-uri-supported" Printer attribute.

2. When the Printer returns the "job-printer-uri" or "job-uri" Job Description attributes, it SHOULD return the same scheme ('ipp', 'ipps', etc.) that the Client supplied in the "printer-uri" or "job-uri" target operation attributes in the Get-Job-Attributes or Get-Jobs request, rather than the scheme used when the Job was created. However, when a Client requests Job attributes using the Get-Job-Attributes or Get-Jobs operations, the Jobs and Job

attributes that the Printer returns depends on: (1) the security in effect when the Job was created, (2) the security in effect in the query request, and (3) the security policy in force.

3. The Printer MUST enforce its security and privacy policies based on the owner of the IPP object and the URI scheme and/or credentials supplied by the Client in the current request.

10. Changes since RFC 2910

The following changes have been made since the publication of RFC 2910:

- Added references to current IPP extension specifications.
- Added optional support for HTTP/2.
- Added collection attribute syntax from RFC 3382.
- Fixed typographical errors.
- Now reference TLS/1.2 and no longer mandate the TLS/1.0 MTI ciphersuites.
- Updated all references.
- Updated document organization to follow current style.
- Updated example ipp: URIs to follow guidelines in RFC 7472.
- Updated version compatibility for all versions of IPP.
- Updated HTTP Digest Authentication to optional for Clients.
- Removed references to (Experimental) IPP/1.0 and usage of http:/https: URLs.

11. References

11.1. Normative References

[PWG5100.12] Sweet, M. and I. McDonald, "IPP Version 2.0, 2.1, and 2.2", October 2015, <http://ftp.pwg.org/pub/pwg/standards/std-ipp20-20151030-5100.12.pdf>.

[RFC20] Cerf, V., "ASCII format for network interchange", STD 80, RFC 20, DOI 10.17487/RFC0020, October 1969, <http://www.rfc-editor.org/info/rfc20>.

[RFC793] Postel, J., "Transmission Control Protocol", STD 7, RFC 793, DOI 10.17487/RFC0793, September 1981, <http://www.rfc-editor.org/info/rfc793>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

[RFC2579] McCloghrie, K., Ed., Perkins, D., Ed., and J. Schoenwaelder, Ed., "Textual Conventions for SMIv2", STD 58, RFC 2579, DOI 10.17487/RFC2579, April 1999, <http://www.rfc-editor.org/info/rfc2579>.

[RFC2817] Khare, R. and S. Lawrence, "Upgrading to TLS Within HTTP/1.1", RFC 2817, DOI 10.17487/RFC2817, May 2000, <http://www.rfc-editor.org/info/rfc2817>.

[RFC2818] Rescorla, E., "HTTP Over TLS", RFC 2818, DOI 10.17487/RFC2818, May 2000, <http://www.rfc-editor.org/info/rfc2818>.

[RFC2978] Freed, N. and J. Postel, "IANA Charset Registration Procedures", BCP 19, RFC 2978, DOI 10.17487/RFC2978, October 2000, <http://www.rfc-editor.org/info/rfc2978>.

[RFC3510] Herriot, R. and I. McDonald, "Internet Printing Protocol/1.1: IPP URL Scheme", RFC 3510, DOI 10.17487/RFC3510, April 2003, <http://www.rfc-editor.org/info/rfc3510>.

[RFC3629] Yergeau, F., "UTF-8, a transformation format of ISO 10646", STD 63, RFC 3629, DOI 10.17487/RFC3629, November 2003, <http://www.rfc-editor.org/info/rfc3629>.

[RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, DOI 10.17487/RFC3986, January 2005, <http://www.rfc-editor.org/info/rfc3986>.

[RFC5234] Crocker, D., Ed. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, DOI 10.17487/RFC5234, January 2008, <http://www.rfc-editor.org/info/rfc5234>.

[RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, DOI 10.17487/RFC5246, August 2008, <http://www.rfc-editor.org/info/rfc5246>.

[RFC7230] Fielding, R., Ed. and J. Reschke, Ed., "Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing", RFC 7230, DOI 10.17487/RFC7230, June 2014, <http://www.rfc-editor.org/info/rfc7230>.

[RFC7231] Fielding, R., Ed. and J. Reschke, Ed., "Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content", RFC 7231, DOI 10.17487/RFC7231, June 2014, <http://www.rfc-editor.org/info/rfc7231>.

[RFC7232] Fielding, R., Ed. and J. Reschke, Ed., "Hypertext Transfer Protocol (HTTP/1.1): Conditional Requests", RFC 7232, DOI 10.17487/RFC7232, June 2014, <http://www.rfc-editor.org/info/rfc7232>.

[RFC7234] Fielding, R., Ed., Nottingham, M., Ed., and J. Reschke, Ed., "Hypertext Transfer Protocol (HTTP/1.1): Caching", RFC 7234, DOI 10.17487/RFC7234, June 2014, <http://www.rfc-editor.org/info/rfc7234>.

[RFC7235] Fielding, R., Ed. and J. Reschke, Ed., "Hypertext Transfer Protocol (HTTP/1.1): Authentication", RFC 7235, DOI 10.17487/RFC7235, June 2014, <http://www.rfc-editor.org/info/rfc7235>.

[RFC7472] McDonald, I. and M. Sweet, "Internet Printing Protocol (IPP) over HTTPS Transport Binding and the 'ipps' URI Scheme", RFC 7472, DOI 10.17487/RFC7472, March 2015, <http://www.rfc-editor.org/info/rfc7472>.

[RFC7540] Belshe, M., Peon, R., and M. Thomson, Ed., "Hypertext Transfer Protocol Version 2 (HTTP/2)", RFC 7540, DOI 10.17487/RFC7540, May 2015, <http://www.rfc-editor.org/info/rfc7540>.

[RFC7541] Peon, R. and H. Ruellan, "HPACK: Header Compression for HTTP/2", RFC 7541, DOI 10.17487/RFC7541, May 2015, <http://www.rfc-editor.org/info/rfc7541>.

[RFC7616] Shekh-Yusef, R., Ed., Ahrens, D., and S. Bremer, "HTTP Digest Access Authentication", RFC 7616, DOI 10.17487/RFC7616, September 2015, <http://www.rfc-editor.org/info/rfc7616>.

[RFC7617] Reschke, J., "The 'Basic' HTTP Authentication Scheme", RFC 7617, DOI 10.17487/RFC7617, September 2015, <http://www.rfc-editor.org/info/rfc7617>.

[RFC8011] Sweet, M. and I. McDonald, "Internet Printing Protocol/1.1: Model and Semantics", RFC 8011, DOI 10.17487/RFC8011, January 2017, <http://www.rfc-editor.org/info/rfc8011>.

11.2. Informative References

[IANA-IPP] IANA, "Internet Printing Protocol (IPP) Registry", <http://www.iana.org/assignments/ipp-registrations/>.

[PWG5100.3] Ocke, K. and T. Hastings, "Internet Printing Protocol (IPP): Production Printing Attributes - Set1", Candidate Standard 5100.3-2001, February 2001, <http://ftp.pwg.org/pub/pwg/candidates/cs-ippprodprint10-20010212-5100.3.pdf>.

[RFC1179] McLaughlin, L., "Line printer daemon protocol", RFC 1179, DOI 10.17487/RFC1179, August 1990, <http://www.rfc-editor.org/info/rfc1179>.

[RFC7258] Farrell, S. and H. Tschofenig, "Pervasive Monitoring Is an Attack", BCP 188, RFC 7258, DOI 10.17487/RFC7258, May 2014, <http://www.rfc-editor.org/info/rfc7258>.

[RFC7435] Dukhovni, V., "Opportunistic Security: Some Protection Most of the Time", RFC 7435, DOI 10.17487/RFC7435, December 2014, <http://www.rfc-editor.org/info/rfc7435>.

[RFC7525] Sheffer, Y., Holz, R., and P. Saint-Andre, "Recommendations for Secure Use of Transport Layer Security (TLS) and Datagram Transport Layer Security (DTLS)", BCP 195, RFC 7525, DOI 10.17487/RFC7525, May 2015, <http://www.rfc-editor.org/info/rfc7525>.

Appendix A. Protocol Examples

A.1. Print-Job Request

The following is an example of a Print-Job request with "job-name", "copies", and "sides" specified. The "ipp-attribute-fidelity" attribute is set to 'true' so that the print request will fail if the "copies" or the "sides" attribute is not supported or their values are not supported.

Octets	Symbolic Value	Protocol field
0x0101	1.1	version-number
0x0002	Print-Job	operation-id
0x00000001	1	request-id
0x01	start operation-attributes	operation-attributes-tag
0x47	charset type	value-tag
0x0012		name-length
attributes-charset	attributes-charset	name
0x0005		value-length
utf-8	UTF-8	value
0x48	natural-language type	value-tag
0x001b		name-length
attributes-natural-language	attributes-natural-language	name
0x0005		value-length
en-us	en-US	value
0x45	uri type	value-tag
0x000b		name-length
printer-uri	printer-uri	name
0x002c		value-length
ipp://printer.example.com/ipp/print/pinetree	printer pinetree	value
0x42	nameWithoutLanguage type	value-tag
0x0008		name-length
job-name	job-name	name
0x0006		value-length
foobar	foobar	value
0x22	boolean type	value-tag
0x0016		name-length
ipp-attribute-fidelity	ipp-attribute-fidelity	name
0x0001		value-length
0x01	true	value
0x02	start job-attributes	job-attributes-

		tag
0x21	integer type	value-tag
0x0006		name-length
copies	copies	name
0x0004		value-length
0x00000014	20	value
0x44	keyword type	value-tag
0x0005		name-length
sides	sides	name
0x0013		value-length
two-sided-long-edge	two-sided-long-edge	value
0x03	end-of-attributes	end-of-attributes-tag
%!PDF...	<PDF Document>	data

A.2. Print-Job Response (Successful)

Here is an example of a successful Print-Job response to the previous Print-Job request. The Printer supported the "copies" and "sides" attributes and their supplied values. The status-code returned is 'successful-ok'.

Octets	Symbolic Value	Protocol field
0x0101	1.1	version-number
0x0000	successful-ok	status-code
0x00000001	1	request-id
0x01	start operation-attributes	operation-attributes-tag
0x47	charset type	value-tag
0x0012		name-length
attributes-charset	attributes-charset	name
0x0005		value-length
utf-8	UTF-8	value
0x48	natural-language type	value-tag
0x001b		name-length
attributes-natural-language	attributes-natural-language	name
0x0005		value-length
en-us	en-US	value
0x41	textWithoutLanguage type	value-tag
0x000e		name-length
status-message	status-message	name
0x000d		value-length
successful-ok	successful-ok	value
0x02	start job-	job-attributes-

	attributes	tag
0x21	integer	value-tag
0x0006		name-length
job-id	job-id	name
0x0004		value-length
147	147	value
0x45	uri type	value-tag
0x0007		name-length
job-uri	job-uri	name
0x0030		value-length
ipp://printer.example.com/ipp/pr int/pinetree/147	job 147 on pinetree	value
0x23	enum type	value-tag
0x0009		name-length
job-state	job-state	name
0x0004		value-length
0x0003	pending	value
0x03	end-of-attributes	end-of-attributes-tag

A.3. Print-Job Response (Failure)

Here is an example of an unsuccessful Print-Job response to the previous Print-Job request. It fails because, in this case, the Printer does not support the "sides" attribute and because the value '20' for the "copies" attribute is not supported. Therefore, no Job is created, and neither a "job-id" nor a "job-uri" operation attribute is returned. The error code returned is 'client-error-attributes-or-values-not-supported' (0x040b).

Octets	Symbolic Value	Protocol field
0x0101	1.1	version-number
0x040b	client-error-attributes-or-values-not-supported	status-code
0x00000001	1	request-id
0x01	start operation-attributes	operation-attributes tag
0x47	charset type	value-tag
0x0012		name-length
attributes-charset	attributes-charset	name
0x0005		value-length
utf-8	UTF-8	value
0x48	natural-language type	value-tag
0x001b		name-length

attributes-natural-language	attributes-natural-language	name
0x0005		value-length
en-us	en-US	value
0x41	textWithoutLanguage type	value-tag
0x000e		name-length
status-message	status-message	name
0x002f		value-length
client-error-attributes-or-values-not-supported	client-error-attributes-or-values-not-supported	value
0x05	start unsupported-attributes	unsupported-attributes tag
0x21	integer type	value-tag
0x0006		name-length
copies	copies	name
0x0004		value-length
0x00000014	20	value
0x10	unsupported (type)	value-tag
0x0005		name-length
sides	sides	name
0x0000		value-length
0x03	end-of-attributes	end-of-attributes-tag

A.4. Print-Job Response (Success with Attributes Ignored)

Here is an example of a successful Print-Job response to a Print-Job request like the previous Print-Job request, except that the value of "ipp-attribute-fidelity" is 'false'. The print request succeeds, even though, in this case, the Printer supports neither the "sides" attribute nor the value '20' for the "copies" attribute. Therefore, a Job is created and both a "job-id" and a "job-uri" operation attribute are returned. The unsupported attributes are also returned in an Unsupported Attributes group. The error code returned is 'successful-ok-ignored-or-substituted-attributes' (0x0001).

Octets	Symbolic Value	Protocol field
0x0101	1.1	version-number
0x0001	successful-ok-ignored-or-substituted-attributes	status-code
0x00000001	1	request-id
0x01	start operation-attributes	operation-attributes-tag
0x47	charset type	value-tag
0x0012		name-length
attributes-charset	attributes-charset	name

0x0005		value-length
utf-8	UTF-8	value
0x48	natural-language type	value-tag
0x001b		name-length
attributes-natural-language	attributes-natural-language	name
0x0005		value-length
en-us	en-US	value
0x41	textWithoutLanguage type	value-tag
0x000e		name-length
status-message	status-message	name
0x002f		value-length
successful-ok-ignored-or-substituted-attributes	successful-ok-ignored-or-substituted-attributes	value
0x05	start unsupported-attributes	unsupported-attributes tag
0x21	integer type	value-tag
0x0006		name-length
copies	copies	name
0x0004		value-length
0x00000014	20	value
0x10	unsupported (type)	value-tag
0x0005		name-length
sides	sides	name
0x0000		value-length
0x02	start job-attributes	job-attributes-tag
0x21	integer	value-tag
0x0006		name-length
job-id	job-id	name
0x0004		value-length
147	147	value
0x45	uri type	value-tag
0x0007		name-length
job-uri	job-uri	name
0x0030		value-length
ipp://printer.example.com/ipp/print/pinetree/147	job 147 on pinetree	value
0x23	enum type	value-tag
0x0009		name-length
job-state	job-state	name
0x0004		value-length
0x0003	pending	value
0x03	end-of-attributes	end-of-attributes-tag

A.5. Print-URI Request

The following is an example of Print-URI request with "copies" and "job-name" parameters:

Octets	Symbolic Value	Protocol field
0x0101	1.1	version-number
0x0003	Print-URI	operation-id
0x00000001	1	request-id
0x01	start operation-attributes	operation-attributes-tag
0x47	charset type	value-tag
0x0012		name-length
attributes-charset	attributes-charset	name
0x0005		value-length
utf-8	UTF-8	value
0x48	natural-language type	value-tag
0x001b		name-length
attributes-natural-language	attributes-natural-language	name
0x0005		value-length
en-us	en-US	value
0x45	uri type	value-tag
0x000b		name-length
printer-uri	printer-uri	name
0x002c		value-length
ipp://printer.example.com/ipp/print/pinetree	printer pinetree	value
0x45	uri type	value-tag
0x000c		name-length
document-uri	document-uri	name
0x0019		value-length
ftp://foo.example.com/foo	ftp://foo.example.com/foo	value
0x42	nameWithoutLanguage type	value-tag
0x0008		name-length
job-name	job-name	name
0x0006		value-length
foobar	foobar	value
0x02	start job-attributes	job-attributes-tag
0x21	integer type	value-tag
0x0006		name-length
copies	copies	name
0x0004		value-length

0x00000001	1	value
0x03	end-of-attributes	end-of-attributes-tag

A.6. Create-Job Request

The following is an example of Create-Job request with no parameters and no attributes:

Octets	Symbolic Value	Protocol field
0x0101	1.1	version-number
0x0005	Create-Job	operation-id
0x00000001	1	request-id
0x01	start operation-attributes	operation-attributes-tag
0x47	charset type	value-tag
0x0012		name-length
attributes-charset	attributes-charset	name
0x0005		value-length
utf-8	UTF-8	value
0x48	natural-language type	value-tag
0x001b		name-length
attributes-natural-language	attributes-natural-language	name
0x0005		value-length
en-us	en-US	value
0x45	uri type	value-tag
0x000b		name-length
printer-uri	printer-uri	name
0x002c		value-length
ipp://printer.example.com/ipp/print/pinetree	printer pinetree	value
0x03	end-of-attributes	end-of-attributes-tag

A.7. Create-Job Request with Collection Attributes

The following is an example of Create-Job request with the "media-col" collection attribute [PWG5100.3] with the value "media-size={x-dimension=21000 y-dimension=29700} media-type='stationery'":

Octets	Symbolic Value	Protocol field
0x0101	1.1	version-number
0x0005	Create-Job	operation-id
0x00000001	1	request-id

0x01	start operation-attributes	operation-attributes-tag
0x47	charset type	value-tag
0x0012		name-length
attributes-charset	attributes-charset	name
0x0005		value-length
utf-8	UTF-8	value
0x48	natural-language type	value-tag
0x001b		name-length
attributes-natural-language	attributes-natural-language	name
0x0005		value-length
en-us	en-US	value
0x45	uri type	value-tag
0x000b		name-length
printer-uri	printer-uri	name
0x002c		value-length
ipp://printer.example.com/ipp/print/pinetree	printer pinetree	value
0x34	begCollection	value-tag
0x0009	9	name-length
media-col	media-col	name
0x0000	0	value-length
0x4a	memberAttrName	value-tag
0x0000	0	name-length
0x000a	10	value-length
media-size	media-size	value (member-name)
0x34	begCollection	member-value-tag
0x0000	0	name-length
0x0000	0	member-value-length
0x4a	memberAttrName	value-tag
0x0000	0	name-length
0x000b	11	value-length
x-dimension	x-dimension	value (member-name)
0x21	integer	member-value-tag
0x0000	0	name-length
0x0004	4	member-value-length
0x00005208	21000	member-value
0x4a	memberAttrName	value-tag
0x0000	0	name-length
0x000b	11	value-length
y-dimension	y-dimension	value (member-name)

0x21	integer	member-value-tag
0x0000	0	name-length
0x0004	4	member-value-length
0x00007404	29700	member-value
0x37	endCollection	end-value-tag
0x0000	0	end-name-length
0x0000	0	end-value-length
0x4a	memberAttrName	value-tag
0x0000	0	name-length
0x000a	10	value-length
media-type	media-type	value (member-name)
0x44	keyword	member-value-tag
0x0000	0	name-length
0x000a	10	member-value-length
stationery	stationery	member-value
0x37	endCollection	end-value-tag
0x0000	0	end-name-length
0x0000	0	end-value-length
0x03	end-of-attributes	end-of-attributes-tag

A.8. Get-Jobs Request

The following is an example of Get-Jobs request with parameters but no attributes:

Octets	Symbolic Value	Protocol field
0x0101	1.1	version-number
0x000a	Get-Jobs	operation-id
0x0000007b	123	request-id
0x01	start operation-attributes	operation-attributes-tag
0x47	charset type	value-tag
0x0012		name-length
attributes-charset	attributes-charset	name
0x0005		value-length
utf-8	UTF-8	value
0x48	natural-language type	value-tag
0x001b		name-length
attributes-natural-language	attributes-natural-language	name
0x0005		value-length
en-us	en-US	value

0x45	uri type	value-tag
0x000b		name-length
printer-uri	printer-uri	name
0x002c		value-length
ipp://printer.example.com/ipp/print/pinetree	printer pinetree	value
0x21	integer type	value-tag
0x0005		name-length
limit	limit	name
0x0004		value-length
0x00000032	50	value
0x44	keyword type	value-tag
0x0014		name-length
requested-attributes	requested-attributes	name
0x0006		value-length
job-id	job-id	value
0x44	keyword type	value-tag
0x0000	additional value	name-length
0x0008		value-length
job-name	job-name	value
0x44	keyword type	value-tag
0x0000	additional value	name-length
0x000f		value-length
document-format	document-format	value
0x03	end-of-attributes	end-of-attributes-tag

A.9. Get-Jobs Response

The following is an example of a Get-Jobs response from a previous request with three Jobs. The Printer returns no information about the second Job (because of security reasons):

Octets	Symbolic Value	Protocol field
0x0101	1.1	version-number
0x0000	successful-ok	status-code
0x0000007b	123	request-id (echoed back)
0x01	start operation-attributes	operation-attributes-tag
0x47	charset type	value-tag
0x0012		name-length
attributes-charset	attributes-charset	name
0x0005		value-length
utf-8	UTF-8	value
0x48	natural-language type	value-tag
0x001b		name-length

```
   attributes-natural-     attributes-natural-     name
   language                language
   0x0005                                          value-length
   en-us                   en-US                   value
   0x41                    textWithoutLanguage     value-tag
                           type
   0x000e                                          name-length
   status-message          status-message          name
   0x000d                                          value-length
   successful-ok           successful-ok           value
   0x02                    start job-attributes    job-attributes-tag
                           (1st  object)
   0x21                    integer type            value-tag
   0x0006                                          name-length
   job-id                  job-id                  name
   0x0004                                          value-length
   147                     147                     value
   0x36                    nameWithLanguage        value-tag
   0x0008                                          name-length
   job-name                job-name                name
   0x000c                                          value-length
   0x0005                                          sub-value-length
   fr-ca                   fr-CA                   value
   0x0003                                          sub-value-length
   fou                     fou                     name
   0x02                    start job-attributes    job-attributes-tag
                           (2nd object)
   0x02                    start job-attributes    job-attributes-tag
                           (3rd object)
   0x21                    integer type            value-tag
   0x0006                                          name-length
   job-id                  job-id                  name
   0x0004                                          value-length
   148                     149                     value
   0x36                    nameWithLanguage        value-tag
   0x0008                                          name-length
   job-name                job-name                name
   0x0012                                          value-length
   0x0005                                          sub-value-length
   de-CH                   de-CH                   value
   0x0009                                          sub-value-length
   isch guet               isch guet               name
   0x03                    end-of-attributes       end-of-attributes-tag
```

Acknowledgements

The authors would like to acknowledge the following individuals for their contributions to the original IPP/1.1 specifications:

Sylvan Butler, Roger deBry, Tom Hastings, Robert Herriot (the original editor of RFC 2910), Paul Moore, Kirk Ocke, Randy Turner, John Wenn, and Peter Zehler.

Authors' Addresses

Michael Sweet
Apple Inc.
1 Infinite Loop
MS 111-HOMC
Cupertino, CA 95014
United States of America

Email: msweet@apple.com

Ira McDonald
High North, Inc.
PO Box 221
Grand Marais, MI 49839
United States of America

Phone: +1 906-494-2434
Email: blueroofmusic@gmail.com

Internet Engineering Task Force (IETF)
Request for Comments: 8011
Obsoletes: 2911, 3381, 3382
Category: Standards Track
ISSN: 2070-1721

M. Sweet
Apple Inc.
I. McDonald
High North, Inc.

January 2017

Internet Printing Protocol/1.1: Model and Semantics

Abstract

The Internet Printing Protocol (IPP) is an application-level protocol for distributed printing using Internet tools and technologies. This document describes a simplified model consisting of abstract objects, attributes, and operations that is independent of encoding and transport. The model consists of several objects, including Printers and Jobs. Jobs optionally support multiple Documents.

IPP semantics allow End Users and Operators to query Printer capabilities; submit Print Jobs; inquire about the status of Print Jobs and Printers; and cancel, hold, and release Print Jobs. IPP semantics also allow Operators to pause and resume Jobs and Printers.

Security, internationalization, and directory issues are also addressed by the model and semantics. The IPP message encoding and transport are described in "Internet Printing Protocol/1.1: Encoding and Transport" (RFC 8010).

This document obsoletes RFCs 2911, 3381, and 3382.

Status of This Memo

This is an Internet Standards Track document.

This document is a product of the Internet Engineering Task Force (IETF). It represents the consensus of the IETF community. It has received public review and has been approved for publication by the Internet Engineering Steering Group (IESG). Further information on Internet Standards is available in Section 2 of RFC 7841.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at http://www.rfc-editor.org/info/rfc8011.

Table of Contents

1. Introduction

The Internet Printing Protocol (IPP) is an application-level protocol for distributed printing using Internet tools and technologies. IPP version 1.1 (IPP/1.1) focuses primarily on End User functionality with a few administrative operations included. IPP versions 2.0, 2.1, and 2.2 provide many new operations and are defined separately.

This document is just one of a suite of documents that fully define IPP. The full set of IETF IPP documents includes:

Design Goals for an Internet Printing Protocol [RFC2567]

Rationale for the Structure of the Model and Protocol for the Internet Printing Protocol [RFC2568]

Internet Printing Protocol/1.1: Model and Semantics (this document)

Internet Printing Protocol/1.1: Encoding and Transport [RFC8010]

Internet Printing Protocol/1.1: Implementor's Guide [RFC3196]

Internet Printing Protocol/1.1: IPP URL Scheme [RFC3510]

Internet Printing Protocol (IPP) over HTTPS Transport Binding and the 'ipps' URI Scheme [RFC7472]

Internet Printing Protocol (IPP): Requirements for Job, Printer, and Device Administrative Operations [RFC3239]

Internet Printing Protocol (IPP): Job and Printer Set Operations [RFC3380]

Internet Printing Protocol (IPP): Job and Printer Administrative Operations [RFC3998]

Internet Printing Protocol (IPP): Requirements for IPP Notifications [RFC3997]

Internet Printing Protocol (IPP): Event Notifications and Subscriptions [RFC3995]

Internet Printing Protocol (IPP): The 'ippget' Delivery Method for Event Notifications [RFC3996]

Mapping between LPD and IPP Protocols [RFC2569]

Anyone reading these documents for the first time is strongly encouraged to read the IPP documents in the above order. Additional IPP specifications have been published by the IEEE-ISTO Printer Working Group's IPP Workgroup [PWG-IPP-WG]. The following standards are highly recommended reading:

PWG Media Standardized Names 2.0 (MSN2) [PWG5101.1]

IPP Finishings 2.0 (FIN) [PWG5100.1]

Internet Printing Protocol (IPP): "output-bin" attribute extension [PWG5100.2]

Internet Printing Protocol (IPP): Production Printing Attributes - Set 1 [PWG5100.3] (for "media-col" Job Template attribute)

Standard for The Internet Printing Protocol (IPP): Document Object [PWG5100.5]

Standard for The Internet Printing Protocol (IPP): Page Overrides [PWG5100.6]

Standard for The Internet Printing Protocol (IPP): Job Extensions [PWG5100.7]

Standard for Internet Printing Protocol (IPP): "-actual" attributes [PWG5100.8]

Internet Printing Protocol (IPP): Printer State Extensions v1.0 [PWG5100.9]

Internet Printing Protocol (IPP): Job and Printer Extensions - Set 2 (JPS2) [PWG5100.11]

IPP Version 2.0, 2.1, and 2.2 [PWG5100.12]

IPP: Job and Printer Extensions - Set 3 (JPS3) [PWG5100.13]

IPP Everywhere [PWG5100.14]

IPP FaxOut Service [PWG5100.15]

IPP Transaction-Based Printing Extensions [PWG5100.16]

IPP Scan Service (SCAN) [PWG5100.17]

IPP Shared Infrastructure Extensions (INFRA) [PWG5100.18]

IPP Implementor’s Guide v2.0 (IG) [PWG5100.19]

This document is organized as follows:

- The rest of Section 1 is an introduction to the IPP simplified model for distributed printing;
- Section 2 defines the terminology and conventions used within this document;
- Section 3 introduces the object types covered in this document with their basic behaviors, attributes, and interactions;
- Section 4 defines the core operations for IPP/1.1. IPP operations are synchronous -- each operation has both a request and a response;
- Section 5 defines the core attributes (and their syntaxes) that are used in the model;

- Sections 6 and 7 summarize the implementation conformance requirements for objects that support the protocol and IANA considerations, respectively;
- Sections 8 and 9 cover the internationalization and security considerations for IPP; and
- The appendices provide a reference for status-code values, processing of IPP attributes, and the generic directory schema.

1.1. Simplified Printing Model

In order to achieve its goal of realizing a workable printing protocol for the Internet, the Internet Printing Protocol (IPP) is based on a simplified printing model that abstracts the many components of real-world printing solutions. The Internet is a distributed computing environment where requesters of print services (Clients, applications, Printer drivers, etc.) cooperate and interact with print service providers. This document (sometimes referred to here as the "Model and Semantics" document) describes a simple, abstract model for IPP even though the underlying configurations can be complex "n-tier" client/server systems. An important simplifying step in the IPP Model is to expose only the key objects and interfaces required for printing. The model described in this document does not include features, interfaces, and relationships that are beyond the scope of IPP/1.1. IPP/1.1 incorporates many of the relevant ideas and lessons learned from other specification and development efforts [HTPP] [ISO10175] [LDPA] [P1387.4] [PSIS] [RFC1179] [SWP]. IPP is heavily influenced by the printing model introduced in the Document Printing Application (DPA) [ISO10175] standard. Although DPA specifies both End User and administrative features, IPP/1.1 focuses primarily on End User functionality with a few additional OPTIONAL operations for Administrators and Operators.

The IPP Model encapsulates the important components of distributed printing into the following IPP object types:

- Printer (Section 3.1)
- Job (Section 3.2)
- Document (see [PWG5100.5])
- Subscription (see [RFC3995])

Each object type has an associated set of operations (see Section 4) and attributes (see Section 5).

It is important, however, to understand that in real system implementations (which lie underneath the abstracted IPP Model), there are other components of a print service that are not explicitly defined in the IPP Model. The following figure illustrates where IPP fits with respect to these other components.

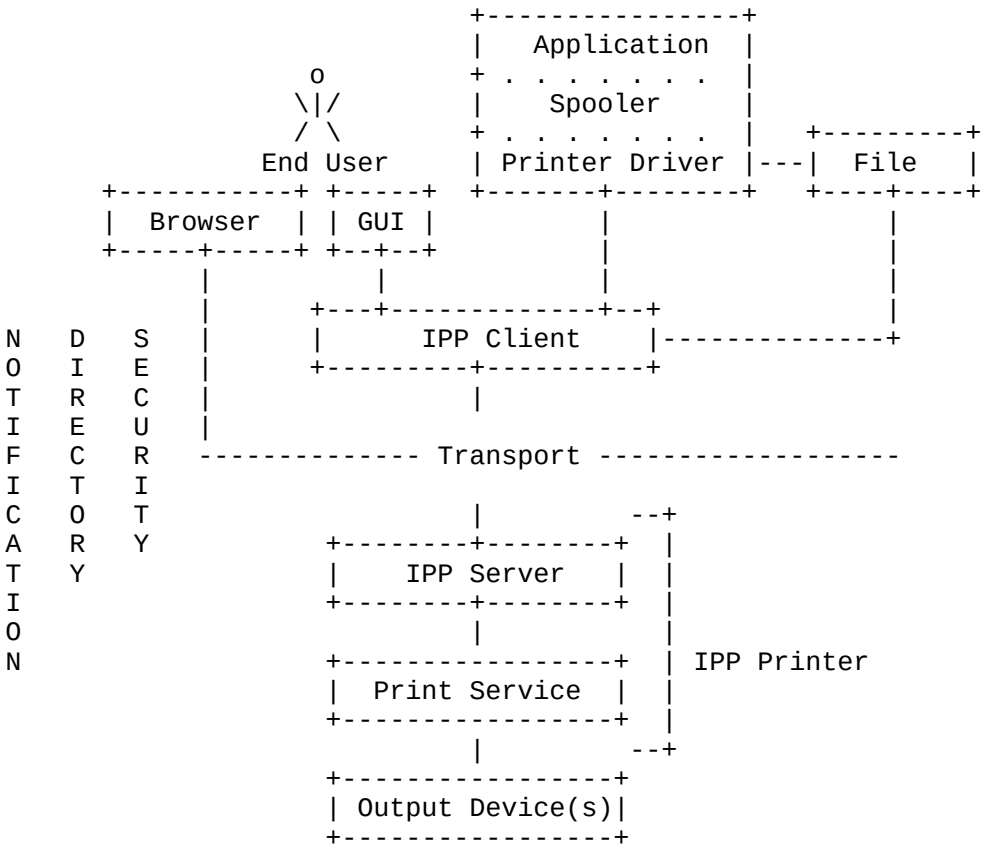


Figure 1: IPP Model

An IPP Printer object ("Printer") encapsulates the functions normally associated with physical Output Devices along with the spooling, scheduling, and multiple device management functions often associated with a print server. Printers are optionally registered as entries in a directory where End Users find and select them based on some sort of filtered context-based searching mechanism (see Appendix D). The directory is used to store relatively static information about the Printer, allowing End Users to search for and find Printers that match their search criteria -- for example, name, location, context, Printer capabilities, etc. The more dynamic information, such as

state, currently loaded and ready media, number of Jobs at the Printer, errors, warnings, and so forth, is directly associated with the Printer itself rather than with the entry in the directory, which only references the Printer.

IPP Clients ("Clients") implement IPP on the Client side and give End Users (or programs running on behalf of End Users) the ability to query Printers and submit and manage Print Jobs. An IPP server is just that part of the Printer object that implements the server-side protocol. The rest of the Printer object implements (or gateways into) the application semantics of the print service itself. Printers can be embedded in an Output Device or can be implemented on a host on the network that communicates with an Output Device.

When a Job is submitted to the Printer and the Printer has validated the attributes in the submission request, the Printer creates a new IPP Job object ("Job"). The End User then interacts with this new Job to query its status and monitor the progress of the Job. An End User can also cancel their Print Jobs by using the Job's Cancel-Job operation. An End User can also hold, release, and restart their Print Jobs using the Job's OPTIONAL Hold-Job, Release-Job, and Restart-Job operations, if implemented.

A privileged Operator or Administrator of a Printer can cancel, hold, release, and restart any user's Job using the REQUIRED Cancel-Job and the OPTIONAL Hold-Job, Release-Job, and Restart-Job operations. In addition, a privileged Operator or Administrator of a Printer can pause, resume, or purge (Jobs from) a Printer using the OPTIONAL Pause-Printer, Resume-Printer, and Purge-Jobs operations, if implemented.

The notification service is defined in "Internet Printing Protocol (IPP): Event Notifications and Subscriptions" [RFC3995]. By using such a notification service, the End User is able to register for and receive Printer-specific and Job-specific events asynchronously. Otherwise, an End User can query the status of Printers and can follow the progress of Jobs by polling using the Get-Printer-Attributes, Get-Jobs, and Get-Job-Attributes operations.

2. Conventions Used in This Document

2.1. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

The key word "DEPRECATED" in this document refers to an operation, attribute, or value that SHOULD NOT be used or supported in new implementations.

2.2. Printing Terminology

Client: Initiator of outgoing IPP session requests and sender of outgoing IPP operation requests (Hypertext Transfer Protocol (HTTP/1.1) user agent, as defined in [RFC7230]).

Document: An object created and managed by a Printer that contains description, processing, and status information. A Document object can have attached data and is bound to a single Job [PWG5100.5].

'ipp' URI: An IPP URI as defined in [RFC3510].

'ipps' URI: An IPP URI as defined in [RFC7472].

Job: An object created and managed by a Printer that contains description, processing, and status information. The Job also contains zero or more Document objects.

Logical Device: A print server, software service, or gateway that processes Jobs and either forwards or stores the processed Job or uses one or more Physical Devices to render output.

Output Device: A single Logical or Physical Device.

Physical Device: A hardware implementation of an endpoint device, e.g., a marking engine, a fax modem, etc.

Printer: Listener for incoming IPP session requests and receiver of incoming IPP operation requests (HTTP/1.1 server, as defined in [RFC7230]) that represents one or more Physical Devices or a Logical Device.

2.3. Model Terminology

2.3.1. Administrator

An End User who is also authorized to manage all aspects of an Output Device or Printer, including creating the Printer instances and controlling the authorization of other End Users and Operators [RFC2567].

2.3.2. Attributes

An attribute is an item of information that is associated with an instance of an IPP object (Printer, Job, etc.). An attribute consists of an attribute name and one or more attribute values. Each attribute has a specific attribute syntax. All object attributes are defined in Section 5, and all operation attributes are defined in Section 4.

Job Template attributes are described in Section 5.2. The Client optionally supplies Job Template attributes in a Job Creation request (operation requests that create Job objects). The Printer object has associated attributes that define supported and default values for the Printer.

2.3.2.1. Attribute Group Name

Related attributes are grouped into named groups. The name of the group is a keyword. The group name can be used in place of naming all the attributes in the group explicitly. Attribute groups are defined in Section 4.

2.3.2.2. Attribute Name

Each attribute is uniquely identified in this document by its attribute name. An attribute name is a keyword. The keyword attribute name is given in the section title in this document that describes that attribute. In running text in this document, attribute names are indicated inside double quotation marks (") where the quotation marks are not part of the keyword itself.

2.3.2.3. Attribute Syntax

Each attribute is defined using an explicit syntax type. In this document, each syntax type is defined as a keyword with specific meaning. The "Encoding and Transport" document [RFC8010] indicates the actual "on-the-wire" encoding rules for each syntax type. Attribute syntax types are defined in Section 5.1.

2.3.2.4. Attribute Value

Each attribute has one or more values. Attribute values are represented in the syntax type specified for that attribute. In running text in this document, attribute values are indicated inside single quotation marks (’), whether their attribute syntax is keyword, integer, text, etc. where the quotation marks are not part of the value itself.

2.3.3. End User

An End User is a person or software process that is authorized to perform basic printing functions, including finding/locating a Printer, creating a local instance of a Printer, viewing Printer status, viewing Printer capabilities, submitting a Print Job, viewing Print Job status, and altering the attributes of a Print Job [RFC2567].

2.3.4. Impression

An Impression is the content imposed upon one side of a Media Sheet by a marking engine, independent of the number of times that the sheet side passes any marker. An Impression contains one or more Input Pages that are imposed (scaled, translated, and/or rotated) during processing of the Document data.

2.3.5. Input Page

An Input Page is a page according to the definition of "pages" in the language used to express the Document data.

2.3.6. Job Creation Operation

A Job Creation operation is any operation that causes the creation of a Job object, e.g., the Create-Job, Print-Job, and Print-URI operations defined in this document.

2.3.7. Keyword

Keywords are used within this document as identifiers of semantic entities within the abstract model (see Section 5.1.4). Attribute names, some attribute values, attribute syntaxes, and attribute group names are represented as keywords.

2.3.8. Media Sheet

A Media Sheet is a single instance of a medium, whether printing on one or both sides of the medium. Media Sheets also include sections of roll media.

2.3.9. Operator

An Operator is an End User that also has special rights on the Output Device or Printer. The Operator typically monitors the status of the Printer and also manages and controls the Jobs at the Output Device [RFC2567]. The Operator is allowed to query and control the Printer, Jobs, and Documents based on site policy.

2.3.10. Set

A Set is a logical boundary between the delivered Media Sheets of a printed Job. For example, in the case of a ten-page single Document with collated pages and a request for 50 copies, each of the 50 printed copies of the Document constitutes a Set. If the pages were uncollated, then 50 copies of each of the individual pages within the Document would represent each Set. Finishing processes operate on Sets.

2.3.11. Support of Attributes

By definition, a Printer supports an attribute only if that Printer accepts it in a request or responds with the corresponding attribute populated with some value(s) in a response to a query for that attribute. A Printer supports an attribute value if the value is one of the Printer's "supported values" attributes. The device behind a Printer can exhibit a behavior that corresponds to some IPP attribute, but if the Printer, when queried for that attribute, doesn't respond with the attribute, then as far as IPP is concerned, that implementation does not support that feature. If the Printer's "xxx-supported" attribute is not populated with a particular value (even if that value is a legal value for that attribute), then that Printer does not support that particular value.

A conforming implementation supports all REQUIRED attributes. However, even for REQUIRED attributes, conformance to IPP does not mandate that all implementations support all possible values representing all possible Job processing behaviors and features. For example, if a given instance of a Printer supports only certain Document formats, then that Printer responds with the "document-format-supported" attribute populated with a set of values, or possibly only one value, taken from the entire set of possible values defined for that attribute. This limited set of values

represents the Printer's set of supported Document formats. Supporting an attribute and some set of values for that attribute enables IPP End Users to be aware of and make use of those features associated with that attribute and those values. If an implementation chooses to not support an attribute or some specific value, then IPP End Users would have no ability to make use of that feature within the context of IPP itself. However, due to existing practice and legacy systems that are not IPP aware, there might be some other mechanism outside the scope of IPP to control or request the "unsupported" feature (such as embedded instructions within the Document data itself).

For example, consider the following for the "finishings-supported" attribute.

1) If a Printer is not physically capable of stapling, the "finishings-supported" attribute MUST NOT be populated with the value of 'staple'.

2) A Printer is physically capable of stapling; however, an implementation chooses not to support stapling in the IPP "finishings" attribute. In this case, 'staple' MUST NOT be a value in the "finishings-supported" Printer Description attribute. Without support for the value 'staple', an IPP End User would have no means within the protocol itself to request that a Job be stapled. However, an existing Document data formatter might be able to request that the Document be stapled directly with an embedded instruction within the Document data. In this case, the IPP implementation does not "support" stapling; however, the End User is still able to have some control over the stapling of the completed Job.

3) A Printer is physically capable of stapling, and an implementation chooses to support stapling in the IPP "finishings" attribute. In this case, 'staple' MUST be a value in the "finishings-supported" Printer attribute. Doing so enables End Users to be aware of and make use of the stapling feature using IPP attributes.

Even though support for Job Template attributes by a Printer is OPTIONAL in IPP/1.1, Printers whose associated device(s) is capable of realizing any feature or function that corresponds to an IPP attribute and some associated value SHOULD support that IPP attribute and value.

The set of values in any of the supported value attributes is set (populated) by some administrative process or automatic sensing mechanism that is outside the scope of this document. For administrative policy and control reasons, an Administrator can choose to make only a subset of possible values visible to the End User. In this case, the real Output Device behind the IPP Printer abstraction can be capable of a certain feature; however, an Administrator is specifying that access to that feature not be exposed to the End User through IPP. Also, since a Printer can represent a logical print device (not just a Physical Device), the actual process for supporting a value is undefined and left up to the implementation. However, if a Printer supports a value, some manual human action might be needed to realize the semantic action associated with the value, but no End User action is required.

For example, if one of the values in the "finishings-supported" attribute is 'staple', the actual process might be an automatic staple action by a Physical Device controlled by some command sent to the device. Or, the actual process of stapling might be a manual action by an Operator at an Operator-attended Printer.

For another example of how supported attributes function, consider an Administrator who desires to control all Print Jobs so that no Job sheets are printed in order to conserve paper. To force no Job sheets, the Administrator sets the only supported value for the "job-sheets-supported" attribute to 'none'. In this case, if a Client requests anything except 'none', the Job Creation request is rejected or the "job-sheets" value is ignored (depending on the value of "ipp-attribute-fidelity"). To force the use of Job start/end sheets on all Jobs, the Administrator does not include the value 'none' in the "job-sheets-supported" attribute. In this case, if a Client requests 'none', the Job Creation request is rejected or the "job-sheets" value is ignored (again depending on the value of "ipp-attribute-fidelity").

Job Template attributes will typically have corresponding "xxx-supported" and "xxx-default" Printer Description attributes that contain the supported and default values for the attribute. For capabilities that are not associated with a Job, the convention is to have an "xxx-supported" Printer Description attribute that lists the supported values and an "xxx-configured" Printer Description attribute that contains the value being used by the Printer. For example, the "charset-supported" Printer Description attribute (Section 5.4.18) lists the supported character sets for the Printer while the "charset-configured" Printer Description attribute (Section 5.4.17) specifies the character set being used by the Printer.

2.3.12. Terminating State

The final state for a Job or other object is called its Terminating State. For example, the 'aborted', 'canceled', and 'completed' Job states are Terminating States.

2.4. Abbreviations

ABNF: Augmented Backus-Naur Form [RFC5234]

ASCII: American Standard Code for Information Interchange [RFC20]

HTTP: Hypertext Transfer Protocol [RFC7230]

HTTPS: HTTP over TLS [RFC2818]

IANA: Internet Assigned Numbers Authority

IEEE: Institute of Electrical and Electronics Engineers

IESG: Internet Engineering Steering Group

IPP: Internet Printing Protocol (this document, [RFC8010], and [PWG5100.12])

ISTO: IEEE Industry Standards and Technology Organization

LPD: Line Printer Daemon Protocol [RFC1179]

PWG: IEEE-ISTO Printer Working Group

RFC: Request for Comments

TCP: Transmission Control Protocol [RFC793]

TLS: Transport Layer Security [RFC5246]

URI: Uniform Resource Identifier [RFC3986]

URL: Uniform Resource Locator [RFC3986]

UTF-8: Unicode Transformation Format - 8-bit [RFC3629]

3. IPP Objects

This document defines IPP objects of types Printer and Job. Each type of object models relevant aspects of a real-world entity such as a real Printer or real Print Job. Each object type is defined as a set of possible attributes that can be supported by instances of that object type. For each object (instance), the actual set of supported attributes and values describe a specific implementation. The object's attributes and values describe its state, capabilities, realizable features, Job processing functions, and default behaviors and characteristics. For example, the Printer object type is defined as a set of attributes that each Printer object potentially supports. In the same manner, the Job object type is defined as a set of attributes that are potentially supported by each Job object.

Each attribute included in the set of attributes defining an object type is labeled as:

- "REQUIRED": each object MUST support the attribute.
- "RECOMMENDED": each object SHOULD support the attribute.
- "OPTIONAL": each object MAY support the attribute.

Some definitions of attribute values indicate that an object MUST or SHOULD support the value; otherwise, support of the value is OPTIONAL. However, if an implementation supports an attribute, it MUST support at least one of the possible values for that attribute.

3.1. Printer Object

The major component of the IPP Model is the Printer object. A Printer object implements the server side of the IPP/1.1 protocol. Using the protocol, End Users can query the attributes of the Printer object and submit Print Jobs to the Printer object. The actual implementation components behind the Printer abstraction can take on different forms and different configurations. However, the model abstraction allows the details of the configuration of real components to remain opaque to the End User. Section 4 describes each of the Printer operations in detail.

The capabilities and state of a Printer object are described by its attributes. Printer attributes are divided into two groups:

- "job-template" attributes: These attributes describe supported Job processing capabilities and defaults for the Printer object (see Section 5.2)

- "printer-description" attributes: These attributes describe the Printer's identification, state, location, references to other sources of information about the Printer object, etc. (see Section 5.4)

Since a Printer object is an abstraction of a generic Document Output Device and print service provider, a Printer object could be used to represent any real or virtual device with semantics consistent with the Printer object, such as a fax device, an imager, or even a CD writer.

Some examples of configurations supporting a Printer object include:

1. An Output Device with no spooling capabilities

2. An Output Device with a built-in spooler

3. A print server supporting IPP with one or more associated Output Devices

    3a. The associated Output Devices are or are not capable of spooling Jobs

    3b. The associated Output Devices possibly support IPP

Figure 2 shows some examples of how Printers can be realized on top of various distributed printing configurations. The embedded case below represents configurations 1 and 2 above. The "hosted Printer" and "fan out" items represent configurations 3a and 3b, respectively.

In this document, the term "Client" refers to a software entity that sends IPP operation requests to an IPP Printer and accepts IPP operation responses. A Client MAY be:

1. contained within software controlled by an End User, e.g., activated by the "Print" menu item in an application, or

2. the print server component that sends IPP requests to either an Output Device or another "downstream" print server.

The term "IPP Printer" is a network entity that accepts IPP operation requests and returns IPP operation responses. As such, an IPP Printer object MAY be:

1. an (embedded) device component that accepts IPP requests and controls the device, or

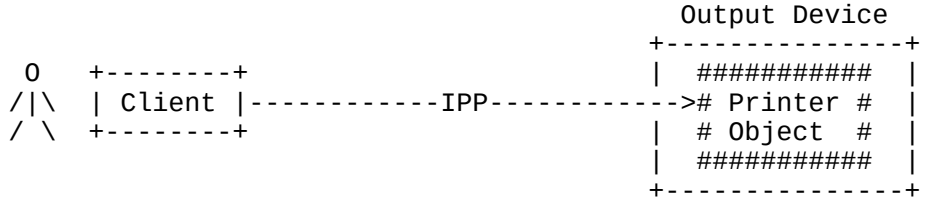
2. a component of a print server that accepts IPP requests (where the print server controls one or more networked devices using IPP or other protocols).

Legend:

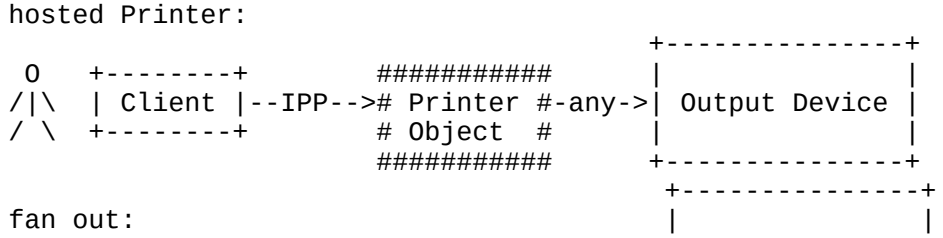
indicates a Printer object that is either embedded in an Output Device or hosted in a server. The Printer object might or might not be capable of queuing/spooling.

any indicates any network protocol or direct connect, including IPP

embedded Printer:



hosted Printer:



fan out:

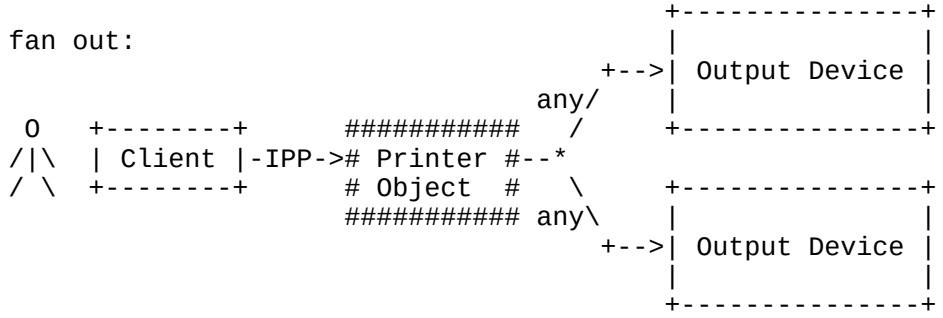


Figure 2: IPP Printer Object Architecture

3.2. Job Object

A Job object is used to model a Print Job. A Job object contains zero or more Documents. The information required to create a Job object is sent in a Job Creation request from the End User via an IPP Client to the Printer. The Printer validates the Job Creation request, and if the Printer accepts the request, the Printer creates the new Job object. Section 4 describes each of the Job operations in detail.

The characteristics and state of a Job object are described by its attributes. Job attributes are grouped into two groups as follows:

- "job-template" attributes: These attributes can be supplied by the Client or End User and include Job processing instructions that are intended to override any Printer defaults and/or instructions embedded within the Document data (see Section 5.2)

- "job-description" attributes: These attributes describe the Job's identification, state, size, etc. The Client supplies some of these attributes, and the Printer generates others (see Section 5.3)

An implementation MUST support at least one Document per Job object. An implementation MAY support multiple Documents per Job object. A Document is either:

- a stream of Document data in a format supported by the Printer (typically a Page Description Language -- PDL), or

- a reference to such a stream of Document data.

All Job processing instructions are modeled as Job object attributes. These attributes are called "Job Template attributes", and they apply equally to all Documents within a Job object.

3.3. Object Relationships

IPP objects have relationships that are maintained persistently along with the persistent storage of the object attributes.

A Printer object can represent either one or more physical Output Devices or a Logical Device that "processes" Jobs but never actually uses a physical Output Device to put marks on paper. Examples of Logical Devices include a web page publisher or a gateway into an online Document archive or repository. A Printer contains zero or more Job objects.

A Job object is contained by exactly one Printer; however, the identical Document data associated with a Job could be sent to either the same or a different Printer. In this case, a second Job object would be created that would be almost identical to the first Job; however, it would have new (different) Job object identifiers (see Section 3.4).

A Job either is empty (before any Documents have been added) or contains one or more Documents. If the contained Document is a stream of Document data, that stream can be contained in only one Document. However, there can be identical copies of the stream in other Documents in the same or different Jobs. If the contained Document is just a reference to a stream of Document data, other Documents (in the same or different Job(s)) contain the same reference.

3.4. Object Identity

All IPP objects (Printers, Jobs, etc.) are identified by a Uniform Resource Identifier (URI) [RFC3986] so that they can be persistently and unambiguously referenced. Since every URL is a specialized form of a URI, even though the more generic term "URI" is used throughout the rest of this document, its usage is intended to cover the more specific notion of "URL" as well.

An Administrator configures Printers to either support or not support authentication and/or message privacy using Transport Layer Security (TLS) [RFC5246]; the mechanism for security configuration is outside the scope of this document. In some situations, both types of connections (both authenticated and unauthenticated) can be established using a single communication channel that has some sort of negotiation mechanism. In other situations, multiple communication channels are used, one for each type of security configuration. Section 9 provides a full description of all security considerations and configurations.

If a Printer supports more than one communication channel, some or all of those channels might support and/or require different security mechanisms. In such cases, an Administrator could expose the simultaneous support for these multiple communication channels as multiple URIs for a single Printer where each URI represents one of the communication channels to the Printer. To support this flexibility, the IPP Printer object type defines a multi-valued identification attribute called the "printer-uri-supported" attribute that MUST contain at least one URI. The "printer-uri-supported" attribute has two companion attributes, the "uri-security-supported" attribute and the "uri-authentication-supported" attribute. Both have the same cardinality as "printer-uri-supported". The purpose of

the "uri-security-supported" attribute is to indicate the security mechanisms (if any) used for each URI listed in "printer-uri-supported". The purpose of the "uri-authentication-supported" attribute is to indicate the authentication mechanisms (if any) used for each URI listed in "printer-uri-supported". These three attributes are fully described in Sections 5.4.1, 5.4.2, and 5.4.3.

When a Job is submitted to the Printer via a Job Creation request, the Client supplies only a single Printer URI. The Client-supplied Printer URI MUST be one of the values in the "printer-uri-supported" Printer attribute.

IPP/1.1 does not specify how the Client obtains the Client-supplied URI, but it is RECOMMENDED that a Printer be registered as an entry in a directory service. End Users and programs can then interrogate the directory, searching for Printers. Appendix D defines a generic schema for Printer object entries in the directory service and describes how the entry acts as a bridge to the actual IPP Printer. The entry in the directory that represents the IPP Printer includes the possibly many URIs for that Printer as values in one of its attributes.

When a Client submits a Job Creation request to the Printer, the Printer validates the request and creates a new Job object. The Printer assigns the new Job a numeric identifier that is stored in the "job-id" Job attribute and a URI that is stored in the "job-uri" Job attribute. Both the numeric identifier and URI can then be used by Clients as the target for subsequent Job operations; the numeric identifier is preferred. The Printer generates the Job numeric identifier and URI based on its configured security policy and the URI used by the Client in the Job Creation request.

For example, consider a Printer that supports both a communication channel secured by the use of TLS (using HTTP over TLS with an "https" schemed URI) and another open communication channel that is not secured with TLS (using a simple "http" schemed URI). If a Client submits a Job using the secure URI, the Printer assigns the new Job a secure URI as well. If a Client were to submit a Job using the open-channel URI, the Printer might assign the new Job an open-channel URI. Clients SHOULD use the "printer-uri" and "job-id" attributes to target a Job to avoid any ambiguity about the security of the communication channel.

In addition, the Printer also populates the Job's "job-printer-uri" attribute. This is a reference back to the Printer that created the Job. If a Client only has access to a Job's "job-uri" identifier, the Client can query the Job's "job-printer-uri" attribute in order

to determine which Printer created the Job. If the Printer supports more than one URI, the Printer picks the one URI supplied by the Client when creating the Job to build the value for and to populate the Job's "job-printer-uri" attribute.

In addition to identifiers, IPP objects have names -- "printer-name" for Printers and "job-name" for Jobs. An object name is not guaranteed to be unique across all instances of all objects. A Printer's name is chosen and set by an Administrator through some mechanism outside the scope of this document. A Job's name can be chosen and supplied by the Client submitting the Job. If the Client does not supply a Job name, the Printer generates a name for the new Job. In all cases, the name only has local meaning.

To summarize:

- Each Printer is identified by one or more URIs. The Printer's "printer-uri-supported" attribute contains the URI(s).
- The Printer's "uri-security-supported" attribute identifies the communication channel security protocols that have been configured for the various Printer URIs (e.g., 'tls' or 'none').
- The Printer's "uri-authentication-supported" attribute identifies the authentication mechanisms that have been configured for the various Printer URIs (e.g., 'digest', 'none', etc.).
- Each Job is identified by a numeric identifier, which is a 32-bit positive integer. The Job's "job-id" attribute contains the Job ID. The Job ID is only unique within the context of the Printer that created the Job.
- Each Job is also identified by a URI. The Job's "job-uri" attribute contains the URI, although its use by Clients is DEPRECATED.
- Each Job has a "job-printer-uri" attribute, which contains the URI of the Printer that was used to create the Job. This attribute is used to determine the Printer that created a Job when given only the URI for the Job. This linkage is necessary to determine the languages, charsets, and operations that are supported on that Job (the basis for such support comes from the creating Printer).
- Each Printer has a name, which is not necessarily unique. The Administrator chooses and sets this name through some mechanism outside the scope of this IPP/1.1 document. The Printer's "printer-name" attribute contains the name.

- Each Job has a name, which is not necessarily unique. The Client optionally supplies this name in the Job Creation request. If the Client does not supply this name, the Printer generates a name for the Job. The Job's "job-name" attribute contains the name.

4. IPP Operations

IPP objects (Printers, Jobs, etc.) support operations. An operation consists of a request and a response. When a Client communicates with a Printer or its Jobs, the Client issues an operation request to the Printer URI and object's numeric identifier, if needed. Operation requests and responses have parameters that identify the operation. Operations also have attributes that affect the runtime characteristics of the operation (the intended target, localization information, etc.). These operation-specific attributes are called "operation attributes" (as compared to object attributes such as Printer attributes or Job attributes). Each request carries along with it any operation attributes, object attributes, and/or Document data required to perform the operation. Each request requires a response from the object. Each response indicates success or failure of the operation with a status-code as a response parameter. The response contains any operation attributes, object attributes, and/or status messages generated during the execution of the operation request.

This section describes the semantics of the IPP operations, both requests and responses, in terms of the parameters, attributes, and other data associated with each operation.

The Printer operations defined in this document are:

Print-Job (Section 4.2.1)

Print-URI (Section 4.2.2)

Validate-Job (Section 4.2.3)

Create-Job (Section 4.2.4)

Get-Printer-Attributes (Section 4.2.5)

Get-Jobs (Section 4.2.6)

Pause-Printer (Section 4.2.7)

Resume-Printer (Section 4.2.8)

Purge-Jobs (Section 4.2.9)

The Job operations defined in this document are:

Send-Document (Section 4.3.1)

Send-URI (Section 4.3.2)

Cancel-Job (Section 4.3.3)

Get-Job-Attributes (Section 4.3.4)

Hold-Job (Section 4.3.5)

Release-Job (Section 4.3.6)

Restart-Job (Section 4.3.7)

The Send-Document and Send-URI Job operations are used to add Documents to an existing Job created using the Create-Job operation.

4.1. Common Semantics

All IPP operations require some common parameters and operation attributes. These common elements and their semantic characteristics are defined and described in more detail in the following sections.

4.1.1. Required Parameters

Every operation request contains the following REQUIRED parameters:

- a "version-number",
- an "operation-id",
- a "request-id", and
- the attributes that are REQUIRED for that type of request.

Every operation response contains the following REQUIRED parameters:

- a "version-number",
- a "status-code",
- the "request-id" that was supplied in the corresponding request, and
- the attributes that are REQUIRED for that type of response.

The Encoding and Transport document [RFC8010] defines special rules for the encoding of these parameters. All other operation elements are represented using the more generic encoding rules for attributes and groups of attributes.

4.1.2. Operation IDs and Request IDs

Each IPP operation request includes an identifying "operation-id" value. Valid values are defined in the "operations-supported" Printer attribute section (see Section 5.4.15). The Client specifies which operation is being requested by supplying the correct "operation-id" value.

In addition, every invocation of an operation is identified by a "request-id" value. For each request, the Client chooses the "request-id", which MUST be an integer (possibly unique, depending on Client requirements) in the range from 1 to 2**31 - 1 (inclusive). This "request-id" allows Clients to manage multiple outstanding requests. The receiving IPP object (Printer, Job, etc.) copies all 32 bits of the Client-supplied "request-id" attribute into the response so that the Client can match the response with the correct outstanding request, even if the "request-id" is out of range. If the request is terminated before the complete "request-id" is received, the IPP object rejects the request and returns a response with a "request-id" of 0.

Note: In some cases, the transport protocol underneath IPP might be a connection-oriented protocol that would make it impossible for a Client to receive responses in any order other than the order in which the corresponding requests were sent. In such cases, the "request-id" attribute would not be essential for correct protocol operation. However, in other transport mappings the operation responses could come back in any order, in which case the "request-id" is essential.

4.1.3. Attributes

Operation requests and responses are both composed of groups of attributes and/or Document data. The attribute groups are:

- Operation Attributes: These attributes are passed in the operation and affect the IPP object’s behavior while processing the operation request, and they can affect other attributes or groups of attributes. Some operation attributes describe the Document data associated with the Print Job and are associated with new Job objects; however, most operation attributes do not persist beyond the life of the operation. The description of each operation attribute includes conformance statements indicating which

operation attributes are REQUIRED and which are OPTIONAL for an IPP object to support, as well as which attributes a Client MUST supply in a request and an IPP object MUST supply in a response.

- Job Template Attributes: These attributes affect the processing of a Job. A Client MAY supply Job Template attributes in a Job Creation request, and the receiving object MUST be prepared to receive all supported attributes. The Job object can later be queried to find out what Job Template attributes were originally requested in the Job Creation request, and such attributes are returned in the response as Job object attributes. The Printer object can be queried about its Job Template attributes to find out what type of Job processing capabilities are supported and/or what the default Job processing behaviors are, though such attributes are returned in the response as Printer object attributes. The "ipp-attribute-fidelity" operation attribute affects processing of all Client-supplied Job Template attributes -- see Section 4.2.1.1 and Appendix C for a full description of "ipp-attribute-fidelity" and its relationship to other attributes.

- Job Object Attributes: These attributes are returned in response to a query operation directed at a Job object.

- Printer Object Attributes: These attributes are returned in response to a query operation directed at a Printer object.

- Unsupported Attributes: In a Job Creation request, the Client supplies a set of operation and Job Template attributes. If any of these attributes or their values are unsupported by the Printer object, the Printer object SHOULD return the set of unsupported attributes in the response. Section 4.1.7, Section 4.2.1.2, and Appendix C give a full description of how Job Template attributes supplied by the Client in a Job Creation request are processed by the Printer object and how unsupported attributes are returned to the Client. Because of extensibility, any IPP object might receive a request that contains new or unknown attributes or values for which it has no support. In such cases, the IPP object processes what it can and returns the unsupported attributes in the response. The Unsupported Attributes group is defined for all operation responses for returning unsupported attributes that the Client supplied in the request.

Later in this section, each operation is formally defined by identifying the allowed and expected groups of attributes for each request and response. The model identifies a specific order for each group in each request or response, but the attributes within each group can be in any order, unless specified otherwise.

The attributes within a group MUST be unique; if an attribute with the same name occurs more than once, the group is malformed. Clients MUST NOT submit such malformed requests, and Printers MUST NOT return such malformed responses. If such a malformed request is submitted to a Printer, the Printer MUST either (1) reject the request with the 'client-error-bad-request' status-code (RECOMMENDED -- see Appendix B.1.4.1) or (2) process the request normally after selecting only one of the attribute instances, depending on implementation. Which attribute is selected when there are duplicate attributes depends on implementation. The IPP Printer MUST NOT use the values from more than one such duplicate attribute instance.

Each attribute definition includes the attribute's name followed by the name of its attribute syntax(es) in parentheses. In addition, each 'integer' attribute can be followed by the allowed range in parentheses, (m:n), for values of that attribute. Each 'text' or 'name' attribute can be followed by the maximum size in octets in parentheses, (size), for values of that attribute. For more details on attribute syntax notation, see the descriptions of these attribute syntaxes in Section 5.1.

Note: Document data included in the operation is not strictly an attribute, but it is treated as a special attribute group for ordering purposes. The only operations defined in this document that support supplying the Document data within an operation request are Print-Job and Send-Document. There are no operations defined in this document whose responses include Document data.

Some operations are REQUIRED for IPP objects to support; the others are OPTIONAL (see Section 6.2.2). Therefore, before using an OPTIONAL operation, a Client SHOULD first use the REQUIRED Get-Printer-Attributes operation to query the Printer's "operations-supported" attribute in order to determine which OPTIONAL operations are actually supported. The Client SHOULD NOT use an OPTIONAL operation that is not supported. When an IPP object receives a request to perform an operation it does not support, it MUST return the 'server-error-operation-not-supported' status-code (see Appendix B.1.5.2). An IPP object is non-conformant if it does not support a REQUIRED operation.

4.1.4. Character Set and Natural Language Operation Attributes

Some Job and Printer attributes have values that are text strings and names intended for human understanding rather than machine understanding (see the 'text' and 'name' attribute syntax descriptions in Section 5.1). The following sections describe two special operation attributes called "attributes-charset" and "attributes-natural-language" whose values are used when interpreting

other attributes using the 'text' and 'name' attribute syntaxes. For Job Creation operations, the IPP Printer implementation also saves these two attributes with the new Job object as Job Status attributes.

The "attributes-charset" and "attributes-natural-language" attributes MUST be the first two attributes in every IPP request and response, as part of the initial Operation Attributes group of the IPP message. The "attributes-charset" attribute MUST be the first attribute in the group, and the "attributes-natural-language" attribute MUST be the second attribute in the group.

For the sake of brevity in this document, these operation attribute descriptions are not repeated with every operation request and response but instead have a reference back to this section.

4.1.4.1. Request Operation Attributes

The Client MUST supply and the Printer object MUST support the following REQUIRED operation attributes in every IPP operation request:

"attributes-charset" (charset):

This operation attribute identifies the charset (coded character set and encoding method) used by any 'text' and 'name' attributes that the Client is supplying in this request. It also identifies the charset that the Printer object MUST use (if supported) for all 'text' and 'name' attributes and status messages that the Printer object returns in the response to this request. See Sections 5.1.2 and 5.1.3 for the definitions of the 'text' and 'name' attribute syntaxes.

All Clients and IPP objects MUST support the 'utf-8' charset [RFC3629] and MAY support additional charsets, provided that they are registered with IANA [RFC2978] [IANA-CS]. If the Printer object does not support the Client-supplied charset value, the Printer object MUST reject the request, set the "attributes-charset" to 'utf-8' in the response, and return the 'client-error-charset-not-supported' status-code and any 'text' or 'name' attributes using the 'utf-8' charset. The Printer MAY return any attributes in the Unsupported Attributes group (see Sections 4.1.7 and 4.2.1.2). The Printer object MUST indicate the charset(s) supported as the values of the "charset-supported" Printer attribute (see Section 5.4.18), so that the Client can query to determine which charset(s) is supported.

Note to Client implementors: Since IPP objects are only required to support the 'utf-8' charset, in order to maximize interoperability with multiple IPP object implementations, a Client SHOULD supply 'utf-8' in the "attributes-charset" operation attribute, even though the Client is only passing and able to present a simpler charset, such as US-ASCII [RFC20] or ISO-8859-1 [ISO8859-1]. Then the Client will have to filter out, perform charset conversion on, or replace those characters that are returned in the response that it cannot present to its user. On the other hand, if both the Client and the IPP objects also support a charset in common besides 'utf-8', the Client can use that charset in order to avoid charset conversion or data loss.

See the 'charset' attribute syntax description in Section 5.1.8 for the syntax and semantic interpretation of the values of this attribute and for example values.

"attributes-natural-language" (naturalLanguage):

This operation attribute identifies the natural language [RFC5646] used by any 'text' and 'name' attributes that the Client is supplying in this request. This attribute also identifies the natural language that the Printer object SHOULD use for all 'text' and 'name' attributes and status messages that the Printer object returns in the response to this request. See the 'naturalLanguage' attribute syntax description in Section 5.1.9 for the syntax and semantic interpretation of the values of this attribute and for example values.

There are no REQUIRED natural languages required for the Printer object to support. However, the Printer's "generated-natural-language-supported" attribute identifies the natural languages supported by the Printer object and any contained Jobs for all text strings generated by the IPP object. A Client MAY query this attribute to determine which natural language(s) is supported for generated messages.

For any of the attributes for which the Printer object generates text, i.e., for the "job-state-message", "printer-state-message", and status messages (see Section 4.1.6), the Printer object MUST be able to generate these text strings in any of its supported natural languages. If the Client requests a natural language that is not supported, the Printer object MUST return these generated messages in the Printer's configured natural language as specified by the Printer's "natural-language-configured" attribute (see Section 5.4.19).

For other 'text' and 'name' attributes supplied by the Client, authentication system, Operator, Administrator, or manufacturer (i.e., for "job-originating-user-name", "printer-name" (name), "printer-location" (text), "printer-info" (text), and "printer-make-and-model" (text)), the Printer object is only required to support the configured natural language of the Printer identified by the Printer's "natural-language-configured" attribute, though support of additional natural languages for these attributes is permitted.

For any 'text' or 'name' attribute in the request that is in a different natural language than the value supplied in the "attributes-natural-language" operation attribute, the Client MUST use the Natural Language Override mechanism (see Sections 5.1.2.2 and 5.1.3.2) for each such attribute value supplied. The Client MAY use the Natural Language Override mechanism redundantly, i.e., use it even when the value is in the same natural language as the value supplied in the "attributes-natural-language" operation attribute of the request.

The IPP object MUST accept any natural language and any Natural Language Override, whether the IPP object supports that natural language or not (and independent of the value of the "ipp-attribute-fidelity" operation attribute). That is, the IPP object accepts all Client-supplied values no matter what the values are in the Printer's "generated-natural-language-supported" attribute. That attribute, "generated-natural-language-supported", only applies to generated messages, not Client-supplied messages. The IPP object MUST remember that natural language for all Client-supplied attributes, and when returning those attributes in response to a query, the IPP object MUST indicate that natural language.

Each value whose attribute syntax type is 'text' or 'name' (see Sections 5.1.2 and 5.1.3) has an Associated Natural Language. This document does not specify how this association is stored in a Printer or Job object. When such a value is encoded in a request or response, the natural language is either implicit or explicit:

* In the implicit case, the value contains only the text/name value, and the language is specified by the "attributes-natural-language" operation attribute in the request or response (see Sections 5.1.2.1 and 5.1.3.1).

* In the explicit case (also known as the Natural Language Override case), the value contains both the language and the text/name value (see Sections 5.1.2.2 and 5.1.3.2).

For example, the "job-name" attribute MAY be supplied by the Client in a Job Creation request. The text value for this attribute will be in the natural language identified by the "attribute-natural-language" attribute, or if different, as identified by the Natural Language Override mechanism. If supplied, the IPP object will use the value of the "job-name" attribute to populate the Job's "job-name" attribute. Whenever any Client queries the Job's "job-name" attribute, the IPP object returns the attribute as stored and uses the Natural Language Override mechanism to specify the natural language, if it is different from that reported in the "attributes-natural-language" operation attribute of the response. The IPP object MAY use the Natural Language Override mechanism redundantly, i.e., use it even when the value is in the same natural language as the value supplied in the "attributes-natural-language" operation attribute of the response.

An IPP object MUST NOT reject a request based on a supplied natural language in an "attributes-natural-language" operation attribute or in any attribute that uses the Natural Language Override.

Note: Supplying 'text' or 'name' attributes with an incompatible combination of natural language and charset can cause undesired behavior. For example, suppose a Printer supports charsets 'utf-8', 'iso-8859-1', and 'iso-8859-7'. Suppose also that it supports natural languages 'en' (English), 'fr' (French), and 'el' (Greek). Although the Printer supports the charset 'iso-8859-1' and natural language 'el', it probably does not support the combination of Greek text strings using the 'iso-8859-1' charset. The Printer handles this apparent incompatibility differently, depending on the context in which it occurs:

* In a Job Creation request: If the Client supplies a 'text' or 'name' attribute (for example, the "job-name" operation attribute) that uses an apparently incompatible combination, it is a Client choice that does not affect the Printer or its correct operation. Therefore, the Printer simply accepts the Client-supplied value, stores it with the Job, and responds back with the same combination whenever the Client (or any Client) queries for that attribute.

* In a query-type operation, like Get-Printer-Attributes: If the Client requests an apparently incompatible combination, the Printer responds (as described in Section 4.1.4.2) using the Printer's configured natural language rather than the natural language requested by the Client.

In either case, the Printer does not reject the request because of the apparent incompatibility. The potential incompatible combination of charset and natural language can occur either at the global operation level or at the Natural Language Override attribute-by-attribute level. In addition, since the response always includes explicit charset and natural language information, there is never any question or ambiguity in how the Client interprets the response.

4.1.4.2. Response Operation Attributes

The Printer MUST supply and the Client MUST support the following REQUIRED operation attributes in every IPP/1.1 operation response:

"attributes-charset" (charset):

This operation attribute identifies the charset used by any 'text' and 'name' attributes that the Printer object is returning in this response. The value in this response MUST be the same value as the "attributes-charset" operation attribute supplied by the Client in the request. If this is not possible (i.e., the charset requested is not supported), the request would have been rejected. See "attributes-charset" described in Section 4.1.4.1 above.

If the Printer object supports more than just the 'utf-8' charset, the Printer object MUST be able to perform code conversion between each of the charsets supported on a "highest fidelity possible" basis in order to return the 'text' and 'name' attributes in the charset requested by the Client. However, some information loss can occur during the charset conversion, depending on the charsets involved. For example, depending on implementation, the Printer object can convert from a UTF-8 'a' to a US-ASCII 'a' (with no loss of information); from an ISO Latin 1 CAPITAL LETTER A WITH ACUTE ACCENT to US-ASCII 'A' (losing the accent); or from a UTF-8 Japanese Kanji character to some ISO Latin 1 error character indication such as '?', a decimal code equivalent, or the absence of a character.

Whether an implementation that supports more than one charset stores the data in the charset supplied by the Client or performs code conversion to one of the other supported charsets depends on implementation. The strategy should try to minimize loss of information during code conversion. On each response, such an implementation converts from its internal charset to that requested.

"attributes-natural-language" (naturalLanguage):

This operation attribute identifies the natural language used by any 'text' and 'name' attributes that the IPP object is returning in this response. Unlike the "attributes-charset" operation attribute, the IPP object MAY return the natural language of the Job object or the Printer's configured natural language as identified by the Printer's "natural-language-configured" attribute, rather than the natural language supplied by the Client. For any 'text' or 'name' attribute or status message in the response that is in a different natural language than the value returned in the "attributes-natural-language" operation attribute, the IPP object MUST use the Natural Language Override mechanism (see Sections 5.1.2.2 and 5.1.3.2) on each attribute value returned. The IPP object MAY use the Natural Language Override mechanism redundantly, i.e., use it even when the value is in the same natural language as the value supplied in the "attributes-natural-language" operation attribute of the response.

4.1.5. Operation Targets

All IPP operations are directed at IPP objects. For Printer operations, the operation is always directed at a Printer object using one of its URIs, i.e., one of the values in the Printer's "printer-uri-supported" attribute. Even if the Printer object supports more than one URI, the Client supplies only one URI as the target of the operation. The Client identifies the target object by supplying the correct URI in the "printer-uri" operation attribute.

For Job operations, the operation is directed at either:

- The Printer object that created the Job object using the Printer object's URI and the Job's numeric identifier (Job ID). Since the Printer object that created the Job object generated the Job ID, it MUST be able to correctly associate the Client-supplied Job ID with the correct Job object. The Client supplies the Printer's URI in the "printer-uri" operation attribute and the Job ID in the "job-id" operation attribute.

- The Job object itself using the Job's URI. In this case, the Client identifies the target object by supplying the correct URI in the "job-uri" (uri) operation attribute (Section 5.3.2).

Clients SHOULD send the "printer-uri" and "job-id" operation attributes in Job operations.

If the operation is directed at the Job object directly using the Job's URI, the Client MUST NOT include the redundant "job-id" operation attribute.

The operation target attributes are REQUIRED operation attributes that are included in every operation request. Like the charset and natural language attributes (see Section 4.1.4), the operation target attributes are specially ordered operation attributes. In all cases, the operation target attributes immediately follow the "attributes-charset" and "attributes-natural-language" attributes within the Operation Attributes group; however, the specific ordering rules are as follows:

- In the case where there is only one operation target attribute (i.e., either only the "printer-uri" attribute or only the "job-uri" attribute), that attribute MUST be the third attribute in the Operation Attributes group.

- In the case where Job operations use two operation target attributes (i.e., the "printer-uri" and "job-id" attributes), the "printer-uri" attribute MUST be the third attribute and the "job-id" attribute MUST be the fourth attribute.

In all cases, the target URIs contained within the body of IPP operation requests and responses MUST be in absolute format rather than relative format (a relative URL identifies a resource with the scope of the HTTP server, but does not include scheme, host, or port).

The following rules apply to the use of port numbers in URIs that identify IPP objects:

1. If the URI scheme allows the port number to be explicitly included in the URI string, and a port number is specified within the URI, then that port number MUST be used by the Client to contact the IPP object.

2. If the URI scheme allows the port number to be explicitly included in the URI string, and a port number is not specified within the URI, then the default port number implied by that URI scheme MUST be used by the Client to contact the IPP object.

3. If the URI scheme does not allow an explicit port number to be specified within the URI, then the default port number implied by that URI MUST be used by the Client to contact the IPP object.

Note: "Internet Printing Protocol/1.1: IPP URL Scheme" [RFC3510] and "Internet Printing Protocol (IPP) over HTTPS Transport Binding and the 'ipps' URI Scheme" [RFC7472] define the mapping of IPP onto HTTP and HTTPS, respectively, and define and register a default port number.

4.1.6. Operation Response Status-Code Values and Status Messages

Every operation response includes a REQUIRED "status-code" parameter, SHOULD include the "status-message" operation attribute, and MAY include the "detailed-status-message" operation attribute. The Print-URI and Send-URI response MAY also include the "document-access-error" operation attribute.

4.1.6.1. "status-code" (type2 enum)

The REQUIRED "status-code" parameter provides information on the processing of a request.

The status-code is intended for use by automata. A Client implementation of IPP SHOULD convert status-code values into any localized message that has semantic meaning to the End User.

The "status-code" value is a numeric value that has semantic meaning. The "status-code" syntax is similar to a "type2 enum" (see Section 5.1 ("Attribute Syntaxes")), except that values can range only from 0x0000 to 0x7fff. Appendix B describes and assigns the status-code values, and suggests a corresponding status message for each status-code for use by the Client when the user's natural language is English.

If the Printer performs an operation with no errors and it encounters no problems, it MUST return the status-code 'successful-ok' in the response. See Appendix B.

If the Client supplies unsupported values for the following parameters or operation attributes, the Printer object MUST reject the operation, MAY return the unsupported attribute value in the Unsupported Attributes group, and MUST return the indicated status-code:

Parameter/Attribute	Status-Code
version-number	server-error-version-not-supported
operation-id	server-error-operation-not-supported
attributes-charset	client-error-charset-not-supported
compression	client-error-compression-not-supported
document-format	client-error-document-format-not-supported
document-uri	client-error-uri-scheme-not-supported, client-error-document-access-error

Table 1: Status-Code Values for All Requests

If the Client supplies unsupported values for other attributes, or unsupported attributes, the Printer returns the status-code defined in Section 4.1.7 ("Unsupported Attributes").

4.1.6.2. "status-message" (text(255))

The RECOMMENDED "status-message" operation attribute provides a short textual description of the status of the operation. The "status-message" attribute's syntax is "text(255)", so the maximum length is 255 octets (see Section 5.1.2). The status message is intended for the human End User. If a response does include a "status-message" attribute, an IPP Client can examine or display the messages in some implementation-specific manner. The "status-message" attribute is especially useful for a later version of a Printer to return as supplemental information for the human user, to accompany a status-code that an earlier version of a Client might not understand.

If the Printer supports the "status-message" operation attribute, it MUST be able to generate this message in any of the natural languages identified by the Printer's "generated-natural-language-supported" attribute and MUST honor any supported value for the "attributes-natural-language" operation attribute (Section 4.1.4.1)

of the Client request. Appendix B suggests the text for the status message returned by the Printer for use with the English natural language.

As described in Section 4.1.4.1, for any returned 'text' attribute, if there is a choice for generating this message, the Printer uses the natural language indicated by the value of "attributes-natural-language" in the Client request, if supported; otherwise, the Printer uses the value in the Printer's own "natural-language-configured" attribute.

If the Printer supports the "status-message" operation attribute, it SHOULD use the REQUIRED 'utf-8' charset to return a status message for the following error status-code values (see Appendix B): 'client-error-bad-request', 'client-error-charset-not-supported', 'server-error-internal-error', 'server-error-operation-not-supported', and 'server-error-version-not-supported'. In this case, it MUST set the value of the "attributes-charset" operation attribute to 'utf-8' in the error response.

4.1.6.3. "detailed-status-message" (text(MAX))

The OPTIONAL "detailed-status-message" operation attribute provides additional more-detailed technical and implementation-specific information about the operation for Administrators or other experienced technical people. The "detailed-status-message" attribute's syntax is "text(MAX)", so the maximum length is 1023 octets (see Section 5.1.2). If the Printer supports the "detailed-status-message" operation attribute, the Printer SHOULD localize the message, unless such localization would obscure the technical meaning of the message. Clients MUST NOT attempt to parse the value of this attribute. See the "document-access-error" operation attribute (Section 4.1.6.4) for additional errors that a program can process.

4.1.6.4. "document-access-error" (text(MAX))

This OPTIONAL operation attribute provides additional information about any Document access errors encountered by the Printer before it returned a response to the Print-URI (Section 4.2.2) or Send-URI (Section 4.3.2) operation. For errors in the protocol identified by the URI scheme in the "document-uri" operation attribute, such as 'http:' or 'ftp:', the error code is returned in parentheses, followed by the URI. For example:

(404) http://www.example.com/filename.pdf

Most Internet protocols use decimal error codes (unlike IPP), so the ASCII error code representation is in decimal.

4.1.7. Unsupported Attributes

The Unsupported Attributes group contains attributes that are not supported by the operation. This group is primarily for the Job Creation operations, but all operations can return this group.

A Printer MUST include an Unsupported Attributes group in a response if the status-code is one of the following: 'successful-ok-ignored-or-substituted-attributes', 'successful-ok-conflicting-attributes', 'client-error-attributes-or-values-not-supported', or 'client-error-conflicting-attributes'.

If the status-code is one of the four specified in the preceding paragraph, the Unsupported Attributes group MUST contain all of those attributes and only those attributes that are:

a. an operation or Job Template attribute supplied in the request, and

b. unsupported by the Printer. See below for details on the three categories of "unsupported" attributes.

If the status-code is one of those in Table 1 in Section 4.1.6.1, the OPTIONAL Unsupported Attributes group contains the unsupported parameter or attribute indicated in that table.

If the Printer is not returning any unsupported attributes in the response, the Printer SHOULD omit the Unsupported Attributes group rather than sending an empty group. However, a Client MUST be able to accept an empty group.

Unsupported attributes fall into three categories:

1. The Printer does not support the supplied attribute (no matter what the attribute syntax or value).

2. The Printer does support the attribute, but it does not support some or all of the particular attribute syntaxes or values supplied by the Client, i.e., the Printer does not have those attribute syntaxes or values in its corresponding "xxx-supported" attribute.

3. The Printer does support the attributes and values supplied, but the particular values are in conflict with one another, because they violate a constraint, such as not being able to staple transparencies.

In the case of an unsupported attribute name, the Printer returns the Client-supplied attribute with a substituted value of 'unsupported'. This value's syntax type is "out-of-band", and its encoding is defined by special rules for "out-of-band" values in the Encoding and Transport document [RFC8010]. Its value indicates no support for the attribute itself -- see the beginning of Section 5.1 in this document.

In the case of a supported attribute with one or more unsupported attribute syntaxes or values, the Printer simply returns the Client-supplied attribute with the unsupported attribute syntaxes or values as supplied by the Client. This indicates support for the attribute but no support for that particular attribute syntax or value. If the Client supplies a multi-valued attribute with more than one value and the Printer supports the attribute but only supports a subset of the Client-supplied attribute syntaxes or values, the Printer MUST return only those attribute syntaxes or values that are unsupported.

In the case of two (or more) supported attribute values that are in conflict with one another (although each is supported independently, the values conflict when requested together within the same Job), the Printer MUST return all the values that it ignores or substitutes to resolve the conflict but not any of the values that it is still using. The choice for exactly how to resolve the conflict is implementation dependent. See Section 4.2.1.2 and Appendix C. See the Implementor's Guides [RFC3196] [PWG5100.19] for examples.

4.1.8. Versions

Each operation request and response carries with it a "version-number" parameter. Each value of the "version-number" parameter is in the form "X.Y" where X is the major version number and Y is the minor version number. By including a version number in the Client request, it allows the Client to identify which version of IPP it is interested in using, i.e., the version whose conformance requirements the Client can depend upon the Printer to meet.

If the IPP object does not support that major version number supplied by the Client, i.e., the "major version number" portion of the "version-number" parameter does not match any of the values of the Printer's "ipp-versions-supported" attribute (see Section 5.4.14), the object MUST respond with a status-code of

'server-error-version-not-supported' along with the closest version number that is supported (see Appendix B.1.5.4). If the major version number is supported but the minor version number is not, the IPP object SHOULD accept the request and attempt to perform it (or reject the request if the operation is not supported); otherwise, it rejects the request and returns the 'server-error-version-not-supported' status-code. In all cases, the IPP object MUST return the "version-number" value that it supports that is closest to the version number supplied by the Client in the request.

There is no version negotiation per se. However, if a Client has received a 'server-error-version-not-supported' status-code from an IPP object, the Client SHOULD try again with a different version number. A Client MAY also determine the versions supported either from a directory that conforms to Appendix D or by querying the Printer's "ipp-versions-supported" attribute (see Section 5.4.14) to determine which versions are supported.

An IPP/1.1 object implementation MUST support version '1.1', i.e., meet the conformance requirements for IPP/1.1 as specified in this document and [RFC8010]. IPP implementations SHOULD accept any request with the major version '1' or '2', or reject the request if the operation is not supported.

There is only one notion of "version number" that covers both IPP Model and IPP protocol changes. Changes to the major version number of the Model and Semantics document can indicate structural or syntactic changes that make it impossible for older versions of IPP Clients and Printers to correctly parse and correctly process the new or changed attributes, operations, and responses. If the major version number changes, the minor version number is set to zero. As an example, adding the REQUIRED "ipp-attribute-fidelity" attribute to version '1.1' (if it had not been part of version '1.0') would have required a change to the major version number, since an IPP/1.0 Printer would not have processed a request with the correct semantics that contained the "ipp-attribute-fidelity" attribute that it did not know about. Items that might affect the changing of the major version number include any changes to the Model and Semantics document (this document) or the Encoding and Transport document [RFC8010] itself, such as:

- reordering of ordered attributes or attribute sets
- changes to the syntax of existing attributes
- adding REQUIRED (for an IPP object to support) Operation Attributes groups

- adding values to existing REQUIRED operation attributes
- adding REQUIRED operations

Changes to the minor version number indicate the addition of new features, attributes, and attribute values that might not be understood by all IPP objects but that can be ignored if not understood. Items that might affect the changing of the minor version number include any changes to the model objects and attributes but not the encoding and transport rules [RFC8010] (except adding attribute syntaxes). Examples of such changes are:

- grouping all extensions not included in a previous version into a new version
- adding new attribute values
- adding new object attributes
- adding OPTIONAL (for an IPP object to support) operation attributes (i.e., those attributes that an IPP object can ignore without confusing Clients)
- adding OPTIONAL (for an IPP object to support) Operation Attributes groups (i.e., those attributes that an IPP object can ignore without confusing Clients)
- adding new attribute syntaxes
- adding OPTIONAL operations
- changing Job attributes or Printer attributes from OPTIONAL to REQUIRED or vice versa
- adding OPTIONAL attribute syntaxes to an existing attribute

The encoding [RFC8010] of the "version-number" parameter MUST NOT change over any version number (either major or minor). This rule guarantees that all future versions will be backwards compatible with all previous versions (at least for checking the "version-number" parameter). In addition, any protocol elements (attributes, error codes, tags, etc.) that are not carried forward from one version to the next are DEPRECATED so that they can never be reused with new semantics.

Implementations that support a certain version SHOULD support all previous Standards Track versions. As each new version is defined (through the release of a new IPP specification document), that version will specify which previous versions MUST and which versions SHOULD be supported in compliant implementations.

4.1.9. Job Creation Operations

In order to "submit a Print Job" and create a new Job, a Client issues a Job Creation request. A Job Creation request is any one of the following three operation requests:

- The Print-Job Request: A Client that wants to submit a Print Job with only a single Document can use the Print-Job operation. The operation allows for the Client to "push" the Document data to the Printer by including the Document data in the request itself. Note that Clients SHOULD instead use the Create-Job and Send-Document requests, if supported by the Printer, since they allow for Job monitoring and control during submission of the Document data.

- The Print-URI Request: A Client that wants to submit a Print Job with only a single Document (where the Printer "pulls" the Document data instead of the Client "pushing" the data to the Printer) uses the Print-URI operation. In this case, the Client includes in the request only a URI reference to the Document data (not the Document data itself).

- The Create-Job Request: A Client that wants to submit a Print Job with zero or more Documents uses the Create-Job operation. This operation is followed by an arbitrary number of Send-Document and/or Send-URI operations, each creating another Document for the newly created Job. The Send-Document operation includes the Document data in the request (the Client "pushes" the Document data to the Printer), and the Send-URI operation includes only a URI reference to the Document data in the request (the Printer "pulls" the Document data from the referenced location). The last Send-Document or Send-URI request for a given Job includes a "last-document" operation attribute set to ’true’ indicating that this is the last request.

Throughout this document, the term "Job Creation request" is used to refer to any of these three operation requests.

A Create-Job operation followed by only one Send-Document operation is semantically equivalent to a Print-Job operation; however, the Client SHOULD use the Create-Job and Send-Document operations (when supported) for all Jobs with a single Document to allow for reliable

Job control and monitoring. Print-Job is a REQUIRED operation (all implementations MUST support it), whereas Create-Job is a RECOMMENDED operation and hence some implementations might not support it.

Job submission time is the point in time when a Client issues a Job Creation request. The initial state of every Job is the 'pending', 'pending-held', or 'processing' state (see Section 5.3.7). When the Printer begins processing the Print Job, the Job's state moves to 'processing'. This is known as Job processing time.

At Job submission time and at the time a Validate-Job operation is received, the Printer MUST do the following:

1. Process the Client-supplied attributes and either accept or reject the request

2. Validate the syntax of and support for the scheme of any Client-supplied URI

At Job submission time, the Printer MUST validate whether the supplied attributes, attribute syntaxes, and values are supported by matching them with the Printer's corresponding "xxx-supported" attributes. See Section 4.1.7 for details. See the Implementor's Guides [RFC3196] [PWG5100.19] for guidance on processing Job Creation requests.

At Job submission time, the Printer MAY perform the validation checks reserved for Job processing time, such as:

1. Validating the format of the Document data

2. Validating the actual contents of any Client-supplied URI (resolve the reference and follow the link to the Document data)

At Job submission time, these additional Job processing time validation checks are essentially useless, since they require actually parsing and interpreting the Document data, are not guaranteed to be 100% accurate, and MUST be done, yet again, at Job processing time. Also, in the case of a URI, checking for availability at Job submission time does not guarantee availability at Job processing time. In addition, at Job processing time, the Printer might discover any of the following conditions that were not detectable at Job submission time:

- runtime errors in the Document data,

- nested Document data that is in an unsupported format,

- the URI reference is no longer valid (i.e., the server hosting the Document might be down), or
- any other Job processing error.

At Job submission time, a Printer, especially a non-spooling Printer, MAY accept Jobs for which it does not have enough space. In such a situation, a Printer MAY stop reading data from a Client for an indefinite period of time. A Client MUST be prepared for a write operation to block for an indefinite period of time (see Section 6.1 ("Client Conformance Requirements")).

When a Printer has too little space for starting a new Job, it MAY reject a new Job Creation request. In this case, a Printer MUST return a response (in reply to the rejected request) with a status-code of 'server-error-busy' (see Appendix B.1.5.8), and it MAY close the connection before receiving all bytes of the operation. A Printer SHOULD indicate that it is temporarily unable to accept Jobs by setting the 'spool-space-full' value in its "printer-state-reasons" attribute and removing the value when it can accept another Job (see Section 5.4.12).

When receiving a 'server-error-busy' status-code in an operation response, a Client MUST be prepared for the Printer to close the connection before the Client has sent all of the data (especially for the Print-Job operation). A Client MUST be prepared to keep submitting a Job Creation request until the Printer accepts the Job Creation request.

At Job processing time, since the Printer has already responded with a successful status-code in the response to the Job Creation request, if the Printer detects an error, the Printer is unable to inform the End User of the error with an operation status-code. In this case, the Printer, depending on the error, can set the Job's "job-state", "job-state-reasons", and/or "job-state-message" attributes to the appropriate value(s) so that later queries can report the correct Job status.

Note: Asynchronous notification of events is defined in "Internet Printing Protocol (IPP): Event Notifications and Subscriptions" [RFC3995].

4.2. Printer Operations

All Printer operations are directed at Printers. A Client MUST always supply the "printer-uri" operation attribute in order to identify the correct target of the operation.

4.2.1. Print-Job Operation

This REQUIRED operation allows a Client to submit a Print Job with only one Document and supply the Document data (rather than just a reference to the data). See Appendix C for the suggested steps for processing Job Creation requests and their operation and Job Template attributes.

4.2.1.1. Print-Job Request

The following groups of attributes are supplied as part of the Print-Job request:

Group 1: Operation Attributes

Natural Language and Character Set:

The "attributes-charset" and "attributes-natural-language" attributes as described in Section 4.1.4.1. The Printer MUST copy these values to the corresponding Job Status attributes described in Sections 5.3.19 and 5.3.20.

Target:

The "printer-uri" (uri) operation attribute, which is the target for this operation as described in Section 4.1.5.

Requesting User Name:

The "requesting-user-name" (name(MAX)) attribute SHOULD be supplied by the Client as described in Section 9.3.

"job-name" (name(MAX)):

The Client MAY supply and the Printer MUST support this attribute. It contains the Client-supplied Job name. If this attribute is supplied by the Client, its value is used for the "job-name" attribute of the newly created Job. The Client MAY automatically include any information that will help the End User distinguish amongst his/her Jobs, such as the name of the application program along with information from the Document, such as the Document name, Document subject, or source file name. If this attribute is not supplied by the Client, the Printer generates a name to use in the "job-name" attribute of the newly created Job (see Section 5.3.5).

"ipp-attribute-fidelity" (boolean):

The Client MAY supply and the Printer MUST support this attribute. The value 'true' indicates that total fidelity to Client-supplied Job Template attributes and values is required; otherwise, the Printer MUST reject the Print-Job request. The value 'false' indicates that a reasonable attempt to print the Job is acceptable and the Printer MUST accept the Print-Job request. If not supplied, the Printer assumes that the value is 'false'. All Printers MUST support both types of Job processing. See Appendix C for a full description of "ipp-attribute-fidelity" and its relationship to other attributes, especially the Printer's "pdl-override-supported" attribute.

"document-name" (name(MAX)):

The Client MAY supply and the Printer MUST support this attribute. It contains the Client-supplied Document name. The Document name MAY be different than the Job name. Typically, the Client software automatically supplies the Document name on behalf of the End User by using a file name or an application-generated name. If this attribute is supplied, its value can be used in a manner defined by each implementation. Examples include the following: printed along with the Job (Job start sheet, page adornments, etc.), used by accounting or resource-tracking management tools, or even stored along with the Document as a Document-level attribute.

"compression" (type2 keyword):

The Client MAY supply and the Printer MUST support this attribute. The Client-supplied "compression" operation attribute identifies the compression algorithm used on the Document data. The following cases exist:

a. If the Client omits this attribute, the Printer MUST assume that the data is not compressed, i.e., the Printer follows the rules below as if the Client supplied the "compression" attribute with a value of 'none'.

b. If the Client supplies this attribute but the value is not supported by the Printer, i.e., the value is not one of the values of the Printer's "compression-supported" attribute, the Printer MUST reject the request and return the 'client-error-compression-not-supported' status-code. See Section 4.1.7 for details on returning unsupported attributes and values.

c. If the Client supplies the attribute and the Printer supports the attribute value, the Printer uses the corresponding decompression algorithm on the Document data.

d. If the decompression algorithm fails before the Printer returns an operation response, the Printer MUST reject the request and return the 'client-error-compression-error' status-code.

e. If the decompression algorithm fails after the Printer returns an operation response, the Printer MUST abort the Job and add the 'compression-error' value to the Job's "job-state-reasons" attribute.

f. If the decompression algorithm succeeds, the Document data MUST then have the format specified by the Job's "document-format" attribute, if supplied (see the "document-format" operation attribute definition below).

"document-format" (mimeMediaType):

The Client MAY supply and the Printer MUST support this attribute. The value identifies the format of the supplied Document data. The following cases exist:

a. If the Client does not supply this attribute, the Printer assumes that the Document data is in the format defined by the Printer's "document-format-default" attribute (i.e., the Printer follows the rules below as if the Client supplied the "document-format" attribute with a value equal to the Printer's default value).

b. If the Client supplies this attribute but the value is not supported by the Printer, i.e., the value is not one of the values of the Printer's "document-format-supported" attribute, the Printer MUST reject the request and return the 'client-error-document-format-not-supported' status-code.

c. If the Client supplies this attribute and its value is 'application/octet-stream' (i.e., to be auto-sensed; see Section 5.1.10.1), and the format is not one of the Document formats that the Printer can auto-sense, and this check occurs before the Printer returns an operation response, then the Printer MUST reject the request and return the 'client-error-document-format-not-supported' status-code.

d. If the Client supplies this attribute and the value is supported by the Printer, the Printer is capable of interpreting the Document data.

e. If interpretation of the Document data fails before the Printer returns an operation response, the Printer MUST reject the request and return the 'client-error-document-format-error' status-code.

f. If interpretation of the Document data fails after the Printer returns an operation response, the Printer MUST abort the Job and add the 'document-format-error' value to the Job's "job-state-reasons" attribute.

"document-natural-language" (naturalLanguage):

The Client MAY supply and the Printer SHOULD support this attribute. The value specifies the natural language of the Document content for those Document formats that require a specification of the natural language in order to properly image the Document.

"job-k-octets" (integer(0:MAX)):

The Client MAY supply and the Printer SHOULD support this attribute. The Client-supplied "job-k-octets" operation attribute identifies the total size of the Document(s) in K octets being submitted (see Section 5.3.17.1 for the complete semantics). If the Client supplies the attribute and the Printer supports the attribute, the value of the attribute is used to populate the Job's "job-k-octets" Job Description attribute.

For this attribute and the following two attributes ("job-impressions" and "job-media-sheets"), if the Client supplies the attribute but the Printer does not support the attribute, the Printer ignores the Client-supplied value. If the Client supplies the attribute and the Printer supports the attribute, and the value is within the range of the corresponding Printer's "xxx-supported" attribute, the Printer MUST use the value to populate the Job's "xxx" attribute. If the Client supplies the attribute and the Printer supports the attribute, but the value is outside the range of the corresponding Printer's "xxx-supported" attribute, the Printer MUST copy the attribute and its value to the Unsupported Attributes group, reject the request, and return the 'client-error-attributes-or-values-not-supported' status-code. If the Client does not supply the attribute, the Printer SHOULD

populate the corresponding Job attribute if it supports the attribute and is able to calculate or discern the correct value.

"job-impressions" (integer(0:MAX)):

The Client MAY supply and the Printer SHOULD support this attribute. The Client-supplied "job-impressions" operation attribute identifies the total size in number of Impressions of the Document(s) being submitted (see Section 5.3.17.2 for the complete semantics).

See the last paragraph under "job-k-octets".

"job-media-sheets" (integer(1:MAX)):

The Client MAY supply and the Printer SHOULD support this attribute. The Client-supplied "job-media-sheets" operation attribute identifies the total number of Media Sheets to be produced for this Job (see Section 5.3.17.3 for the complete semantics).

See the last paragraph under "job-k-octets".

Group 2: Job Template Attributes

The Client MAY supply a set of Job Template attributes as defined in Section 5.2. If the Client is not supplying any Job Template attributes in the request, the Client SHOULD omit Group 2 rather than sending an empty group. However, a Printer MUST be able to accept an empty group.

Group 3: Document Data

The Client MUST supply the Document data to be processed.

The simplest Print-Job request consists of just the "attributes-charset" and "attributes-natural-language" operation attributes, the "printer-uri" target operation attribute, and the Document data. In this simple case, the Printer:

- creates a new Job containing a single Document,
- stores a generated Job name in the "job-name" attribute in the natural language and charset requested (see Section 4.1.4.1) (if those are supported; otherwise, using the Printer’s default natural language and charset), and

- at Job processing time, uses its corresponding default value attributes for the supported Job Template attributes that were not supplied by the Client as an IPP attribute or embedded instructions in the Document data.

4.2.1.2. Print-Job Response

The Printer MUST return to the Client the following sets of attributes as part of the Print-Job response:

Group 1: Operation Attributes

Natural Language and Character Set:

The "attributes-charset" and "attributes-natural-language" attributes as described in Section 4.1.4.2.

Status Message:

In addition to the REQUIRED status-code returned in every response, the response MAY include a "status-message" (text(255)) and/or a "detailed-status-message" (text(MAX)) operation attribute as described in Appendix B and Section 4.1.6. If the Client supplies unsupported or conflicting Job Template attributes or values, the Printer MUST reject or accept the Print-Job request, depending on whether the Client supplied a 'true' or 'false' value for the "ipp-attribute-fidelity" operation attribute. See the Implementor's Guides [RFC3196] [PWG5100.19] for guidance on processing Job Creation requests.

Group 2: Unsupported Attributes

See Section 4.1.7 for details on returning unsupported attributes.

The value of "ipp-attribute-fidelity" supplied by the Client does not affect what attributes the Printer returns in this group. The value of "ipp-attribute-fidelity" only affects whether the Print-Job operation is accepted or rejected. If the Job is accepted, the Client can query the Job using the Get-Job-Attributes operation, requesting the unsupported attributes that were returned in the Print-Job response to see which attributes were ignored (not stored in the Job) and which attributes were stored with other (substituted) values.

Group 3: Job Attributes

"job-id" (integer(1:MAX)):

The Printer MUST return the Job's ID in the REQUIRED "job-id" Job attribute. The Client uses this "job-id" attribute in conjunction with the "printer-uri" attribute used in the Print-Job request when directing Job operations at the Printer.

"job-uri" (uri):

The Printer MUST return the Job's URI by returning the contents of the REQUIRED "job-uri" Job attribute.

"job-state" (type1 enum):

The Printer MUST return the Job's REQUIRED "job-state" attribute. The value of this attribute along with the value of the "job-state-reasons" attribute is a "snapshot" of the new Job's state when the Printer returns the response.

"job-state-reasons" (1setOf type2 keyword):

The Printer MUST return the Job's REQUIRED "job-state-reasons" attribute.

"job-state-message" (text(MAX)):

The Printer SHOULD return the Job's RECOMMENDED "job-state-message" attribute. If the Printer supports this attribute, then it MUST be returned in the response. If this attribute is not returned in the response, the Client can assume that the "job-state-message" attribute is not supported and will not be returned in a subsequent Job query.

"number-of-intervening-jobs" (integer(0:MAX)):

The Printer SHOULD return the Job's RECOMMENDED "number-of-intervening-jobs" attribute. If the Printer supports this attribute, then it MUST be returned in the response. If this attribute is not returned in the response, the Client can assume that the "number-of-intervening-jobs" attribute is not supported and will not be returned in a subsequent Job query.

Note: Since any Printer state information that affects a Job's state is reflected in the "job-state" and "job-state-reasons" attributes, it is sufficient to return only these attributes and no additional Printer Status attributes.

Note: The simplest response consists of just the "attributes-charset" and "attributes-natural-language" operation attributes and the "job-uri", "job-id", and "job-state" Job attributes. In this simplest case, the status-code is 'successful-ok' and there is no "status-message" or "detailed-status-message" operation attribute.

4.2.2. Print-URI Operation

This OPTIONAL operation is identical to the Print-Job operation (Section 4.2.1), except that a Client supplies a URI reference to the Document data using the "document-uri" (uri) operation attribute (in Group 1) rather than including the Document data itself. Before returning the response, the Printer MUST validate that the Printer supports the retrieval method (e.g., 'http', 'ftp', etc.) implied by the URI and MUST check for valid URI syntax. If the Client-supplied URI scheme is not supported, i.e., the value is not in the Printer's "referenced-uri-scheme-supported" attribute, the Printer MUST reject the request and return the 'client-error-uri-scheme-not-supported' status-code.

The Printer MAY validate the accessibility of the Document as part of the operation, or subsequently. If the Printer discovers an accessibility problem before returning an operation response, it MUST reject the request and return the 'client-error-document-access-error' status-code. The Printer MAY also return a specific Document access error code using the "document-access-error" operation attribute (see Section 4.1.6.4).

If the Printer discovers this Document accessibility problem after accepting the request and returning an operation response with one of the successful status-code values, the Printer MUST add the "document-access-error" value to the Job's "job-state-reasons" attribute and MAY populate the Job's "job-document-access-errors" Job Status attribute (see Section 5.3.11). See the Implementor's Guides [RFC3196] [PWG5100.19] for guidance on processing Job Creation requests.

If the Printer supports this operation, it MUST support the "reference-uri-schemes-supported" Printer attribute (see Section 5.4.27).

It is up to the Printer to interpret the URI and subsequently "pull" the Document data from the source referenced by the URI string.

4.2.3. Validate-Job Operation

This REQUIRED operation is similar to the Print-Job operation (Section 4.2.1), except that a Client supplies no Document data and the Printer allocates no resources, i.e., it does not create a new Job. This operation is used only to verify the capabilities of a Printer against whatever attributes are supplied by the Client in the Validate-Job request. By using the Validate-Job operation, a Client can validate that an identical Job Creation request (with the Document data) would be accepted. The Validate-Job operation also performs the same security negotiation as the Print-Job, Print-URI, and Create-Job operations (see Section 9) so that a Client can check that the Client and Printer security requirements can be met before performing a Job Creation request.

The Validate-Job operation does not accept a "document-uri" attribute in order to allow a Client to check that the same Print-URI operation will be accepted, since the Client doesn't send the data with the Print-URI operation. The Client SHOULD just issue the Print-URI request.

The Printer returns the same status-code values, Operation Attributes (Group 1), and Unsupported Attributes (Group 2) as the Print-Job operation. However, no Job Attributes (Group 3) are returned, since no Job is created.

4.2.4. Create-Job Operation

This RECOMMENDED operation is similar to the Print-Job operation (Section 4.2.1), except that in the Create-Job request, a Client does not supply Document data or any reference to Document data. Also, the Client does not supply any of the "document-name", "document-format", "compression", or "document-natural-language" operation attributes. This operation is followed by one or more Send-Document or Send-URI operations. In each of those operation requests, the Client MAY supply the "document-name", "document-format", and "document-natural-language" attributes for each Document in the Job.

If a Printer supports the Create-Job operation, it MUST also support the Send-Document operation. If the Printer supports the Create-Job and Print-URI operations, it MUST also support the Send-URI operation.

If the Printer supports this operation, it MUST support the "multiple-operation-time-out" Printer attribute (see Section 5.4.31).

If the Printer supports this operation, then it MUST support the "multiple-document-jobs-supported" Printer Description attribute (see Section 5.4.16) and indicate whether it supports multiple Documents in a Job.

If the Printer supports this operation and supports multiple Documents in a Job, then it MUST support the "multiple-document-handling" Job Template attribute with at least one value (see Section 5.2.4), and the associated "multiple-document-handling-default" and "multiple-document-handling-supported" Printer attributes (see Section 5.2).

After the Create-Job operation has completed, the value of the "job-state" attribute is similar to the "job-state" after a Print-Job operation, even though no Document data has arrived. A Printer MAY set the 'job-data-insufficient' value of the Job's "job-state-reasons" attribute to indicate that processing cannot begin until sufficient data has arrived and set the "job-state" to either 'pending' or 'pending-held'. A non-spooling Printer that doesn't implement the 'pending' Job state can set "job-state" to 'processing', even though there is not yet any data to process. See Sections 5.3.7 and 5.3.8.

4.2.5. Get-Printer-Attributes Operation

This REQUIRED operation allows a Client to request the values of the attributes of a Printer. In the request, the Client supplies the set of Printer attribute names and/or attribute group names in which the requester is interested. In the response, the Printer returns a corresponding attribute set with the appropriate attribute values filled in.

For Printers, the possible names of attribute groups are:

- 'job-template': the subset of the Job Template attributes that apply to a Printer (the last two columns of Table 8 in Section 5.2) that the implementation supports for Printers.
- 'printer-description': the subset of the attributes specified in Section 5.4 that the implementation supports for Printers.
- 'all': the special group 'all' that includes all attributes that the implementation supports for Printers.

Since a Client MAY request specific attributes or named groups, there is a potential for some overlap. For example, if a Client requests 'printer-name' and 'all', the Client is actually requesting the

"printer-name" attribute twice: once by naming it explicitly, and once by inclusion in the 'all' group. In such cases, the Printer returns each attribute only once in the response even if it is requested multiple times. The Client SHOULD NOT request the same attribute in multiple ways.

Printers MUST support all group names and MUST return all supported attributes belonging to the group.

4.2.5.1. Get-Printer-Attributes Request

The following sets of attributes are part of the Get-Printer-Attributes request:

Group 1: Operation Attributes

Natural Language and Character Set:

The "attributes-charset" and "attributes-natural-language" attributes as described in Section 4.1.4.1.

Target:

The "printer-uri" (uri) operation attribute, which is the target for this operation as described in Section 4.1.5.

Requesting User Name:

The "requesting-user-name" (name(MAX)) attribute SHOULD be supplied by the Client as described in Section 9.3.

"requested-attributes" (1setOf keyword):

The Client MAY supply a set of attribute names and/or attribute group names in whose values the requester is interested. The Printer MUST support this attribute. If the Client omits this attribute, the Printer MUST respond as if this attribute had been supplied with a value of 'all'.

"document-format" (mimeMediaType):

The Client MAY supply and the Printer MUST support this attribute. It is useful for a Client to determine the set of supported attribute values that relate to the requested Document format. The Printer MUST return the attributes and values that it uses to validate a Job in a Job Creation or Validate-Job operation in which this Document format is supplied. The Printer SHOULD return only (1) those attributes

that are supported for the specified format and (2) the attribute values that are supported for the specified Document format. By specifying the Document format, the Client can get the Printer to eliminate the attributes and values that are not supported for a specific Document format. For example, a Printer might have multiple interpreters to support both 'application/postscript' (for PostScript) and 'text/plain' (for text) Documents. However, only one of those interpreters might support the "number-up" Job Template attribute with values of '1', '2', and '4'. The other interpreter might only be able to support the "number-up" Job Template attribute with a value of '1'. Thus, a Client can use the Get-Printer-Attributes operation to obtain the attributes and values that will be used to accept/reject a Job Creation request.

If the Printer does not distinguish between different sets of supported values for each different Document format when validating Jobs in the Create-Job, Print-Job, Print-URI, and Validate-Job operations, it MUST NOT distinguish between different Document formats in the Get-Printer-Attributes operation. If the Printer does distinguish between different sets of supported values for each different Document format specified by the Client, this specialization applies only to the following Printer attributes:

+ Printer attributes that are Job Template attributes ("xxx-default", "xxx-supported", and "xxx-ready") (see Table 8 in Section 5.2),

+ "pdl-override-supported",

+ "compression-supported",

+ "job-k-octets-supported",

+ "job-impressions-supported,

+ "job-media-sheets-supported",

+ "printer-driver-installer",

+ "color-supported", and

+ "reference-uri-schemes-supported"

The values of all other Printer attributes (including "document-format-supported") remain invariant with respect to the Client-supplied Document format (except for new Printer Description attributes as registered according to Section 7.2).

If the Client omits this "document-format" operation attribute, the Printer MUST respond as if the attribute had been supplied with the value of the Printer's "document-format-default" attribute. Clients SHOULD always supply a value for "document-format", since the Printer's "document-format-default" value can be 'application/octet-stream', in which case the returned attributes and values are for the union of the Document formats that the Printer can automatically sense. For more details, see the description of the 'mimeMediaType' attribute syntax in Section 5.1.10.

If the Client supplies a value for the "document-format" operation attribute that is not supported by the Printer, i.e., is not among the values of the Printer's "document-format-supported" attribute, the Printer MUST reject the operation and return the 'client-error-document-format-not-supported' status-code.

4.2.5.2. Get-Printer-Attributes Response

The Printer returns the following sets of attributes as part of the Get-Printer-Attributes response:

Group 1: Operation Attributes

Natural Language and Character Set:

The "attributes-charset" and "attributes-natural-language" attributes as described in Section 4.1.4.2.

Status Message:

In addition to the REQUIRED status-code returned in every response, the response MAY include a "status-message" (text(255)) and/or a "detailed-status-message" (text(MAX)) operation attribute as described in Appendix B and Section 4.1.6.

Group 2: Unsupported Attributes

See Section 4.1.7 for details on returning unsupported attributes.

The response MAY contain the "requested-attributes" operation attribute with any supplied values (attribute keywords) that were requested by the Client but are not supported by the Printer. If the Printer does return unsupported attributes referenced in the "requested-attributes" operation attribute and that attribute included group names, such as 'all', the unsupported attributes MUST NOT include attributes described in this document but not supported by the implementation.

Group 3: Printer Attributes

This is the set of requested attributes and their current values. The Printer ignores (does not respond with) any requested attribute that is not supported. The Printer MAY respond with a subset of the supported attributes and values, depending on the security policy in force. However, the Printer MUST respond with the 'unknown' value for any supported attribute (including all REQUIRED attributes) for which the Printer does not know the value. Also, the Printer MUST respond with 'no-value' for any supported attribute (including all REQUIRED attributes) for which the Administrator has not configured a value. See the description of the "out-of-band" values in the beginning of Section 5.1.

4.2.6. Get-Jobs Operation

This REQUIRED operation allows a Client to retrieve the list of Jobs belonging to the target Printer. The Client can also supply a list of Job attribute names and/or attribute group names. A group of Job attributes will be returned for each Job that is returned.

This operation is similar to the Get-Job-Attributes operation, except that this Get-Jobs operation returns attributes from possibly more than one Job.

4.2.6.1. Get-Jobs Request

The Client submits the Get-Jobs request to a Printer.

The following groups of attributes are part of the Get-Jobs request:

Group 1: Operation Attributes

Natural Language and Character Set:

The "attributes-charset" and "attributes-natural-language" attributes as described in Section 4.1.4.1.

Target:

The "printer-uri" (uri) operation attribute, which is the target for this operation as described in Section 4.1.5.

Requesting User Name:

The "requesting-user-name" (name(MAX)) attribute SHOULD be supplied by the Client as described in Section 9.3.

"limit" (integer(1:MAX)):

The Client MAY supply and the Printer MUST support this attribute. It is an integer value that determines the maximum number of Jobs that a Client will receive from the Printer even if "which-jobs" or "my-jobs" (described below) constrain which Jobs are returned. The limit is a "stateless limit" in that if the value supplied by the Client is 'N', then only the first 'N' Jobs are returned in the Get-Jobs response. If the Client does not supply this attribute, the Printer responds with all applicable Jobs.

"requested-attributes" (1setOf type2 keyword):

The Client MAY supply and the Printer MUST support this attribute. It is a set of Job attribute names and/or attribute group names in whose values the requester is interested. This set of attributes is returned for each Job that is returned. The allowed attribute group names are the same as those defined in the Get-Job-Attributes operation in Section 4.3.4. If the Client does not supply this attribute, the Printer MUST respond as if the Client had supplied this attribute with two values: "job-uri" and "job-id".

"which-jobs" (type2 keyword):

The Client MAY supply and the Printer MUST support this attribute. It indicates which Jobs MUST be returned by the Printer. The values for this attribute include:

+ 'completed': Any Job whose state is 'completed', 'canceled', or 'aborted'.

+ 'not-completed': Any Job whose state is 'pending', 'processing', 'processing-stopped', or 'pending-held'.

A Printer MUST support both values. However, if the implementation does not keep Jobs in the 'completed', 'canceled', and 'aborted' states, then it returns no Jobs when the 'completed' value is supplied.

If a Client supplies some other value that is not supported by the Printer, the Printer MUST copy the attribute and the unsupported value to the Unsupported Attributes group, reject the request, and return the 'client-error-attributes-or-values-not-supported' status-code.

If the Client does not supply this attribute, the Printer MUST respond as if the Client had supplied the attribute with a value of 'not-completed'.

"my-jobs" (boolean):

The Client MAY supply and the Printer MUST support this attribute. It indicates whether Jobs from all users or just the Jobs submitted by the requesting user of this request MUST be considered as candidate Jobs to be returned by the Printer. If the Client does not supply this attribute, the Printer MUST respond as if the Client had supplied the attribute with a value of 'false', i.e., Jobs from all users. The means for authenticating the requesting user and matching the Jobs is described in Section 9.

4.2.6.2. Get-Jobs Response

The Printer returns all of the Jobs up to the number specified by the "limit" attribute that match the criteria as defined by the attribute values supplied by the Client in the request. It is possible that no Jobs are returned, since there can literally be no Jobs at the Printer or there can be no Jobs that match the criteria supplied by the Client. If the Client requests any Job attributes at all, there is a set of Job Attributes returned for each Job.

It is not an error for the Printer to return 0 Jobs. If the response returns 0 Jobs because there are no Jobs matching the criteria, and the request would have returned one or more Jobs with a status-code of 'successful-ok' if there had been Jobs matching the criteria, then the status-code for 0 Jobs MUST be 'successful-ok'.

Group 1: Operation Attributes

Natural Language and Character Set:

The "attributes-charset" and "attributes-natural-language" attributes as described in Section 4.1.4.2.

Status Message:

In addition to the REQUIRED status-code returned in every response, the response MAY include a "status-message" (text(255)) and/or a "detailed-status-message" (text(MAX)) operation attribute as described in Appendix B and Section 4.1.6.

Group 2: Unsupported Attributes

See Section 4.1.7 for details on returning unsupported attributes.

The response MAY contain the "requested-attributes" operation attribute with any supplied values (attribute keywords) that were requested by the Client but are not supported by the Printer. If the Printer does return unsupported attributes referenced in the "requested-attributes" operation attribute and that attribute included group names, such as 'all', the unsupported attributes MUST NOT include attributes described in this document but not supported by the implementation.

Groups 3 to N: Job Attributes

The Printer responds with one set of Job Attributes for each returned Job. The Printer ignores (does not respond with) any requested attribute or value that is not supported or that is restricted by the security policy in force, including whether the requesting user is the user that submitted the Job (Job-originating user) or not (see Section 9). However, the Printer MUST respond with the 'unknown' value for any supported attribute (including all REQUIRED attributes) for which the Printer does not know the value, unless it would violate the security policy. See the description of the "out-of-band" values in the beginning of Section 5.1.

Jobs are returned in the following order:

* If the Client requests all 'completed' Jobs (Jobs in the 'completed', 'aborted', or 'canceled' states), then the Jobs are returned newest to oldest (with respect to actual completion time).

* If the Client requests all 'not-completed' Jobs (Jobs in the 'pending', 'processing', 'pending-held', and 'processing-stopped' states), then Jobs are returned in relative chronological order of expected time to complete (based on whatever scheduling algorithm is configured for the Printer).

4.2.7. Pause-Printer Operation

This OPTIONAL operation allows a Client to stop the Printer from scheduling Jobs on all its devices. Depending on implementation, the Pause-Printer operation MAY also stop the Printer from processing the current Job or Jobs. Any Job that is currently being printed is either (1) stopped as soon as the implementation permits or (2) completed, depending on implementation. The Printer MUST still accept Job Creation requests to create new Jobs but MUST prevent any Jobs from entering the 'processing' state.

If the Pause-Printer operation is supported, then the Resume-Printer operation MUST be supported, and vice versa.

The IPP Printer stops the current Job(s) on its device or devices that were in the 'processing' or 'processing-stopped' state as soon as the implementation permits. If the implementation will take appreciable time to stop, the IPP Printer adds the 'moving-to-paused' value to the Printer's "printer-state-reasons" attribute (see Section 5.4.12). When the device or devices have all stopped, the IPP Printer transitions the Printer to the 'stopped' state; removes the 'moving-to-paused' value, if present; and adds the 'paused' value to the Printer's "printer-state-reasons" attribute.

When the current Job or Jobs complete that were in the 'processing' state, the IPP Printer transitions them to the 'completed' state. When the current Job or Jobs stop in mid-processing that were in the 'processing' state, the IPP Printer transitions them to the 'processing-stopped' state and adds the 'printer-stopped' value to the Jobs' "job-state-reasons" attribute.

For any Jobs that are 'pending' or 'pending-held', the 'printer-stopped' value of the Jobs' "job-state-reasons" attribute also applies. However, the IPP Printer MAY update those Jobs' "job-state-reasons" values when those Jobs are queried (so-called "lazy evaluation").

The IPP Printer MUST accept the request in any state and transition the Printer to the indicated new "printer-state" before returning, as shown in Table 2.

Access Rights: The authenticated user (see Section 9.3) performing this operation MUST be an Operator or Administrator of the Printer (see Sections 1 and 9.5). Otherwise, the IPP Printer MUST reject the operation and return 'client-error-forbidden', 'client-error-not-authenticated', or 'client-error-not-authorized' as appropriate.

Current "printer-state"	New "printer-state"	"printer-state-reasons"	IPP Printer's response status-code and action:
'idle'	'stopped'	'paused'	'successful-ok'
'processing'	'processing'	'moving-to-paused'	Option 1: 'successful-ok'; Later, when all output has stopped, the "printer-state" becomes 'stopped', and the 'paused' value replaces the 'moving-to-paused' value in the "printer-state-reasons" attribute
'processing'	'stopped'	'paused'	Option 2: 'successful-ok'; all device output stopped immediately
'stopped'	'stopped'	'paused'	'successful-ok'

Table 2: Pause-Printer State Transitions

4.2.7.1. Pause-Printer Request

The following groups of attributes are part of the Pause-Printer request:

Group 1: Operation Attributes

Natural Language and Character Set:

The "attributes-charset" and "attributes-natural-language" attributes as described in Section 4.1.4.1.

Target:

The "printer-uri" (uri) operation attribute, which is the target for this operation as described in Section 4.1.5.

Requesting User Name:

The "requesting-user-name" (name(MAX)) attribute SHOULD be supplied by the Client as described in Section 9.3.

4.2.7.2. Pause-Printer Response

The following groups of attributes are part of the Pause-Printer response:

Group 1: Operation Attributes

Natural Language and Character Set:

The "attributes-charset" and "attributes-natural-language" attributes as described in Section 4.1.4.2.

Status Message:

In addition to the REQUIRED status-code returned in every response, the response MAY include a "status-message" (text(255)) and/or a "detailed-status-message" (text(MAX)) operation attribute as described in Appendix B and Section 4.1.6.

Group 2: Unsupported Attributes

See Section 4.1.7 for details on returning unsupported attributes.

4.2.8. Resume-Printer Operation

This OPTIONAL operation allows a Client to resume the Printer scheduling Jobs on all its devices. The Printer MUST remove the 'paused' and 'moving-to-paused' values from the Printer's "printer-state-reasons" attribute, if present. If there are no other reasons to keep a device paused (such as a media jam), the IPP Printer is free to transition itself to the 'processing' or 'idle' state, depending on whether there are Jobs to be processed or not, respectively, and the device(s) resumes processing Jobs.

If the Pause-Printer operation is supported, then the Resume-Printer operation MUST be supported, and vice versa.

The IPP Printer removes the 'printer-stopped' value from any Job's "job-state-reasons" attributes contained in that Printer.

The IPP Printer MUST accept the request in any state and transition the Printer to the indicated new state as shown in Table 3.

Access Rights: The authenticated user (see Section 9.3) performing this operation MUST be an Operator or Administrator of the Printer (see Sections 1 and 9.5). Otherwise, the IPP Printer MUST reject the operation and return 'client-error-forbidden', 'client-error-not-authenticated', or 'client-error-not-authorized' as appropriate.

The Resume-Printer request and Resume-Printer response have the same attribute groups and attributes as the Pause-Printer operation (see Sections 4.2.7.1 and 4.2.7.2).

Current "printer-state"	New "printer-state"	IPP Printer's response status-code and action:
'idle'	'idle'	'successful-ok'
'processing'	'processing'	'successful-ok'
'stopped'	'processing'	'successful-ok', when there are Jobs to be processed
'stopped'	'idle'	'successful-ok', when there are no Jobs to be processed

Table 3: Resume-Printer State Transitions

4.2.9. Purge-Jobs Operation

This DEPRECATED operation allows a Client to remove all Jobs from a Printer, regardless of their Job states, including Jobs in the Printer's Job History (see Section 5.3.7.2). After a Purge-Jobs operation has been performed, a Printer MUST return no Jobs in subsequent Get-Job-Attributes and Get-Jobs responses (until new Jobs are submitted).

Note: This operation SHOULD NOT be supported in new implementations, since it destroys Printer accounting information.

Whether the Purge-Jobs (and Get-Jobs) operation affects Jobs that were submitted to the device from sources other than the IPP Printer in the same way that the Purge-Jobs operation affects Jobs that were submitted to the IPP Printer using IPP depends on implementation, i.e., on whether IPP is being used as a universal management protocol or just to manage IPP Jobs, respectively.

Note: If an Operator wants to cancel all Jobs without clearing out the Job History, the Operator uses the Cancel-Job operation on each Job instead of using the Purge-Jobs operation.

If this OPTIONAL operation is supported, the Printer MUST accept this operation in any state and transition the Printer to the 'idle' state.

Access Rights: The authenticated user (see Section 9.3) performing this operation MUST be an Operator or Administrator of the Printer (see Sections 1 and 9.5). Otherwise, the Printer MUST reject the operation and return 'client-error-forbidden', 'client-error-not-authenticated', and 'client-error-not-authorized' as appropriate.

The Purge-Jobs request and Purge-Jobs response have the same attribute groups and attributes as the Pause-Printer operation (see Sections 4.2.7.1 and 4.2.7.2).

4.3. Job Operations

All Job operations are directed at Jobs. A Client MUST always supply some means of identifying the Job object in order to identify the correct target of the operation. That Job identification SHOULD be the combination of a Printer URI with a Job ID but MAY be the (single) Job URI. The IPP implementation MUST support both forms of identification for every Job.

4.3.1. Send-Document Operation

This RECOMMENDED operation allows a Client to add a Document to a Job that was created using the Create-Job operation. In the Create-Job response, the Printer returns the Job's URI (the "job-uri" attribute) and the Job's 32-bit identifier (the "job-id" attribute). For each new Document that the Client desires to add, the Client uses a Send-Document operation. Each Send-Document request contains the entire stream of Document data for one Document.

If the Printer supports this operation but does not support multiple Documents per Job, the Printer MUST reject subsequent Send-Document operations supplied with data and return the 'server-error-multiple-document-jobs-not-supported' status-code. However, the Printer MUST accept the first Document with a 'true' or 'false' value for the "last-document" operation attribute (see below), so that Clients MAY always submit one Document Job with a 'false' value for "last-document" in the first Send-Document and a 'true' value for "last-document" in the second Send-Document (with no data).

Since the Create-Job and the send operations (Send-Document or Send-URI operations) that follow could occur over an arbitrarily long period of time for a particular Job, a Client MUST send another send operation within a minimum time interval, as defined by the IPP Printer, after the receipt of the previous request for the Job. If a Printer supports the Create-Job and Send-Document operations, the Printer MUST support the "multiple-operation-time-out" attribute (see Section 5.4.31). This attribute indicates the minimum number of seconds the Printer will wait for the next send operation before taking some recovery action.

A Printer MUST recover from an errant Client that does not supply a send operation, sometime after the minimum time interval specified by the Printer's "multiple-operation-time-out" attribute. Such recovery MAY include any of the following actions or other recovery actions:

1. Assume that the Job is an invalid Job, start the process of changing the Job state to 'aborted', add the 'aborted-by-system' value to the Job's "job-state-reasons" attribute (see Section 5.3.8), and clean up all resources associated with the Job. In this case, if another send operation is finally received, the Printer responds with a 'client-error-not-possible' or 'client-error-not-found' status-code, depending on whether the Job is still around when the send operation finally arrives.

2. Assume that the last send operation received was in fact the last Document (as if the "last-document" flag had been set to 'true'), close the Job, and proceed to process it (i.e., move the Job's state to 'pending').

3. Assume that the last send operation received was in fact the last Document and close the Job, but move it to the 'pending-held' state and add the 'submission-interrupted' value to the Job's "job-state-reasons" attribute (see Section 5.3.8). This action allows the user or an Operator to determine whether to continue processing the Job by moving it back to the 'pending' state using the Release-Job operation (see Section 4.3.6) or to cancel the Job using the Cancel-Job operation (see Section 4.3.3).

Each implementation is free to decide the "best" action to take, depending on the following: local policy, whether any Documents have been added, whether the implementation spools Jobs or not, and/or any other piece of information available to it. If the choice is to abort the Job, it is possible that the Job has already been processed to the point that some Media Sheet pages have been printed.

Access Rights: The authenticated user (see Section 9.3) performing this operation must be either the Job owner (as determined in the Create-Job operation) or an Operator or Administrator of the Printer (see Sections 1 and 9.5). Otherwise, the Printer MUST reject the operation and return 'client-error-forbidden', 'client-error-not-authenticated', or 'client-error-not-authorized' as appropriate.

4.3.1.1. Send-Document Request

The following attribute sets are part of the Send-Document request:

Group 1: Operation Attributes

Natural Language and Character Set:

The "attributes-charset" and "attributes-natural-language" attributes as described in Section 4.1.4.1.

Target:

Either the "printer-uri" (uri) plus "job-id" (integer(1:MAX)), or the "job-uri" (uri) operation attribute(s), which define the target for this operation as described in Section 4.1.5.

Requesting User Name:

The "requesting-user-name" (name(MAX)) attribute SHOULD be supplied by the Client as described in Section 9.3.

"document-name" (name(MAX)):

The Client MAY supply and the Printer MUST support this attribute. It contains the Client-supplied Document name. The Document name MAY be different than the Job name and is not guaranteed to be unique across multiple Documents in the same Job. Typically, the Client software automatically supplies the Document name on behalf of the End User by using a file name or an application-generated name. See the description of the "document-name" operation attribute in the Print-Job request (Section 4.2.1.1) for more information about this attribute.

"compression" (type2 keyword):

See the description of "compression" for the Print-Job operation in Section 4.2.1.1.

"document-format" (mimeMediaType):

See the description of "document-format" for the Print-Job operation in Section 4.2.1.1.

"document-natural-language" (naturalLanguage):

The Client MAY supply and the Printer MAY support this attribute. It specifies the natural language of the Document content for those Document formats that require a specification of the natural language in order to properly image the Document.

"last-document" (boolean):

The Client MUST supply and the Printer MUST support this attribute. It is a boolean flag that is set to 'true' if this is the last Document for the Job; otherwise, it is set to 'false'.

Group 2: Document Data

The Client MUST supply the Document data if the "last-document" flag is set to 'false'. However, since a Client might not know that the previous Document sent with a Send-Document (or Send-URI) operation was the last Document (i.e., the "last-document" attribute was set to 'false'), it is legal to send a Send-Document request with no Document data where the "last-document" flag is set to 'true'. Such a request MUST NOT increment the value of the Job's "number-of-documents" attribute, since no real Document was added to the Job. It is not an error for a Client to submit a Job with no actual Document data, i.e., only a single Create-Job and Send-Document request with a "last-document" operation attribute set to 'true' with no Document data.

4.3.1.2. Send-Document Response

The following sets of attributes are part of the Send-Document response:

Group 1: Operation Attributes

Natural Language and Character Set:

The "attributes-charset" and "attributes-natural-language" attributes as described in Section 4.1.4.2.

Status Message:

In addition to the REQUIRED status-code returned in every response, the response MAY include a "status-message" (text(255)) and/or a "detailed-status-message" (text(MAX)) operation attribute as described in Appendix B and Section 4.1.6.

Group 2: Unsupported Attributes

See Section 4.1.7 for details on returning unsupported attributes.

Group 3: Job Object Attributes

This is the same set of attributes as those described in the Print-Job response (see Section 4.2.1.2).

4.3.2. Send-URI Operation

This RECOMMENDED operation is identical to the Send-Document operation (see Section 4.3.1), except that a Client MUST supply a URI reference ("document-uri" operation attribute) rather than the Document data itself. If a Printer supports this operation, Clients can use both Send-URI and Send-Document operations to add new Documents to an existing Job. However, if a Client needs to indicate that the previous Send-URI or Send-Document was the last Document, the Client MUST use the Send-Document operation with no Document data and the "last-document" flag set to 'true' (rather than using a Send-URI operation with no "document-uri" operation attribute).

If a Printer supports this operation, it MUST also support the Print-URI operation (see Section 4.2.2).

The Printer MUST validate the syntax and URI scheme of the supplied URI before returning a response, just as in the Print-URI operation. The Printer MAY validate the accessibility of the Document as part of the operation, or subsequently (see Section 4.2.2).

4.3.3. Cancel-Job Operation

This REQUIRED operation allows a Client to cancel a Print Job from the time the Job is created up to the time it is completed, canceled, or aborted. Since a Job might already be printing by the time a Cancel-Job is received, some Media Sheet pages might be printed before the Job is actually terminated.

The Printer MUST accept or reject the request based on the Job's current state and transition the Job to the indicated new state as shown in Table 4.

Access Rights: The authenticated user (see Section 9.3) performing this operation must be either the Job owner or an Operator or Administrator of the Printer (see Sections 1 and 9.5). Otherwise, the Printer MUST reject the operation and return 'client-error-forbidden', 'client-error-not-authenticated', or 'client-error-not-authorized' as appropriate.

Current "job-state"	New "job-state"	Printer's response status-code and action:
'pending'	'canceled'	'successful-ok'
'pending-held'	'canceled'	'successful-ok'
'processing'	'canceled'	'successful-ok'
'processing'	'processing'	'successful-ok' (note 1)
'processing'	'processing'	'client-error-not-possible' (note 2)
'processing-stopped'	'canceled'	'successful-ok'
'processing-stopped'	'processing-stopped'	'successful-ok' (note 1)
'processing-stopped'	'processing-stopped'	'client-error-not-possible' (note 2)
'completed'	'completed'	'client-error-not-possible'
'canceled'	'canceled'	'client-error-not-possible'
'aborted'	'aborted'	'client-error-not-possible'

Table 4: Cancel-Job State Transitions

Note 1: If the implementation requires some measurable time to cancel the Job in the 'processing' or 'processing-stopped' Job state, the Printer MUST add the 'processing-to-stop-point' value to the Job's "job-state-reasons" attribute and then transition the Job to the 'canceled' state when the processing ceases (see Section 5.3.8).

Note 2: If the Job already has the 'processing-to-stop-point' value in its "job-state-reasons" attribute, then the Printer MUST reject a Cancel-Job operation.

4.3.3.1. Cancel-Job Request

The following groups of attributes are part of the Cancel-Job request:

Group 1: Operation Attributes

Natural Language and Character Set:

The "attributes-charset" and "attributes-natural-language" attributes as described in Section 4.1.4.1.

Target:

Either the "printer-uri" (uri) plus "job-id" (integer(1:MAX)), or the "job-uri" (uri) operation attribute(s), which define the target for this operation as described in Section 4.1.5.

Requesting User Name:

The "requesting-user-name" (name(MAX)) attribute SHOULD be supplied by the Client as described in Section 9.3.

"message" (text(127)):

The Client MAY supply and the Printer MAY support this attribute. It is a message to the Operator. This "message" attribute is not the same as the "job-message-from-operator" attribute. That attribute is used to report a message from the Operator to the End User that queries that attribute. This "message" operation attribute is used to send a message from the Client to the Operator along with the operation request. How or where to display this message to the Operator (if at all) is an implementation decision.

4.3.3.2. Cancel-Job Response

The following sets of attributes are part of the Cancel-Job response:

Group 1: Operation Attributes

Natural Language and Character Set:

The "attributes-charset" and "attributes-natural-language" attributes as described in Section 4.1.4.2.

Status Message:

In addition to the REQUIRED status-code returned in every response, the response MAY include a "status-message" (text(255)) and/or a "detailed-status-message" (text(MAX)) operation attribute as described in Appendix B and Section 4.1.6.

Group 2: Unsupported Attributes

See Section 4.1.7 for details on returning unsupported attributes.

Once a successful response has been sent, the implementation guarantees that the Job will eventually end up in the 'canceled' state. Between the time that the Cancel-Job operation is accepted and when the Job enters the 'canceled' job-state (see Section 5.3.7), the "job-state-reasons" attribute SHOULD contain the 'processing-to-stop-point' value, which indicates to later queries that although the Job might still be 'processing' it will eventually end up in the 'canceled' state, not the 'completed' state.

4.3.4. Get-Job-Attributes Operation

This REQUIRED operation allows a Client to request the values of attributes of a Job, and it is almost identical to the Get-Printer-Attributes operation (see Section 4.2.5). The only differences are that the operation is directed at a Job rather than a Printer, there is no "document-format" operation attribute used when querying a Job, and the returned attribute group is a set of Job attributes rather than a set of Printer attributes.

For Jobs, the possible names of attribute groups are:

- 'job-template': the subset of the Job Template attributes that apply to a Job (the first column of Table 8 in Section 5.2) that the implementation supports for Jobs.

- 'job-description': the subset of the Job Description and Status attributes specified in Section 5.3 that the implementation supports for Jobs.

- 'all': the special group 'all' that includes all attributes that the implementation supports for Jobs.

Since a Client MAY request specific attributes or named groups, there is a potential for some overlap. For example, if a Client requests 'job-name' and 'job-description', the Client is actually requesting the "job-name" attribute once by naming it explicitly, and once by inclusion in the 'job-description' group. In such cases, the Printer returns the attribute only once in the response even if it is requested multiple times. The Client SHOULD NOT request the same attribute in multiple ways.

Jobs MUST support all group names and MUST return all supported attributes belonging to the group.

4.3.4.1. Get-Job-Attributes Request

The following groups of attributes are part of the Get-Job-Attributes request when the request is directed at a Job:

Group 1: Operation Attributes

Natural Language and Character Set:

The "attributes-charset" and "attributes-natural-language" attributes as described in Section 4.1.4.1.

Target:

Either the "printer-uri" (uri) plus "job-id" (integer(1:MAX)), or the "job-uri" (uri) operation attribute(s), which define the target for this operation as described in Section 4.1.5.

Requesting User Name:

The "requesting-user-name" (name(MAX)) attribute SHOULD be supplied by the Client as described in Section 9.3.

"requested-attributes" (1setOf keyword):

The Client MAY supply and the Printer MUST support this attribute. It is a set of attribute names and/or attribute group names in whose values the requester is interested. If the Client omits this attribute, the Printer MUST respond as if this attribute had been supplied with a value of 'all'.

4.3.4.2. Get-Job-Attributes Response

The Printer returns the following sets of attributes as part of the Get-Job-Attributes response:

Group 1: Operation Attributes

Natural Language and Character Set:

The "attributes-charset" and "attributes-natural-language" attributes as described in Section 4.1.4.2. "attributes-natural-language" MAY be the natural language of the Job, rather than the one requested.

Status Message:

In addition to the REQUIRED status-code returned in every response, the response MAY include a "status-message" (text(255)) and/or a "detailed-status-message" (text(MAX)) operation attribute as described in Appendix B and Section 4.1.6.

Group 2: Unsupported Attributes

See Section 4.1.7 for details on returning unsupported attributes.

The response MAY contain the "requested-attributes" operation attribute with any supplied values (attribute keywords) that were requested by the Client but are not supported by the Printer. If the Printer does return unsupported attributes referenced in the "requested-attributes" operation attribute and that attribute included group names, such as 'all', the unsupported attributes MUST NOT include attributes described in this document but not supported by the implementation.

Group 3: Job Attributes

This is the set of requested attributes and their current values. The Printer ignores (does not respond with) any requested attribute or value that is not supported or that is restricted by the security policy in force, including whether the requesting user is the user that submitted the Job (Job-originating user) or not (see Section 9). However, the Printer MUST respond with the 'unknown' value for any supported attribute (including all REQUIRED attributes) for which the Printer does not know the value, unless it would violate the security policy. See the description of the "out-of-band" values in the beginning of Section 5.1.

4.3.5. Hold-Job Operation

This OPTIONAL operation allows a Client to hold a pending Job in the queue so that it is not eligible for scheduling. If the Hold-Job operation is supported, then the Release-Job operation MUST be supported, and vice versa. The OPTIONAL "job-hold-until" operation attribute allows a Client to specify whether to hold the Job indefinitely or until a specified time period, if supported.

The Printer MUST accept or reject the request based on the Job's current state and transition the Job to the indicated new state as shown in Table 5.

Note: In order to keep the Hold-Job operation simple, such a request is rejected when the Job is in the 'processing' or 'processing-stopped' state. If an operation is needed to hold Jobs while in either of these states, it will be added as an additional operation, rather than overloading the Hold-Job operation. Then it is clear to Clients by querying the Printer's "operations-supported" (see Section 5.4.15) and the Job's "job-state" (see Section 5.3.7) attributes which operations are possible.

Access Rights: The authenticated user (see Section 9.3) performing this operation must be either the Job owner or an Operator or Administrator of the Printer (see Sections 1 and 9.5). Otherwise, the Printer MUST reject the operation and return 'client-error-forbidden', 'client-error-not-authenticated', or 'client-error-not-authorized' as appropriate.

Current "job-state"	New "job-state"	Printer's response status-code and action:
'pending'	'pending-held'	'successful-ok' (note 1)
'pending'	'pending'	'successful-ok' (note 2)
'pending-held'	'pending-held'	'successful-ok' (note 1)
'pending-held'	'pending'	'successful-ok' (note 2)
'processing'	'processing'	'client-error-not-possible'
'processing-stopped'	'processing-stopped'	'client-error-not-possible'
'completed'	'completed'	'client-error-not-possible'
'canceled'	'canceled'	'client-error-not-possible'
'aborted'	'aborted'	'client-error-not-possible'

Table 5: Hold-Job State Transitions

Note 1: If the implementation supports multiple reasons for a Job to be in the 'pending-held' state, the Printer MUST add the "job-hold-until-specified" value to the Job's "job-state-reasons" attribute.

Note 2: If the Printer supports the "job-hold-until" operation attribute, but the specified time period has already started (or is the 'no-hold' value) and there are no other reasons to hold the Job, the Printer MUST make the Job be a candidate for processing immediately (see Section 5.2.2) by putting the Job in the 'pending' state.

4.3.5.1. Hold-Job Request

The groups and operation attributes are the same as those defined for a Cancel-Job request (see Section 4.3.3.1), with the addition of the following Group 1 operation attribute:

"job-hold-until" (type2 keyword | name(MAX)):

The Client MAY supply and the Printer MUST support this operation attribute in a Hold-Job request if it supports the "job-hold-until" Job Template attribute in Job Creation requests. See Section 5.2.2. The Printer SHOULD support the "job-hold-until" Job Template attribute for use in Job Creation requests with at least the 'indefinite' value, if it supports the Hold-Job operation. Otherwise, a Client cannot create a Job and hold it immediately (without picking some supported time period in the future).

If supplied and supported as specified in the Printer's "job-hold-until-supported" attribute, the Printer copies the supplied operation attribute to the Job, replacing the Job's previous "job-hold-until" attribute, if present, and makes the Job a candidate for scheduling during the supplied named time period.

If supplied but either the "job-hold-until" operation attribute itself or the value supplied is not supported, the Printer accepts the request, returns the unsupported attribute or value in the Unsupported Attributes group according to Section 4.1.7, returns the 'successful-ok-ignored-or-substituted-attributes' status-code, and holds the Job indefinitely until a Client performs a subsequent Release-Job operation.

If (1) the Client supplies either a value that specifies a time period that has already started or the 'no-hold' value (meaning don't hold the Job) and (2) the Printer supports the "job-hold-until" operation attribute and there are no other reasons to hold the Job, the Printer MUST accept the operation and make the Job be a candidate for processing immediately (see Section 5.2.2).

If the Client does not supply a "job-hold-until" operation attribute in the request, the Printer MUST populate the Job with a "job-hold-until" attribute with the 'indefinite' value (if the Printer supports the "job-hold-until" attribute) and hold the Job indefinitely, until a Client performs a Release-Job operation.

4.3.5.2. Hold-Job Response

The groups and attributes are the same as those defined for a Cancel-Job response (see Section 4.3.3.2).

4.3.6. Release-Job Operation

This OPTIONAL operation allows a Client to release a previously held Job so that it is again eligible for scheduling. If the Hold-Job operation is supported, then the Release-Job operation MUST be supported, and vice versa.

This operation removes the "job-hold-until" Job attribute, if present, from the Job that had been supplied in the Create-Job or most recent Hold-Job or Restart-Job operation and removes its effect on the Job. The Printer MUST remove the "job-hold-until-specified" value from the Job's "job-state-reasons" attribute, if present. See Section 5.3.8.

The Printer MUST accept or reject the request based on the Job's current state and transition the Job to the indicated new state as shown in Table 6.

Access Rights: The authenticated user (see Section 9.3) performing this operation must be either the Job owner or an Operator or Administrator of the Printer (see Sections 1 and 9.5). Otherwise, the Printer MUST reject the operation and return 'client-error-forbidden', 'client-error-not-authenticated', or 'client-error-not-authorized' as appropriate.

The Release-Job request and Release-Job response have the same attribute groups and attributes as the Cancel-Job operation (see Sections 4.3.3.1 and 4.3.3.2).

Current "job-state"	New "job-state"	Printer's response status-code and action:
'pending'	'pending'	'successful-ok'. No effect on the Job.
'pending-held'	'pending-held'	'successful-ok' (note 1)
'pending-held'	'pending'	'successful-ok'
'processing'	'processing'	'successful-ok'. No effect on the Job.
'processing-stopped'	'processing-stopped'	'successful-ok'. No effect on the Job.
'completed'	'completed'	'client-error-not-possible'
'canceled'	'canceled'	'client-error-not-possible'
'aborted'	'aborted'	'client-error-not-possible'

Table 6: Release-Job State Transitions

Note 1: If there are other reasons to keep the Job in the 'pending-held' state, such as 'resources-are-not-ready', the Job remains in the 'pending-held' state. Thus, the 'pending-held' state is not just for Jobs that have the "job-hold-until" attribute applied to them but is also used for any reason that will keep the Job from being a candidate for scheduling and processing, such as 'resources-are-not-ready'. See the "job-hold-until" attribute (Section 5.2.2).

4.3.7. Restart-Job Operation

This DEPRECATED operation allows a Client to restart a Job that is retained in the queue after processing has completed (see Section 5.3.7.2).

Note: This operation SHOULD NOT be supported in new implementations, since it destroys Printer accounting information. The Resubmit-Job operation [PWG5100.11] is the safe replacement for this operation and makes a copy of the Job, assigns a new "job-uri" and "job-id" to the copy, and resets the Job progress attributes in the new copy only.

The Restart-Job operation moves the Job to the 'pending' or 'pending-held' Job state and restarts at the beginning on the same Printer with the same attribute values. If any of the Documents in the Job were passed by reference (Print-URI or Send-URI), the Printer MUST refetch the data, since the semantics of Restart-Job are to repeat all Job processing. The Job Status attributes that accumulate Job progress, such as "job-impressions-completed", "job-media-sheets-completed", and "job-k-octets-processed", MUST be reset to 0 so that they give an accurate record of the Job from its restart point. The Job MUST continue to use the same "job-uri" and "job-id" attribute values.

The Printer MUST accept or reject the request based on the Job's current state and transition the Job to the indicated new state as shown in Table 7.

Note: In order to prevent a user from inadvertently restarting a Job in the middle, the Restart-Job request is rejected when the Job is in the 'processing' or 'processing-stopped' state. If in the future an operation is needed to hold or restart Jobs while in either of these states, it will be added as an additional operation, rather than overloading the Restart-Job operation, so that it is clear that the user intended that the current Job not be completed.

Access Rights: The authenticated user (see Section 9.3) performing this operation must be either the Job owner or an Operator or Administrator of the Printer (see Sections 1 and 9.5). Otherwise, the Printer MUST reject the operation and return 'client-error-forbidden', 'client-error-not-authenticated', or 'client-error-not-authorized' as appropriate.

Current "job-state"	New "job-state"	Printer's response status-code and action:
'pending'	'pending'	'client-error-not-possible'
'pending-held'	'pending-held'	'client-error-not-possible'
'processing'	'processing'	'client-error-not-possible'
'processing-stopped'	'processing-stopped'	'client-error-not-possible'
'completed'	'pending' or 'pending-held'	'successful-ok' - Job is started over.
'completed'	'completed'	'client-error-not-possible' - see Rule 1.
'canceled'	'pending' or 'pending-held'	'successful-ok' - Job is started over.
'canceled'	'canceled'	'client-error-not-possible' - see Rule 1.
'aborted'	'pending' or 'pending-held'	'successful-ok' - Job is started over.
'aborted'	'aborted'	'client-error-not-possible' - see Rule 1.

Table 7: Restart-Job State Transitions

Rule 1: If the Job Retention Period has expired for the Job in this state, then the Printer rejects the operation. See Section 5.3.7.2.

4.3.7.1. Restart-Job Request

The groups and attributes are the same as those defined for a Cancel-Job request (see Section 4.3.3.1), with the addition of the following Group 1 operation attribute:

"job-hold-until" (type2 keyword | name(MAX)):

The Client MAY supply and the Printer MUST support this operation attribute in a Restart-Job request if it supports the "job-hold-until" Job Template attribute in Job Creation requests. See Section 5.2.2.

If supplied and supported as specified in the Printer's "job-hold-until-supported" attribute, the Printer copies the supplied operation attribute to the Job, replacing the Job's previous "job-hold-until" attribute, if present, and makes the Job a candidate for scheduling during the supplied named time period. See Section 5.2.2.

If supplied but the value is not supported, the Printer accepts the request, returns the unsupported attribute or value in the Unsupported Attributes group according to Section 4.1.7, returns the 'successful-ok-ignored-or-substituted-attributes' status-code, and holds the Job indefinitely until a Client performs a subsequent Release-Job operation.

If supplied but the "job-hold-until" operation attribute itself is not supported, the Printer accepts the request, returns the unsupported attribute with the out-of-band 'unsupported' value in the Unsupported Attributes group according to Section 4.1.7, returns the 'successful-ok-ignored-or-substituted-attributes' status-code, and restarts the Job, i.e., ignores the "job-hold-until" attribute.

If (1) the Client supplies either a value that specifies a time period that has already started or the 'no-hold' value (meaning don't hold the Job) and (2) the Printer supports the "job-hold-until" operation attribute and there are no other reasons to hold the Job, the Printer makes the Job a candidate for processing immediately (see Section 5.2.2).

If the Client does not supply a "job-hold-until" operation attribute in the request, the Printer removes the "job-hold-until" attribute, if present, from the Job. If there are no other reasons to hold the Job, the Restart-Job operation makes the Job a candidate for processing immediately (see Section 5.2.2).

4.3.7.2. Restart-Job Response

The groups and attributes are the same as those defined for a Cancel-Job response (see Section 4.3.3.2).

5. Object Attributes

This section describes the attributes with their corresponding attribute syntaxes and values that are part of the IPP Model. The sections below show the objects and their associated attributes that are included within the scope of this protocol. Many of these attributes are derived from other relevant documents:

- Document Printing Application (DPA) [ISO10175]
- Printer MIB v2 [RFC3805]

Each attribute is uniquely identified in this document using a "keyword" (see Section 2.3.7) that is the name of the attribute. The keyword is included in the section title describing that attribute.

Note: Not only are keywords used to identify attributes, but one of the attribute syntaxes described below is "keyword" so that some attributes have ’keyword’ values. Therefore, these attributes are defined as having an attribute syntax that is a set of keywords.

5.1. Attribute Syntaxes

This section defines the basic attribute syntax types that all Clients and IPP objects MUST be able to accept in responses and accept in requests, respectively. Each attribute description in Sections 4 and 5 includes in the section title the name of the attribute with its syntax(es) in parentheses. A conforming implementation of an attribute MUST include the semantics of the attribute syntax(es) so identified. Section 7.7 describes how the protocol can be extended with new attribute syntaxes.

The attribute syntaxes are specified in the following subsections, where the subsection title is the keyword name of the attribute syntax inside the single quotes. In operation requests and responses, each attribute value MUST be represented as one of the attribute syntaxes specified in the subsection title for the attribute. In addition, the value of an attribute in a response (but not in a request) MAY be one of the "out-of-band" values (Section 5.1.1) whose special encoding rules are defined in the Encoding and Transport document [RFC8010].

All attributes in a request MUST have one or more values as defined in Sections 5.2, 5.3, and 5.4. All attributes in a response MUST have either (1) one or more values as defined in Sections 5.2, 5.3, and 5.4 or (2) a single "out-of-band" value.

Most attributes are defined to have a single attribute syntax. However, a few attributes (e.g., "job-sheet", "media", "job-hold-until") are defined to have several attribute syntaxes, depending on the value. These multiple attribute syntaxes are separated by the "|" character in the subsection title to indicate the choice. Since each value MUST be tagged as to its attribute syntax in the protocol, a single-valued attribute instance can have any one of its attribute syntaxes and a multi-valued attribute instance can have a mixture of its defined attribute syntaxes.

5.1.1. Out-of-Band Values - 'unknown', 'unsupported', and 'no-value'

This document defines three "out-of-band" values that are used in place of an attribute's defined syntax:

- 'unknown': The attribute is supported by the IPP object, but the value is unknown to the IPP object for some reason. This out-of-band value is used for attributes that have an intrinsic, physical value that cannot be determined by the IPP object at a given time, e.g., sheet count, geo-location, etc.

- 'unsupported': The attribute is unsupported by the IPP object. This value MUST be returned only as the value of an attribute in the Unsupported Attributes group.

- 'no-value': The attribute is supported by the Printer, but the Administrator has not yet configured a value.

5.1.2. 'text'

A 'text' attribute is an attribute whose value is a sequence of zero or more characters encoded in a maximum of 1023 ('MAX') octets. MAX is the maximum length for each value of any 'text' attribute. However, if an attribute will always contain values whose maximum length is much less than MAX, the definition of that attribute will include a qualifier that defines the maximum length for values of that attribute. For example, the "printer-location" attribute is specified as "printer-location (text(127))". In this case, text values for "printer-location" MUST NOT exceed 127 octets; if supplied with a longer text string via some external interface (other than the protocol), implementations are free to truncate to this shorter length limitation.

In this document, all 'text' attributes are defined using the 'text' syntax. However, 'text' is used only for brevity; the formal interpretation of 'text' is 'textWithoutLanguage | textWithLanguage'. That is, for any attribute defined in this document using the 'text' attribute syntax, all IPP objects and Clients MUST support both the 'textWithoutLanguage' and 'textWithLanguage' attribute syntaxes. However, in actual usage and protocol execution, IPP objects and Clients accept and return only one of the two syntaxes per attribute. The syntax 'text' never appears "on-the-wire".

Both 'textWithoutLanguage' and 'textWithLanguage' are needed to support the real-world needs of interoperability between sites and systems that use different natural languages as the basis for human communication. Generally, one natural language applies to all 'text' attributes in a given request or response. The language is indicated by the "attributes-natural-language" operation attribute defined in Section 4.1.4 or the "attributes-natural-language" Job attribute defined in Section 5.3.20, and there is no need to identify the natural language for each text string on a value-by-value basis. In these cases, the attribute syntax 'textWithoutLanguage' is used for 'text' attributes. In other cases, the Client needs to supply or the Printer needs to return a text value in a natural language that is different from the rest of the text values in the request or response. In these cases, the Client or Printer uses the attribute syntax 'textWithLanguage' for 'text' attributes (this is the Natural Language Override mechanism described in Section 4.1.4).

The 'textWithoutLanguage' and 'textWithLanguage' attribute syntaxes are described in more detail in the following sections.

5.1.2.1. 'textWithoutLanguage'

The 'textWithoutLanguage' syntax indicates a value that is a sequence of zero or more characters encoded in a maximum of 1023 (MAX) octets. Text strings are encoded using the rules of some charset. The Printer MUST support the UTF-8 charset [RFC3629] and MAY support additional charsets to represent 'text' values, provided that the charsets are registered with IANA [IANA-CS]. See Section 5.1.8 for the definition of the 'charset' attribute syntax, including restricted semantics and examples of charsets.

5.1.2.2. 'textWithLanguage'

The 'textWithLanguage' attribute syntax is a compound attribute syntax consisting of two parts: a 'textWithoutLanguage' part encoded in a maximum of 1023 (MAX) octets plus an additional 'naturalLanguage' (see Section 5.1.9) part that overrides the natural language in force. The 'naturalLanguage' part explicitly identifies

the natural language that applies to the text part of that value and that value alone. For any given 'text' attribute, the 'textWithoutLanguage' part is limited to the maximum length defined for that 'text' attribute, and the 'naturalLanguage' part is always limited to 63 (additional) octets. Using the 'textWithLanguage' attribute syntax rather than the normal 'textWithoutLanguage' syntax is the so-called "Natural Language Override mechanism" and MUST be supported by all IPP objects and Clients.

If the attribute is multi-valued (1setOf text), then the 'textWithLanguage' attribute syntax MUST be used to explicitly specify each attribute value whose natural language needs to be overridden. Other values in a multi-valued 'text' attribute in a request or a response revert to the natural language of the operation attribute.

In a Job Creation request, the Printer MUST accept and store with the Job any natural language in the "attributes-natural-language" operation attribute, whether the Printer supports that natural language or not. Furthermore, the Printer MUST accept and store any 'textWithLanguage' attribute value, whether the Printer supports that natural language or not. These requirements are independent of the value of the "ipp-attribute-fidelity" operation attribute that the Client MAY supply.

Example: If the Client supplies the "attributes-natural-language" operation attribute with the value 'en' indicating English but the value of the "job-name" attribute is in French, the Client MUST use the 'textWithLanguage' attribute syntax with the following two values:

'fr': Natural Language Override indicating French

'Rapport Mensuel': the Job name in French

See the Encoding and Transport document [RFC8010] for the encoding of the two parts and a detailed example of the 'textWithLanguage' attribute syntax.

5.1.3. 'name'

This syntax type is used for user-friendly strings, such as a Printer name, that, for humans, are more meaningful than identifiers. Names are never translated from one natural language to another. The 'name' attribute syntax is essentially the same as 'text', including the REQUIRED support of UTF-8, except that the sequence of characters is limited so that its encoded form MUST NOT exceed 255 (MAX) octets.

Also, like 'text', 'name' is really an abbreviated notation for either 'nameWithoutLanguage' or 'nameWithLanguage'. That is, all IPP objects and Clients MUST support both the 'nameWithoutLanguage' and 'nameWithLanguage' attribute syntaxes. However, in actual usage and protocol execution, IPP objects and Clients accept and return only one of the two syntaxes per attribute. The syntax 'name' never appears "on-the-wire".

Only the 'text' and 'name' attribute syntaxes permit the Natural Language Override mechanism.

Some attributes are defined as 'type2 keyword | name'. These attributes support values that are either type2 keywords or names. This dual-syntax mechanism enables a site Administrator to extend these attributes to legally include values that are locally defined by the site Administrator. Such names are not registered with IANA.

5.1.3.1. 'nameWithoutLanguage'

The 'nameWithoutLanguage' syntax indicates a value that is a sequence of zero or more characters encoded in a maximum of 255 (MAX) octets.

5.1.3.2. 'nameWithLanguage'

The 'nameWithLanguage' attribute syntax is a compound attribute syntax consisting of two parts: a 'nameWithoutLanguage' (see Section 5.1.3.1) part plus an additional 'naturalLanguage' (see Section 5.1.9) part that overrides the natural language in force. The 'naturalLanguage' part explicitly identifies the natural language that applies to that name value and that name value alone. For any given 'name' attribute, the 'nameWithoutLanguage' part is limited to the maximum length defined for that 'name' attribute, and the 'naturalLanguage' part is always limited to 63 (additional) octets. Using the 'nameWithLanguage' attribute syntax rather than the normal 'nameWithoutLanguage' syntax is the Natural Language Override mechanism and MUST be supported by all IPP objects and Clients.

The 'nameWithLanguage' attribute syntax behaves the same as the 'textWithLanguage' syntax. If a name is in a language that is different than the rest of the object or operation, then this 'nameWithLanguage' syntax is used rather than the generic 'nameWithoutLanguage' syntax.

If the attribute is multi-valued (1setOf name), then the 'nameWithLanguage' attribute syntax MUST be used to explicitly specify each attribute value whose natural language needs to be

overridden. Other values in a multi-valued 'name' attribute in a request or a response revert to the natural language of the operation attribute.

In a Job Creation request, the Printer MUST accept and store with the Job any natural language in the "attributes-natural-language" operation attribute, whether the Printer supports that natural language or not. Furthermore, the Printer MUST accept and store any 'nameWithLanguage' attribute value, whether the Printer supports that natural language or not. These requirements are independent of the value of the "ipp-attribute-fidelity" operation attribute that the Client MAY supply.

Example: If the Client supplies the "attributes-natural-language" operation attribute with the value 'en' indicating English but the "printer-name" attribute is in German, the Client MUST use the 'nameWithLanguage' attribute syntax as follows:

'de': Natural Language Override indicating German

'Farbdrucker': the Printer name in German

See the Encoding and Transport document [RFC8010] for the encoding of the two parts and a detailed example of the 'nameWithLanguage' attribute syntax.

5.1.3.3. Matching 'name' Attribute Values

For purposes of matching two 'name' attribute values for equality, such as in Job validation (where a Client-supplied value for attribute "xxx" is checked to see if the value is among the values of the Printer's corresponding "xxx-supported" attribute), the following match rules apply:

1. 'keyword' values never match 'name' values.

2. 'name' ('nameWithoutLanguage' and 'nameWithLanguage') values match if (1) the name parts match and (2) the Associated Natural Language parts (see Section 4.1.4.1) match. The matching rules are as follows:

   2a. The name parts match if the two names are identical character by character, except that it is RECOMMENDED that case be ignored as defined in "i;unicode-casemap - Simple Unicode Collation Algorithm" [RFC5051]. For example, 'Ajax-letter-head-white' MUST match 'Ajax-letter-head-white' and SHOULD match 'ajax-letter-head-white' and 'AJAX-LETTER-HEAD-WHITE'.

2b. The Associated Natural Language parts match if the shorter of the two meets the syntactic requirements defined in Section 2.1 of RFC 5646 [RFC5646] and matches (byte for byte, since IPP language tags are lowercase) with the longer. For example, 'en' matches 'en', 'en-us', and 'en-gb' but matches neither 'fr' nor 'e'.

5.1.4. 'keyword'

The 'keyword' attribute syntax is a sequence of characters, of length 1 to 255, containing only the US-ASCII [RFC20] encoded values for lowercase letters ("a"-"z"), digits ("0"-"9"), hyphen ("-"), dot ("."), and underscore ("_"). The first character MUST be a lowercase letter. Furthermore, keywords MUST be in US English.

This syntax type is used for enumerating semantic identifiers of entities in the abstract protocol, i.e., entities identified in this document. Keywords are used as attribute names or values of attributes. Unlike 'text' and 'name' attribute values, 'keyword' values MUST NOT use the Natural Language Override mechanism, since they MUST always be US-ASCII and US English.

Keywords are for use in the protocol. A user interface will likely provide a mapping between protocol keywords and displayable user-friendly words and phrases that are localized to the natural language of the user. While the keywords specified in this document MAY be displayed to users whose natural language is US English, they MAY be mapped to other US English words for US English users, since the user interface is outside the scope of this document.

In the definition for each attribute of this syntax type, the full set of 'keyword' values being defined for that attribute is listed. The IANA IPP registry will always contain the complete and current list of 'keyword' values for the attribute.

When a keyword is used to represent an attribute (its name), it MUST be unique within the full scope of all IPP objects and attributes. When a keyword is used to represent a value of an attribute, it MUST be unique just within the scope of that attribute. That is, the same keyword MUST NOT be used for two different values within the same attribute to mean two different semantic ideas. However, the same keyword MAY be used across two or more attributes, representing

different semantic ideas for each attribute. Section 7.3 describes how the protocol can be extended with new 'keyword' values. Examples of attribute name keywords are:

"job-name"

"attributes-charset"

Note: This document uses "type1" and "type2" prefixes to the "keyword" basic syntax to indicate different levels of review for extensions (see Section 7.3).

5.1.5. 'enum'

The 'enum' attribute syntax is an enumerated integer value that is in the range from 1 to 2**31 - 1 (MAX). Each value has an associated 'keyword' name. In the definition for each attribute of this syntax type, the full set of possible values for that attribute is listed. This syntax type is used for attributes for which there are enum values assigned by other standards, such as SNMP MIBs. A number of attribute enum values in this document are also used for corresponding attributes in other standards [RFC3805]. This syntax type is not used for attributes to which the Administrator can assign values. Section 7.4 describes how the protocol can be extended with new enum values.

Enum values are for use in the protocol. A user interface will provide a mapping between protocol enum values and displayable user-friendly words and phrases that are localized to the natural language of the user. While the enum symbols specified in this document MAY be displayed to users whose natural language is US English, they MAY be mapped to other US English words for US English users, since the user interface is outside the scope of this document.

Note: Some SNMP MIBs use '2' for 'unknown', which corresponds to the IPP "out-of-band" value 'unknown'. See the description of the "out-of-band" values at the beginning of Section 5.1. Therefore, attributes of type 'enum' typically start at '3'.

Note: This document uses "type1" and "type2" prefixes to the "enum" basic syntax to indicate different levels of review for extensions (see Section 7.4).

5.1.6. 'uri'

The 'uri' attribute syntax is any valid Uniform Resource Identifier (URI) [RFC3986]. Most often, URIs are simply Uniform Resource Locators (URLs). The maximum length of URIs used as values of IPP attributes is 1023 octets. Although most other IPP attribute syntax types allow for only lowercase values, this attribute syntax type conforms to the case-sensitive and case-insensitive rules specified in [RFC3986]. See also [RFC3196] for a discussion of case in URIs.

5.1.7. 'uriScheme'

The 'uriScheme' attribute syntax is a sequence of characters representing a URI scheme according to RFC 3986 [RFC3986]. Though RFC 3986 requires that the values be case insensitive, IPP requires all lowercase values in IPP attributes, to simplify comparing by IPP Clients and Printers.

Standard values for this syntax type include the following keywords:

- 'ipp': for IPP schemed URIs, e.g., "ipp://example.com/ipp/..." [RFC3510]
- 'ipps': for IPPS schemed URIs, e.g., "ipps://example.com/ipp/..." [RFC7472]
- 'http': for HTTP schemed URIs, e.g., "http://example.com/path/to/filename" [RFC7230]
- 'https': for HTTPS schemed URIs, e.g., "https://example.com/path/to/filename" [RFC7230]
- 'ftp': for FTP schemed URIs, e.g., "ftp://example.com/path/to/filename" [RFC1738]
- 'mailto': for SMTP schemed URIs, e.g., "mailto:user@example.com" [RFC6068]
- 'file': for file schemed URIs, e.g., "file:///path/to/filename" [RFC1738]
- 'urn': for Uniform Resource Name schemed URIs, e.g., "urn:uuid:01234567-89ab-cdef-fedc-ba9876543210" [RFC4122]

A Printer MAY support any URI 'scheme' that has been registered with IANA [IANA-MT]. The maximum length of URI 'scheme' values used to represent IPP attribute values is 63 octets.

5.1.8. 'charset'

The 'charset' attribute syntax is a standard identifier for a charset. A charset is a coded character set and encoding scheme. Charsets are used for labeling certain Document contents, 'text' attribute values, and 'name' attribute values. The syntax and semantics of this attribute syntax are specified in RFC 2046 [RFC2046] and contained in the IANA "Character Sets" registry [IANA-CS] according to the IANA procedures [RFC2978]. Though RFC 2046 requires that the values be case-insensitive US-ASCII [RFC20], IPP requires all lowercase values in IPP attributes, to simplify comparing by IPP Clients and Printers. When a character set in the IANA registry has more than one name (alias), the name labeled as "(preferred MIME name)", if present, MUST be used.

The maximum length of 'charset' values used to represent IPP attribute values is 63 octets.

Some examples are:

- 'utf-8': ISO 10646 Universal Multiple-Octet Coded Character Set (UCS) [ISO10646] represented as the UTF-8 [RFC3629] transfer encoding scheme in which US-ASCII [RFC20] is a subset charset.

- 'us-ascii': 7-bit American Standard Code for Information Interchange (ASCII) [RFC20].

- 'iso-8859-1': 8-bit One-Byte Coded Character Set, Latin Alphabet No. 1 [ISO8859-1]. That standard defines a coded character set that is used by Latin languages in the Western Hemisphere and Western Europe. US-ASCII is a subset charset.

Some attribute descriptions MAY place additional requirements on charset values that can be used, such as REQUIRED values that MUST be supported or additional restrictions, such as requiring that the charset have US-ASCII as a subset charset.

5.1.9. 'naturalLanguage'

The 'naturalLanguage' attribute syntax is a standard identifier for a natural language and, optionally, a country or region. The values for this syntax type are defined by RFC 5646 [RFC5646]. Though RFC 5646 requires that the values be case-insensitive US-ASCII, IPP requires all lowercase values in IPP attributes, to simplify comparing by IPP Clients and Printers. Examples include:

- 'en': for English
- 'en-us': for US English
- 'fr': for French
- 'de': for German

The maximum length of 'naturalLanguage' values used to represent IPP attribute values is 63 octets.

Note: While any standard natural language identifier defined in RFC 5646 can be used, Clients typically only support a subset of these identifiers. When comparing two identifiers or performing lookups, Printers SHOULD be prepared to match legacy identifiers with their corresponding modern equivalents and vice versa.

5.1.10. 'mimeMediaType'

The 'mimeMediaType' attribute syntax is the Internet media type (sometimes called "MIME type") as defined by RFC 2046 [RFC2046] and registered according to the procedures of RFC 6838 [RFC6838] for identifying a Document format. The value MAY include a charset parameter, or some other parameter, depending on the specification of the media type in the IANA "Media Types" registry [IANA-MT]. Although most other IPP syntax types allow for only lowercase values, this syntax type allows for mixed-case values that are case insensitive.

Examples are:

- 'text/html': An HTML Document
- 'text/plain': A plain text Document in US-ASCII (RFC 2046 indicates that in the absence of the charset parameter MUST mean US-ASCII rather than simply unspecified) [RFC2046]
- 'text/plain; charset = US-ASCII': A plain text Document in US-ASCII

- 'text/plain; charset = ISO-8859-1': A plain text Document in ISO 8859-1 (Latin 1) [ISO8859-1]

- 'text/plain; charset = utf-8': A plain text Document in ISO 10646 represented as UTF-8 [RFC3629]

- 'application/postscript': A PostScript Document [RFC2046]

- 'application/vnd.hp-PCL': A PCL Document [IANA-MT] (charset escape sequence embedded in the Document data)

- 'application/pdf': Portable Document Format [ISO32000]

- 'application/octet-stream': Auto-sense - see Section 5.1.10.1

The maximum length of a 'mimeMediaType' value to represent IPP attribute values is 255 octets.

5.1.10.1. 'application/octet-stream' - Auto-Sensing the Document Format

One special type is 'application/octet-stream'. If the Printer supports this value, the Printer MUST be capable of auto-sensing the format of the Document data using an implementation-dependent method that examines some number of octets of the Document data, either as part of the Job Creation request and/or at Document processing time. During auto-sensing, a Printer can determine that the Document data has a format that the Printer doesn't recognize. If the Printer determines this problem before returning an operation response, it rejects the request and returns the 'client-error-document-format-not-supported' status-code. If the Printer determines this problem after accepting the request and returning an operation response with one of the successful status-code values, the Printer adds the 'unsupported-document-format' value to the Job's "job-state-reasons" attribute.

If the Printer's default value attribute "document-format-default" is set to 'application/octet-stream', the Printer not only supports auto-sensing of the Document format but will depend on the result of applying its auto-sensing when the Client does not supply the "document-format" attribute. If the Client supplies a Document format value, the Printer MUST rely on the supplied attribute, rather than trust its auto-sensing algorithm. To summarize:

1. If the Client does not supply a Document format value, the Printer MUST rely on its default value setting (which can be 'application/octet-stream' indicating an auto-sensing mechanism).

2. If the Client supplies a value other than 'application/octet-stream', the Client is supplying valid information about the format of the Document data and the Printer MUST trust the Client-supplied value more than the outcome of applying an automatic format detection mechanism. For example, the Client can request the printing of a PostScript file as a 'text/plain' Document. The Printer MUST print a text representation of the PostScript commands rather than interpret the stream of PostScript commands and print the result.

3. If the Client supplies a value of 'application/octet-stream', the Client is indicating that the Printer MUST use its auto-sensing mechanism on the Client-supplied Document data whether auto-sensing is the Printer's default or not.

Note: Since the auto-sensing algorithm is probabilistic, if the Client requests both auto-sensing ("document-format" set to 'application/octet-stream') and true fidelity ("ipp-attribute-fidelity" set to 'true'), the Printer might not be able to guarantee exactly what the End User intended (the auto-sensing algorithm might mistake one Document format for another), but it is able to guarantee that its auto-sensing mechanism will be used.

5.1.11. 'octetString'

The 'octetString' attribute syntax is a sequence of octets encoded in a maximum of 1023 octets that is indicated in syntax definitions using the notation 'octetString(MAX)'. This syntax type is used for opaque data.

5.1.12. 'boolean'

The 'boolean' attribute syntax has only two values: 'true' and 'false'.

5.1.13. 'integer'

The 'integer' attribute syntax is an integer value that is in the range from -2**31 (MIN) to 2**31 - 1 (MAX). Each individual attribute can specify the range constraint explicitly if the range is different from the full range of possible integer values -- for example, job-priority (integer(1:100)) for the "job-priority" attribute, as shown in the title of Section 5.2.1. However, the enforcement of that additional constraint is up to the IPP objects, not the protocol.

5.1.14. 'rangeOfInteger'

The 'rangeOfInteger' attribute syntax is an ordered pair of integers that defines an inclusive range of integer values. The first integer specifies the lower bound, and the second specifies the upper bound. If a range constraint is specified in the attribute definition, i.e., 'rangeOfInteger(X:Y)' indicating X as a minimum value and Y as a maximum value, then the constraint applies to both integers.

5.1.15. 'dateTime'

The 'dateTime' attribute syntax is a standard, fixed-length, 11-octet representation of the "DateAndTime" syntax as defined in RFC 2579 [RFC2579]. RFC 2579 also identifies an 8-octet representation of a "DateAndTime" value, but IPP objects MUST use the 11-octet representation. A user interface will provide a mapping between protocol dateTime values and displayable user-friendly words or presentation values and phrases that are localized to the natural language and date format of the user, including time zone.

5.1.16. 'resolution'

The 'resolution' attribute syntax specifies a two-dimensional resolution in the indicated units. It consists of three values: a cross-feed direction resolution (positive integer value), a feed direction resolution (positive integer value), and a units value. The semantics of these three components are taken from the suggested values in the Printer MIB [RFC3805]. That is, the cross-feed direction resolution component is the same as the prtMarkerAddressabilityXFeedDir object in the Printer MIB, the feed direction resolution component is the same as the prtMarkerAddressabilityFeedDir in the Printer MIB, and the units component is the same as the prtMarkerAddressabilityUnit object in the Printer MIB (namely, '3' indicates dots per inch and '4' indicates dots per centimeter). All three values MUST be present even if the first two values are the same. For example, '300', '600', '3' indicates a 300-dpi cross-feed direction resolution and a 600-dpi feed direction resolution, since a '3' indicates dots per inch (dpi).

5.1.17. 'collection'

The 'collection' attribute syntax is a container holding one or more named values (i.e., attributes), which are called "member attributes". Each 'collection' attribute definition Document lists the mandatory and optional member attributes of each collection value. A collection value is similar to an IPP attribute group in a

request or a response, such as the Operation Attributes group -- they both consist of a set of attributes. Collections can also be nested, i.e., a collection in a collection.

A collection value consists of three separate components:

- A 'begCollection' value with an optional octet string value starting the collection,
- Zero or more member attributes defined using a series of unnamed values starting with a 'memberAttrName' value that specifies the member attribute name, and
- An 'endCollection' value with an optional name plus octet string value finishing the collection.

5.1.18. '1setOf X'

The '1setOf X' attribute syntax is one or more values of attribute syntax type X. This syntax type is used for multi-valued attributes. The syntax type is called '1setOf' rather than just 'setOf' as a reminder that the set of values MUST NOT be empty (i.e., a set of size 0). Sets are normally unordered; however, each attribute description of this type can specify that the values MUST be in a certain order for that attribute.

5.2. Job Template Attributes

Job Template attributes describe Job processing intent. Clients MAY supply (in Job Creation requests) and Printers SHOULD support Job Template attributes. See Section 2.3.11 for a description of support for OPTIONAL attributes.

Job Template attributes conform to the following rules. For each Job Template attribute called "xxx":

1. If the Printer supports "xxx", then it MUST support both an "xxx-default" attribute (unless there is a "No" in Table 8 below) and an "xxx-supported" attribute. If the Printer doesn't support "xxx", then it MUST support neither an "xxx-default" attribute nor an "xxx-supported" attribute, and it MUST treat an attribute "xxx" supplied by a Client as unsupported. An attribute "xxx" can be supported for some Document formats and not supported for other Document formats. For example, it is expected that a Printer would only support "orientation-requested" for some Document formats (such as 'text/plain' or 'text/html') but not others (such as 'application/postscript').

2. Clients MAY supply "xxx" in a Job Creation request. If "xxx" is supplied, the Client is indicating a desired Job processing behavior for this Job. When "xxx" is not supplied, the Client is indicating that the Printer apply its default Job processing behavior at Job processing time if the Document content does not contain an embedded instruction indicating an xxx-related behavior.

   Since an Administrator MAY change the default value attribute after a Job has been submitted but before it has been processed, the default value used by the Printer at Job processing time can be different than the default value in effect at Job submission time.

3. The "xxx-supported" attribute is a Printer attribute that describes which Job processing behaviors are supported by that Printer. A Client can query the Printer to find out what xxx-related behaviors are supported by inspecting the returned values of the "xxx-supported" attribute.

   Note: The "xxx" in each "xxx-supported" attribute name is singular, even though an "xxx-supported" attribute usually has more than one value, such as "print-quality-supported", unless the "xxx" Job Template attribute is plural, such as "finishings" or "sides". In such cases, the "xxx-supported" attribute names are "finishings-supported" and "sides-supported".

4. The "xxx-default" default value attribute describes what will be done at Job processing time when no other Job processing information is supplied by the Client (either explicitly as an IPP attribute in the Job Creation request or implicitly as an embedded instruction within the Document data).

If an application wishes to present an End User with a list of supported values from which to choose, the application SHOULD query the Printer for its supported value attributes. The application SHOULD also query the default value attributes. If the application then limits selectable values to only those values that are supported, the application can guarantee that the values supplied by the Client in the Job Creation request all fall within the set of supported values at the Printer. When querying the Printer, the Client MAY enumerate each attribute by name in the Get-Printer-Attributes request, or the Client MAY just name the "job-template" group in order to get the complete set of supported attributes (both supported and default attributes).

The "finishings" attribute is an example of a Job Template attribute. It can take on a set of values such as '4' ('staple'), '5' ('punch'), and/or '6' ('cover'); see Table 10 in Section 5.2.6. A Client can query the Printer for the "finishings-supported" attribute and the "finishings-default" attribute. The supported attribute contains a set of supported values. The default value attribute contains the finishing value(s) that will be used for a new Job if the Client does not supply a "finishings" attribute in the Job Creation request and the Document data does not contain any corresponding finishing instructions. If the Client does supply the "finishings" attribute in the Job Creation request, the Printer validates the value or values to make sure that they are a subset of the supported values identified in the Printer's "finishings-supported" attribute. See Section 4.1.7.

Table 8 below summarizes the names and relationships for all Job Template attributes. The first column of the table (labeled "Job Attribute") shows the name and syntax for each Job Template attribute in the Job. These are the attributes that can optionally be supplied by the Client in a Job Creation request. The last two columns (labeled "Printer: Default Value Attribute" and "Printer: "Supported Values" Attribute") show the name and syntax for each Job Template attribute in the Printer (the default value attributes and the "supported values" attributes). A "No" in the table means the Printer MUST NOT support the attribute (that is, the attribute is simply not applicable). For brevity in the table, the 'text' and 'name' entries do not show the maximum length for each attribute.

Job Attribute	Printer: Default Value Attribute	Printer: "Supported Values" Attribute
job-priority (integer 1:100)	job-priority-default (integer 1:100)	job-priority-supported (integer 1:100)
job-hold-until (type2 keyword | name)	job-hold-until-default (type2 keyword | name)	job-hold-until-supported (1setOf (type2 keyword | name))
job-sheets (type2 keyword | name)	job-sheets-default (type2 keyword | name)	job-sheets-supported (1setOf (type2 keyword | name))
multiple-document-handling (type2 keyword)	multiple-document-handling-default (type2 keyword)	multiple-document-handling-supported (1setOf type2 keyword)

copies (integer(1:MAX))	copies-default (integer(1:MAX))	copies-supported (rangeOfInteger(1:MAX))
finishings (1setOf type2 enum)	finishings-default (1setOf type2 enum)	finishings-supported (1setOf type2 enum)
page-ranges (1setOf rangeOfInteger (1:MAX))	No	page-ranges-supported (boolean)
sides (type2 keyword)	sides-default (type2 keyword)	sides-supported (1setOf type2 keyword)
number-up (integer(1:MAX))	number-up-default (integer(1:MAX))	number-up-supported (1setOf (integer(1:MAX) | rangeOfInteger(1:MAX)))
orientation-requested (type2 enum)	orientation-requested-default (type2 enum)	orientation-requested-supported (1setOf type2 enum)
media (type2 keyword | name)	media-default (type2 keyword | name)	media-supported (1setOf (type2 keyword | name)) media-ready (1setOf (type2 keyword | name))
printer-resolution (resolution)	printer-resolution-default (resolution)	printer-resolution-supported (1setOf resolution)
print-quality (type2 enum)	print-quality-default (type2 enum)	print-quality-supported (1setOf type2 enum)

Table 8: Job Template Attributes

5.2.1. job-priority (integer(1:100))

This attribute specifies a priority for scheduling the Job. A higher value specifies a higher priority. The value 1 indicates the lowest possible priority. The value 100 indicates the highest possible priority. Among those Jobs that are ready to print, a Printer MUST print all Jobs with a priority value of n before printing those with a priority value of n - 1 for all n.

If the Printer supports this attribute, it MUST always support the full range from 1 to 100. No administrative restrictions are permitted. This way, an End User can always make full use of the entire range with any Printer. If privileged Jobs are implemented outside IPP, they MUST have priorities higher than 100, rather than restricting the range available to End Users.

If the Client does not supply this attribute and this attribute is supported by the Printer, the Printer MUST use the value of the Printer's "job-priority-default" attribute at Job submission time (unlike most Job Template attributes that are used if necessary at Job processing time).

The syntax for the "job-priority-supported" attribute is also integer(1:100). This single integer value indicates the number of priority levels supported. The Printer MUST take the value supplied by the Client and map it to the closest integer in a sequence of n integer values that are evenly distributed over the range from 1 to 100 using the formula:

```
roundToNearestInt((100x + 50) / n)
```

where n is the value of "job-priority-supported" and x ranges from 0 through (n - 1).

For example, if n = 1, the sequence of values is 50; if n = 2, the sequence of values is 25 and 75; if n = 3, the sequence of values is 17, 50, and 83; if n = 10, the sequence of values is 5, 15, 25, 35, 45, 55, 65, 75, 85, and 95; if n = 100, the sequence of values is 1, 2, 3, ... 100.

Table 9 shows how a Printer maps Client-supplied "job-priority" values for example values of n.

job-priority	n = 1	n = 2	n = 10
1	50	17	5
10	50	17	5
20	50	17	15
30	50	17	25
40	50	50	35
50	50	50	45
60	50	50	55
70	50	50	65
80	50	83	75
90	50	83	85
100	50	83	95

Table 9: "job-priority" Values

5.2.2. job-hold-until (type2 keyword | name(MAX))

This attribute specifies the named time period during which the Job MUST become a candidate for printing.

Standard ’keyword’ values for named time periods are:

- ’no-hold’: immediately, if there are no other reasons to hold the job
- ’indefinite’: the Job is held indefinitely, until a Client performs a Release-Job (Section 4.3.6)
- ’day-time’: during the day
- ’evening’: evening

- 'night': night
- 'weekend': weekend
- 'second-shift': second shift (after close of business)
- 'third-shift': third shift (after midnight)

An Administrator MUST associate allowable print times with a named time period (by means outside the scope of this IPP/1.1 document). An Administrator is encouraged to pick names that suggest the type of time period. An Administrator MAY define additional values using the 'name' or 'keyword' attribute syntax, depending on implementation.

If the value of this attribute specifies a time period that is in the future, the Printer SHOULD add the "job-hold-until-specified" value to the Job's "job-state-reasons" attribute, MUST move the Job to the 'pending-held' state, and MUST NOT schedule the Job for printing until the specified time period arrives.

When the specified time period arrives, the Printer MUST remove the "job-hold-until-specified" value from the Job's "job-state-reasons" attribute, if present. If there are no other Job state reasons that keep the Job in the 'pending-held' state, the Printer MUST consider the Job as a candidate for processing by moving the Job to the 'pending' state.

If this Job attribute value is the named value 'no-hold' or the specified time period has already started, the Job MUST be a candidate for processing immediately.

If the Client does not supply this attribute and this attribute is supported by the Printer, the Printer MUST use the value of the Printer's "job-hold-until-default" at Job submission time (unlike most Job Template attributes that are used if necessary at Job processing time).

5.2.3. job-sheets (type2 keyword | name(MAX))

This attribute determines which Job start/end sheet(s), if any, MUST be printed with a Job.

Standard 'keyword' values are:

- 'none': no Job sheet is printed
- 'standard': one or more site-specific standard Job sheets are printed, e.g., a single start sheet or both start and end sheets

An Administrator MAY define additional values using the 'name' or 'keyword' attribute syntax, depending on implementation.

The effect of this attribute on Jobs with multiple Documents MAY be affected by the "multiple-document-handling" Job attribute (Section 5.2.4), depending on the Job sheet semantics.

5.2.4. multiple-document-handling (type2 keyword)

This RECOMMENDED attribute controls which Impressions and Media Sheets constitute a Set for copy generation and finishing processes. When the value of the "copies" attribute exceeds '1', it also controls the order in which the copies that result from processing the Documents are produced. For the purposes of this explanation, if "a" represents an instance of Document data, then the result of processing the data in Document "a" is a sequence of Media Sheets represented by "a(*)". This attribute MUST be supported with at least one value if the Printer supports multiple Documents per Job (see Sections 4.2.4 and 4.3.1).

Standard 'keyword' values are:

- 'single-document': If a Job has multiple Documents, say, the Document data is called "a" and "b", then the result of processing all the Document data (a and then b) MUST be treated as a single sequence of Media Sheets for finishing processes; that is, finishing is performed on the concatenation of the sequences a(*),b(*). The Printer MUST NOT force the data in each Document instance to be formatted onto a new Impression, nor to start a new Impression on a new Media Sheet. If more than one copy is made, the ordering of the sets of Media Sheets resulting from processing the Document data MUST be a(*), b(*), a(*), b(*), ..., and the Printer MUST force each copy (a(*),b(*)) to start on a new Media Sheet.

- 'separate-documents-uncollated-copies': If a Job has multiple Documents, say, the Document data is called "a" and "b", then the result of processing the data in each Document instance MUST be treated as a single sequence of Media Sheets for finishing processes; that is, the sets a(*) and b(*) would each be finished separately. The Printer MUST force each copy of the result of processing the data in a single Document to start on a new Media Sheet. If more than one copy is made, the ordering of the sets of Media Sheets resulting from processing the Document data MUST be a(*), a(*), ..., b(*), b(*), ...

- 'separate-documents-collated-copies': If a Job has multiple Documents, say, the Document data is called "a" and "b", then the result of processing the data in each Document instance MUST be treated as a single sequence of Media Sheets for finishing processes; that is, the sets a(*) and b(*) would each be finished separately. The Printer MUST force each copy of the result of processing the data in a single Document to start on a new Media Sheet. If more than one copy is made, the ordering of the sets of Media Sheets resulting from processing the Document data MUST be a(*), b(*), a(*), b(*), ...

- 'single-document-new-sheet': Same as 'single-document', except that the Printer MUST ensure that the first Impression of each Document instance in the Job is placed on a new Media Sheet. This value allows multiple Documents to be stapled together with a single staple where each Document starts on a new Media Sheet.

The 'single-document' value is the same as 'separate-documents-collated-copies' with respect to the ordering of Input Pages, but not Media Sheet generation, since 'single-document' will put the first page of the next Document on the back side of a Media Sheet if an odd number of pages have been produced so far for the Job, while 'separate-documents-collated-copies' always forces the next Document or Document copy on to a new Media Sheet. In addition, if the "finishings" attribute specifies 'staple', then with 'single-document', Documents a and b are stapled together as a single Set with no regard to a new Media Sheet, while with 'single-document-new-sheet', Documents a and b are stapled together as a single Set but Document b starts on a new Media Sheet. With 'separate-documents-uncollated-copies' and 'separate-documents-collated-copies', Documents a and b are stapled separately.

Note: The value 'separate-documents-uncollated-copies' produces uncollated Media Sheets within a Set, e.g., when "copies" is '2' a two-Document Job will be printed as Media Sheets a(1), a(1), a(2), a(2), ... a(n), a(n), b(1), b(1), ..., b(n), b(n). All other values produce collated Media Sheets within a Set.

The relationship of this attribute and the other attributes that control Document processing is described in Appendix C.3.

5.2.5. copies (integer(1:MAX))

This RECOMMENDED attribute specifies the number of copies to be printed.

On many devices, the supported number of collated copies will be limited by the number of physical output bins on the device and can be different from the number of uncollated copies that can be supported.

Note: The effect of this attribute on Jobs with multiple Documents is controlled by the "multiple-document-handling" Job attribute (Section 5.2.4). The relationship of this attribute and the other attributes that control Document processing is described in Appendix C.3.

5.2.6. finishings (1setOf type2 enum)

This RECOMMENDED attribute identifies the finishing processes that the Printer uses for each copy of each printed Document in the Job. For Jobs with multiple Documents, the "multiple-document-handling" attribute determines what constitutes a "copy" for purposes of finishing.

Standard enum values defined in this document are listed in Table 10. The 'staple-xxx' values are specified with respect to the Document as if the Document were in portrait orientation with the origin of each Media Sheet at the top left corner. If the Document is actually in landscape or reverse-landscape orientation, the Client supplies the appropriate transformed value. For example, to position a staple in the upper left-hand corner of a landscape Document when held for reading, the Client supplies the 'staple-bottom-left' value, since landscape is defined as a +90 degree rotation of the image with respect to the media from portrait, i.e., counterclockwise. On the other hand, to position a staple in the upper left-hand corner of a reverse-landscape Document when held for reading, the Client supplies the 'staple-top-right' value, since reverse-landscape is defined as a -90 degree rotation of the image with respect to the media from portrait, i.e., clockwise.

The angle (vertical, horizontal, angled) of each staple with respect to the Document depends on the implementation, which can in turn depend on the value of the attribute.

Note: The effect of this attribute on Jobs with multiple Documents is controlled by the "multiple-document-handling" Job attribute (Section 5.2.4). The relationship of this attribute and the other attributes that control Document processing is described in Appendix C.3.

Note: The value of '3' ('none') has no effect when combined with any other values.

Note: The "finishings-col" attribute [PWG5100.1] is an alternative to the "finishings" attribute that allows the Client to specify finishing intent in greater detail.

Value	Symbolic Name and Description
'3'	'none': Perform no finishing.
'4'	'staple': Bind the Document(s) with one or more staples. The exact number and placement of the staples are site defined.
'5'	'punch': This value indicates that holes are required in the finished Document. The exact number and placement of the holes are site defined. The punch specification MAY be satisfied (in a site-specific and implementation-specific manner) either by drilling/punching or by substituting pre-drilled media.
'6'	'cover': This value is specified when it is desired to select a non-printed (or pre-printed) cover for the Document. This does not supplant the specification of a printed cover (on cover stock medium) by the Document itself.
'7'	'bind': This value indicates that a binding is to be applied to the Document; the type and placement of the binding are site defined.
'8'	'saddle-stitch': Bind the Document(s) with one or more staples (wire stitches) along the middle fold. The exact number and placement of the staples and the middle fold are implementation defined and/or site defined.

'9'	'edge-stitch': Bind the Document(s) with one or more staples (wire stitches) along one edge. The exact number and placement of the staples are implementation defined and/or site defined.
'10'-'19'	reserved for future generic finishing enum values.
'20'	'staple-top-left': Bind the Document(s) with one or more staples in the top left corner.
'21'	'staple-bottom-left': Bind the Document(s) with one or more staples in the bottom left corner.
'22'	'staple-top-right': Bind the Document(s) with one or more staples in the top right corner.
'23'	'staple-bottom-right': Bind the Document(s) with one or more staples in the bottom right corner.
'24'	'edge-stitch-left': Bind the Document(s) with one or more staples (wire stitches) along the left edge. The exact number and placement of the staples are implementation defined and/or site defined.
'25'	'edge-stitch-top': Bind the Document(s) with one or more staples (wire stitches) along the top edge. The exact number and placement of the staples are implementation defined and/or site defined.
'26'	'edge-stitch-right': Bind the Document(s) with one or more staples (wire stitches) along the right edge. The exact number and placement of the staples are implementation defined and/or site defined.
'27'	'edge-stitch-bottom': Bind the Document(s) with one or more staples (wire stitches) along the bottom edge. The exact number and placement of the staples are implementation defined and/or site defined.
'28'	'staple-dual-left': Bind the Document(s) with two staples (wire stitches) along the left edge, assuming a portrait Document (see above).

'29'	'staple-dual-top': Bind the Document(s) with two staples (wire stitches) along the top edge, assuming a portrait Document (see above).
'30'	'staple-dual-right': Bind the Document(s) with two staples (wire stitches) along the right edge, assuming a portrait Document (see above).
'31'	'staple-dual-bottom': Bind the Document(s) with two staples (wire stitches) along the bottom edge, assuming a portrait Document (see above).

Table 10: "finishings" Enum Values

5.2.7. page-ranges (1setOf rangeOfInteger(1:MAX))

This RECOMMENDED attribute identifies the range(s) of Input Pages that the Printer uses for each Set to be printed prior to imposition of those pages onto Impressions. Nothing is printed for any pages identified that do not exist in the Set/Document(s). Ranges MUST be in ascending order (1-3, 5-7, 15-19, etc.) and MUST NOT overlap so that a non-spooling Printer can process the Job in a single pass. If the ranges are not ascending or are overlapping, the Printer MUST reject the request and return the 'client-error-bad-request' status-code. The attribute is associated with Input Pages and not application-numbered pages such as the page numbers found in the headers and/or footers for certain word processing applications.

For Jobs with multiple Documents, the "multiple-document-handling" attribute determines what constitutes a Set for purposes of the specified page range(s). When "multiple-document-handling" is 'single-document', the Printer MUST apply each supplied page range once to the concatenation of the Input Pages. For example, if there are 8 Documents of 10 pages each, the page range '41-60' prints the pages in the 5th and 6th Documents as a single Document, and none of the pages of the other Documents are printed. When "multiple-document-handling" is 'separate-documents-uncollated-copies' or 'separate-documents-collated-copies', the Printer MUST apply each supplied page range repeatedly to each Document copy. For the same Job, the page range '1-3, 10-10' would print the first 3 pages and the 10th page of each of the 8 Documents in the Job, as 8 separate Sets.

"page-ranges-supported" is a boolean value indicating whether the Printer is capable of supporting the printing of page ranges. This capability can differ from one PDL to another. There is no "page-ranges-default" attribute. If the "page-ranges" attribute is not supplied by the Client, all pages of the Document are printed.

Note: In many cases, the Client supplies only those Input Pages that need to be printed in the Document data, and the "page-ranges" Job Template attribute is not used. However, Clients that submit already-generated Document data (either static content from some web site or previously submitted content the End User wishes to reprint) can use this attribute to print just a subset of the pages contained in the Document. In this case, if a "page-ranges" value of 'n-m' is specified, the first page to be printed will be page n. All subsequent pages of the Document will be printed through and including page m.

Note: The effect of this attribute on Jobs with multiple Documents is controlled by the "multiple-document-handling" Job attribute (Section 5.2.4). The relationship of this attribute and the other attributes that control Document processing is described in Appendix C.3.

5.2.8. sides (type2 keyword)

This RECOMMENDED attribute specifies how Impressions are placed upon the sides of a Media Sheet.

The standard 'keyword' values are:

- 'one-sided': imposes each consecutive Impression upon the same side of consecutive Media Sheets.

- 'two-sided-long-edge': imposes each consecutive pair of Impressions upon front and back sides of consecutive Media Sheets, such that the orientation of each pair of Impressions on the medium would be correct for the reader as if for binding on the long edge. This imposition is sometimes called 'duplex' or 'head-to-head'.

- 'two-sided-short-edge': imposes each consecutive pair of Impressions upon front and back sides of consecutive Media Sheets, such that the orientation of each pair of Impressions on the medium would be correct for the reader as if for binding on the short edge. This imposition is sometimes called 'tumble' or 'head-to-toe'.

Note: The effect of this attribute on Jobs with multiple Documents is controlled by the "multiple-document-handling" Job attribute (Section 5.2.4). The relationship of this attribute and the other attributes that control Document processing is described in Appendix C.3.

5.2.9. number-up (integer(1:MAX))

This attribute specifies the number of Input Pages to impose upon a single Impression. For example, if the value is:

- '1': the Printer MUST place one Input Page on a single Impression.
- '2': the Printer MUST place two Input Pages on a single Impression.
- '4': the Printer MUST place four Input Pages on a single Impression.

In all cases, the Printer MAY add some sort of translation, scaling, or rotation of Input Pages when imposing them.

Note: The effect of this attribute on Jobs with multiple Documents is controlled by the "multiple-document-handling" Job attribute (Section 5.2.4). The relationship of this attribute and the other attributes that control Document processing is described in Appendix C.3.

5.2.10. orientation-requested (type2 enum)

This RECOMMENDED attribute indicates the desired orientation for printed Input Pages; it does not describe the orientation of the Client-supplied Input Pages.

For some Document formats (such as 'application/postscript'), the desired orientation of the Input Pages is sometimes specified within the Document data. This information is generated by a Printer driver prior to the submission of the Print Job. Other Document formats such as 'text/plain' do not include the notion of desired orientation within the Document data. In the latter case, it is possible for the Printer to bind the desired orientation to the Document data after it has been submitted. Printers MAY only support "orientation-requested" for some Document formats (e.g., 'text/plain' or 'text/html') but not others (e.g., 'application/postscript'). This is no different than any other Job Template attribute, since Section 5.2, item 1, points out that a Printer can support or not support any Job Template attribute based on the Document format

supplied by the Client. However, a special mention is made here, since it is very likely that a Printer will support "orientation-requested" for only a subset of the supported Document formats.

Standard enum values are listed in Table 11.

Note: The effect of this attribute on Jobs with multiple Documents is controlled by the "multiple-document-handling" Job attribute (Section 5.2.4). The relationship of this attribute and the other attributes that control Document processing is described in Appendix C.3.

Value	Symbolic Name and Description
'3'	'portrait': The content will be imaged across the short edge of the medium.
'4'	'landscape': The content will be imaged across the long edge of the medium. Landscape is defined to be a rotation of the Input Page to be imaged by +90 degrees with respect to the medium (i.e., counterclockwise) from the portrait orientation. Note: The +90 direction was chosen because simple finishing on the long edge is the same edge whether portrait or landscape.
'5'	'reverse-landscape': The content will be imaged across the long edge of the medium. Reverse-landscape is defined to be a rotation of the Input Page to be imaged by -90 degrees with respect to the medium (i.e., clockwise) from the portrait orientation. Note: The 'reverse-landscape' value was added because some applications rotate landscape -90 degrees from portrait, rather than +90 degrees.
'6'	'reverse-portrait': The content will be imaged across the short edge of the medium. Reverse-portrait is defined to be a rotation of the Input Page to be imaged by 180 degrees with respect to the medium from the portrait orientation. Note: The 'reverse-portrait' value was added for use with the "finishings" attribute in cases where the opposite edge is desired for finishing a portrait Document on simple finishing devices that have only one finishing position. Thus, a 'text'/plain' portrait Document can be stapled "on the right" by a simple finishing device, as is common use with some Middle Eastern languages such as Hebrew.

Table 11: "orientation-requested" Enum Values

5.2.11. media (type2 keyword | name(MAX))

This RECOMMENDED attribute identifies the medium that the Printer uses for all Impressions of the Job.

The values for "media" historically have included medium names, medium sizes, input trays, and electronic forms so that one attribute specifies the media. However, the Client SHOULD only use the media attribute to specify medium sizes using PWG Media Standardized Names [PWG5101.1].

If a Printer supports a medium name as a value of this attribute, such a medium name implicitly selects an input tray that contains the specified medium. If a Printer supports a medium size as a value of this attribute, such a medium size implicitly selects a medium name that in turn implicitly selects an input tray that contains the medium with the specified size. If a Printer supports an input tray as the value of this attribute, such an input tray implicitly selects the medium that is in that input tray at the time the Job prints. This case includes manual-feed input trays. If a Printer supports an electronic form as the value of this attribute, such an electronic form implicitly selects a medium name that in turn implicitly selects an input tray that contains the medium specified by the electronic form. The electronic form also implicitly selects an image that the Printer MUST merge with the Document data as it prints each page.

PWG Media Standardized Names [PWG5101.1] SHOULD be used. Legacy 'keyword' values are taken from ISO DPA [ISO10175], the Printer MIB [RFC3805], and ASME-Y14.1M [ASME-Y14.1M]. An Administrator MAY define additional values using the 'name' or 'keyword' attribute syntax, depending on implementation.

There is also an additional Printer attribute named "media-ready", which differs from "media-supported" in that legal values only include the subset of "media-supported" values that are physically loaded and ready for printing with no Operator intervention required.

The relationship of this attribute and the other attributes that control Document processing is described in Appendix C.3.

Note: If supported by the Printer, Clients MAY use the alternative "media-col" attribute [PWG5100.3] [PWG5100.13] to specify medium requirements in greater detail.

5.2.12. printer-resolution (resolution)

This RECOMMENDED attribute identifies the output resolution that the Printer uses for the Job.

Note: This attribute and the "print-quality" attribute (Section 5.2.13) are both used to specify the overall output quality of the Job. If a Client specifies conflicting "printer-resolution" and "print-quality" values, Printers SHOULD use the "print-quality" value.

5.2.13. print-quality (type2 enum)

This RECOMMENDED attribute specifies the print quality that the Printer uses for the Job.

The standard enum values are listed in Table 12.

Note: This attribute and the "printer-resolution" attribute (Section 5.2.12) are both used to specify the overall output quality of the Job. If a Client specifies conflicting "printer-resolution" and "print-quality" values, Printers SHOULD use the "print-quality" value.

Value	Symbolic Name and Description
'3'	'draft': lowest quality available on the Printer
'4'	'normal': normal or intermediate quality on the Printer
'5'	'high': highest quality available on the Printer

Table 12: "print-quality" Enum Values

5.3. Job Description and Status Attributes

The attributes in this section form the attribute group called "job-description". Tables 13 and 14 summarize these attributes. The third column of each table indicates whether the attribute is a REQUIRED attribute that MUST be supported by Printers. If it is not indicated as REQUIRED, then it is OPTIONAL. The maximum size in octets for 'text' and 'name' attributes is indicated in parentheses.

Attribute	Syntax	REQUIRED?
job-impressions	integer(0:MAX)	
job-k-octets	integer(0:MAX)	
job-media-sheets	integer(1:MAX)	
job-name	name(MAX)	REQUIRED

Table 13: Job Description Attributes (READ-WRITE)

Attribute	Syntax	REQUIRED?
attributes-charset	charset	REQUIRED
attributes-natural-language	naturalLanguage	REQUIRED
date-time-at-completed	dateTime|unknown|no-value	
date-time-at-creation	dateTime|unknown	
date-time-at-processing	dateTime|unknown|no-value	
job-detailed-status-messages	1setOf text(MAX)	
job-document-access-errors	1setOf text(MAX)	
job-id	integer(1:MAX)	REQUIRED
job-impressions-completed	integer(0:MAX)	
job-k-octets-processed	integer(0:MAX)	
job-media-sheets-completed	integer(0:MAX)	
job-message-from-operator	text(127)	
job-more-info	uri	
job-originating-user-name	name(MAX)	REQUIRED

job-printer-up-time	integer(1:MAX)	REQUIRED
job-printer-uri	uri	REQUIRED
job-state	type1 enum	REQUIRED
job-state-message	text(MAX)	
job-state-reasons	1setOf type2 keyword	REQUIRED
job-uri	uri	REQUIRED
number-of-documents	integer(0:MAX)	
number-of-intervening-jobs	integer(0:MAX)	
output-device-assigned	name(127)	
time-at-completed	integer(MIN:MAX)	REQUIRED
time-at-creation	integer(MIN:MAX)	REQUIRED
time-at-processing	integer(MIN:MAX)	REQUIRED

Table 14: Job Status Attributes (READ-ONLY)

5.3.1. job-id (integer(1:MAX))

This REQUIRED attribute contains the ID of the Job. The Printer, on receipt of a new Job, generates an ID that identifies the new Job on that Printer. The Printer returns the value of the "job-id" attribute as part of the response to a Job Creation request.

For a description of this attribute and its relationship to the "job-uri" and "job-printer-uri" attributes, see the discussion in Section 3.4 ("Object Identity").

5.3.2. job-uri (uri)

This REQUIRED attribute contains the URI for the Job. The Printer, on receipt of a new Job, generates a URI that identifies the new Job. The Printer returns the value of the "job-uri" attribute as part of the response to a Job Creation request. The precise format of a Job URI is implementation dependent [RFC3510] [RFC7472]. If the Printer supports more than one URI and there is some relationship between the newly formed Job URI and the Printer’s URI, the Printer uses the

Printer URI supplied by the Client in the Job Creation request. For example, if the Job Creation request comes in over a secure channel, the new Job URI MUST use the same secure channel. This can be guaranteed because the Printer is responsible for generating the Job URI and the Printer is aware of its security configuration and policy as well as the Printer URI used in the Job Creation request.

For a description of this attribute and its relationship to the "job-id" and "job-printer-uri" attributes, see the discussion in Section 3.4 ("Object Identity").

5.3.3. job-printer-uri (uri)

This REQUIRED attribute identifies the Printer that created this Job. When a Printer creates a Job, it populates this attribute with the Printer URI that was used in the Job Creation request. This attribute permits a Client to identify the Printer that created this Job when only the Job's URI is available to the Client. The Client queries the creating Printer to determine which languages, charsets, and operations are supported for this Job.

For a description of this attribute and its relationship to the "job-uri" and "job-id" attributes, see the discussion in Section 3.4 ("Object Identity").

5.3.4. job-more-info (uri)

Similar to "printer-more-info", this attribute contains the URI referencing some resource with more information about this Job, perhaps an HTML page containing status information about the Job.

5.3.5. job-name (name(MAX))

This REQUIRED attribute is the name of the Job. It is a name that is more user friendly than the "job-uri" or "job-id" attribute values. It does not need to be unique between Jobs. The Job's "job-name" attribute is set to the value supplied by the Client in the "job-name" operation attribute in the Job Creation request (see Section 4.2.1.1). If, however, the "job-name" operation attribute is not supplied by the Client in the Job Creation request, the Printer, on creation of the Job, MUST generate a name. The Printer SHOULD generate the value of the Job's "job-name" attribute from the first of the following sources that produces a value: (1) the "document-name" operation attribute of the first (or only) Document, (2) the "document-URI" attribute of the first (or only) Document, or (3) any other piece of Job-specific and/or Document data.

5.3.6. job-originating-user-name (name(MAX))

This REQUIRED attribute contains the name of the End User that submitted the Print Job. The Printer sets this attribute to the most authenticated printable name that it can obtain from the authentication service over which the IPP operation was received. Only if such a name is not available does the Printer use the value supplied by the Client in the "requesting-user-name" operation attribute of the Job Creation request (see Sections 5.4.2, 5.4.3, and 9).

Note: The Printer needs to keep an internal originating user ID of some form, typically as a credential of a principal, with the Job. Since such an internal attribute is implementation dependent and not of interest to Clients, it is not specified as a Job attribute. This originating user ID is used for authorization checks (if any) on all subsequent operations.

5.3.7. job-state (type1 enum)

This REQUIRED attribute identifies the current state of the Job. Even though IPP defines seven values for Job states (plus the out-of-band 'unknown' value -- see Section 5.1), implementations only need to support those states that are appropriate for the particular implementation. In other words, a Printer supports only those Job states implemented by the Output Device and available to the Printer implementation.

Standard enum values are listed in Table 15.

The final value for this attribute MUST be one of the following -- 'completed', 'canceled', or 'aborted' -- before the Printer removes the Job altogether. The length of time that Jobs remain in the 'canceled', 'aborted', and 'completed' states depends on implementation. See Section 5.3.7.2.

Figure 3 shows the normal Job state transitions. Normally, a Job progresses from left to right. Other state transitions are unlikely but are not forbidden. Not shown are the transitions to the 'canceled' state from the 'pending', 'pending-held', and 'processing-stopped' states.

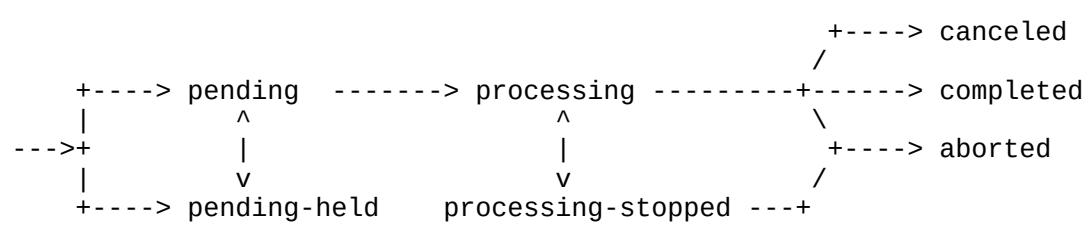


Figure 3: IPP Job Life Cycle

Jobs reach one of the three terminal states -- 'completed', 'canceled', or 'aborted' -- after the Jobs have completed all activity, including stacking output media, and all Job Status attributes have reached their final values for the Job.

Values	Symbolic Name and Description
'3'	'pending': The Job is a candidate to start processing but is not yet processing.
'4'	'pending-held': The Job is not a candidate for processing for any number of reasons but will return to the 'pending' state as soon as the reasons are no longer present. The Job's "job-state-reasons" attribute MUST indicate why the Job is no longer a candidate for processing.
'5'	'processing': One or more of the following: (1) the Job is using, or is attempting to use, one or more purely software processes that are analyzing, creating, or interpreting a PDL, etc.; (2) the Job is using, or is attempting to use, one or more hardware devices that are interpreting a PDL; making marks on a medium; and/or performing finishing, such as stapling, etc.; (3) the Printer has made the Job ready for printing, but the Output Device is not yet printing it, either because the Job hasn't reached the Output Device or because the Job is queued in the Output Device or some other spooler, waiting for the Output Device to print it. When the Job is in the 'processing' state, the entire Job state includes the detailed status represented in the Printer's "printer-state", "printer-state-reasons", and "printer-state-message" attributes. Implementations MAY include additional values in the Job's "job-state-reasons" attribute to indicate the progress of the Job, such as adding the 'job-printing' value to indicate when the Output Device is actually making marks on paper and/or the 'processing-to-stop-point' value to indicate that the Printer is in the process of canceling or aborting the Job.

'6'	'processing-stopped': The Job has stopped while processing for any number of reasons and will return to the 'processing' state as soon as the reasons are no longer present. The Job's "job-state-reasons" attribute MAY indicate why the Job has stopped processing. For example, if the Output Device is stopped, the 'printer-stopped' value MAY be included in the Job's "job-state-reasons" attribute. Note: When an Output Device is stopped, the device usually indicates its condition in human-readable form locally at the device. A Client can obtain more complete device status remotely by querying the Printer's "printer-state", "printer-state-reasons", and "printer-state-message" attributes.
'7'	'canceled': The Job has been canceled by a Cancel-Job operation, and the Printer has completed canceling the Job. All Job Status attributes have reached their final values for the Job. While the Printer is canceling the Job, the Job remains in its current state, but the Job's "job-state-reasons" attribute SHOULD contain the 'processing-to-stop-point' value and one of the 'canceled-by-user', 'canceled-by-operator', or 'canceled-at-device' values. When the Job moves to the 'canceled' state, the 'processing-to-stop-point' value, if present, MUST be removed, but 'canceled-by-xxx', if present, MUST remain.
'8'	'aborted': The Job has been aborted by the system, usually while the Job was in the 'processing' or 'processing-stopped' state, and the Printer has completed aborting the Job; all Job Status attributes have reached their final values for the Job. While the Printer is aborting the Job, the Job remains in its current state, but the Job's "job-state-reasons" attribute SHOULD contain the 'processing-to-stop-point' and 'aborted-by-system' values. When the Job moves to the 'aborted' state, the 'processing-to-stop-point' value, if present, MUST be removed, but the 'aborted-by-system' value, if present, MUST remain.

'9'	'completed': The Job has completed successfully or with warnings or errors after processing, all of the Job Media Sheets have been successfully stacked in the appropriate output bin(s), and all Job Status attributes have reached their final values for the Job. The Job's "job-state-reasons" attribute SHOULD contain one of the 'completed-successfully', 'completed-with-warnings', or 'completed-with-errors' values.

Table 15: "job-state" Enum Values

5.3.7.1. Forwarding Servers

As with all other IPP attributes, if the implementation cannot determine the correct value for this attribute, it SHOULD respond with the out-of-band 'unknown' value (see Section 5.1) rather than try to guess at some possibly incorrect value and confuse the End User about the state of the Job. For example, if the implementation is just a gateway into some printing system from which it can normally get status, but temporarily is unable, then the implementation should return the 'unknown' value. However, if the implementation is a gateway to a printing system that never provides detailed status about the Print Job, the implementation MAY set the IPP Job's state to 'completed', provided that it also sets the 'queued-in-device' value in the Job's "job-state-reasons" attribute (see Section 5.3.8).

5.3.7.2. Partitioning of Job States

This section describes the partitioning of the seven Job states into phases: Job Not Completed, Job Retention, Job History, and Job Removal. This section also explains the 'job-restartable' value of the "job-state-reasons" Job Status attribute for use with the Restart-Job and Resubmit-Job [PWG5100.11] operations.

Job Not Completed: When a Job is in the 'pending', 'pending-held', 'processing', or 'processing-stopped' state, the Job is not completed.

Job Retention: When a Job enters one of the three terminal Job states -- 'completed', 'canceled', or 'aborted' -- the IPP Printer MAY "retain" the Job in a restartable condition for an implementation-defined time period. This time period MAY be zero seconds and MAY depend on the terminal Job state. This phase is called "Job Retention". While in the Job Retention phase, the Job's Document data is retained and a Client can restart the Job using the

Restart-Job operation. If the Printer supports the Restart-Job or Resubmit-Job operation, then it SHOULD indicate that the Job is restartable by adding the 'job-restartable' value to the Job's "job-state-reasons" attribute (see Section 5.3.8) during the Job Retention phase.

Job History: After the Job Retention phase expires for a Job, the Printer deletes the Document data for the Job and the Job becomes part of the Job History. The Printer MAY also delete any number of the Job attributes. Since the Job is no longer restartable, the Printer MUST remove the 'job-restartable' value from the Job's "job-state-reasons" attribute, if present. Printers SHOULD keep the Job in the Job History phase for at least 60 seconds to allow Clients to discover the final disposition of the Job.

Job Removal: After the Job has remained in the Job History for an implementation-defined time, such as when the number of Jobs exceeds a fixed number or after a fixed time period (which MAY be zero seconds), the IPP Printer removes the Job from the system.

Using the Get-Jobs operation and supplying the 'not-completed' value for the "which-jobs" operation attribute, a Client is requesting Jobs in the Job Not Completed phase. Using the Get-Jobs operation and supplying the 'completed' value for the "which-jobs" operation attribute, a Client is requesting Jobs in the Job Retention and Job History phases. Using the Get-Job-Attributes operation, a Client is requesting a Job in any phase except Job Removal. After Job Removal, the Get-Job-Attributes and Get-Jobs operations no longer are capable of returning any information about a Job.

5.3.8. job-state-reasons (1setOf type2 keyword)

This REQUIRED attribute provides additional information about the Job's current state, i.e., information that augments the value of the Job's "job-state" attribute.

These values MAY be used with any Job state or states for which the reason makes sense. Some of these value definitions indicate conformance requirements; the rest are OPTIONAL. Furthermore, when implemented, the Printer MUST return these values when the reason applies and MUST NOT return them when the reason no longer applies, whether the value of the Job's "job-state" attribute changed or not. When the Job does not have any reasons for being in its current state, the value of the Job's "job-state-reasons" attribute MUST be 'none'.

Note: While values cannot be added to the "job-state" attribute without impacting deployed Clients that take actions upon receiving "job-state" values, it is the intent that additional "job-state-reasons" values can be defined and registered without impacting such deployed Clients. In other words, the "job-state-reasons" attribute is intended to be extensible.

The following standard 'keyword' values are defined. For ease of understanding, the values are presented in the order in which the reasons are likely to occur (if implemented):

- 'none': There are no reasons for the Job's current state. This state reason is semantically equivalent to "job-state-reasons" without any value and MUST be used when there is no other value, since the '1setOf' attribute syntax requires at least one value.

- 'job-incoming': Either (1) the Printer has accepted the Create-Job operation and is expecting additional Send-Document and/or Send-URI operations or (2) the Printer is retrieving/accepting Document data as a result of a Print-Job, Print-URI, Send-Document, or Send-URI operation.

- 'job-data-insufficient': The Create-Job operation has been accepted by the Printer, but the Printer is expecting additional Document data before it can move the Job into the 'processing' state. If a Printer starts processing before it has received all data, the Printer removes the 'job-data-insufficient' reason, but the 'job-incoming' reason remains. If a Printer starts processing after it has received all data, the Printer removes the 'job-data-insufficient' reason and the 'job-incoming' reason at the same time.

- 'document-access-error': After accepting a Print-URI or Send-URI request, the Printer could not access one or more Documents passed by reference. This reason is intended to cover any file access problem, including 'file does not exist' and 'access denied' because of an access control problem. The Printer MAY also indicate the Document access error using the "job-document-access-errors" Job Status attribute (see Section 5.3.11). The Printer can (1) abort the Job and move the Job to the 'aborted' Job state or (2) print all Documents that are accessible and move the Job to the 'completed' Job state with the 'completed-with-errors' value in the Job's "job-state-reasons" attribute. This value SHOULD be supported if the Print-URI or Send-URI operations are supported.

- 'submission-interrupted': The Job was not completely submitted for some unforeseen reason, such as (1) the Printer has crashed before the Job was closed by the Client, (2) the Printer or the Document transfer method has crashed in some non-recoverable way before the Document data was entirely transferred to the Printer, or (3) the Client crashed or failed to close the Job before the time-out period. See Section 5.4.31.

- 'job-outgoing': The Printer is transmitting the Job to the Output Device.

- 'job-hold-until-specified': The value of the Job's "job-hold-until" attribute was specified with a time period that is still in the future. The Job MUST NOT be a candidate for processing until this reason is removed and there are no other reasons to hold the Job. This value SHOULD be supported if the "job-hold-until" Job Template attribute is supported.

- 'resources-are-not-ready': At least one of the resources needed by the Job, such as media, fonts, resource objects, etc., is not ready on any of the physical Output Devices for which the Job is a candidate. This condition MAY be detected when the Job is accepted, or subsequently while the Job is pending or processing, depending on implementation. The Job can remain in its current state or be moved to the 'pending-held' state, depending on implementation and/or Job scheduling policy.

- 'printer-stopped-partly': The value of the Printer's "printer-state-reasons" attribute contains the value 'stopped-partly'.

- 'printer-stopped': The value of the Printer's "printer-state" attribute is 'stopped'.

- 'job-interpreting': The Job is in the 'processing' state, but, more specifically, the Printer is interpreting the Document data.

- 'job-queued': The Job is in the 'processing' state, but, more specifically, the Printer has queued the Document data.

- 'job-transforming': The Job is in the 'processing' state, but, more specifically, the Printer is interpreting Document data and producing another electronic representation.

- 'job-queued-for-marker': The Job is in any of the 'pending-held', 'pending', or 'processing' states, but, more specifically, the Printer has completed enough processing of the Document to be able to start marking, and the Job is waiting for the marker. Systems

that require human intervention to release Jobs using the Release-Job operation put the Job into the 'pending-held' Job state. Systems that automatically select a Job to use the marker put the Job into the 'pending' Job state or keep the Job in the 'processing' Job state while waiting for the marker, depending on implementation. All implementations put the Job into the 'processing' state when marking does begin.

- 'job-printing': The Output Device is marking media. This value is useful for Printers that spend a great deal of time processing (1) when no marking is happening and they want to show that marking is now happening or (2) when the Job is in the process of being canceled or aborted while the Job remains in the 'processing' state, but the marking has not yet stopped so that Impression or sheet counts are still increasing for the Job.

- 'job-canceled-by-user': The Job was canceled by the owner of the Job using the Cancel-Job request, i.e., by a user whose authenticated identity is the same as the value of the originating user that created the Job, or by some other authorized End User, such as a member of the Job owner's security group. This value SHOULD be supported.

- 'job-canceled-by-operator': The Job was canceled by the Operator using the Cancel-Job request, i.e., by a user who has been authenticated as having Operator privileges (whether local or remote). If the security policy is to allow anyone to cancel anyone's Job, then this value can be used when the Job is canceled by other than the owner of the Job. For such a security policy, in effect, everyone is an Operator as far as canceling Jobs with IPP is concerned. This value SHOULD be supported if the implementation permits canceling by other than the owner of the Job.

- 'job-canceled-at-device': The Job was canceled by an unidentified local user, i.e., a user at a console at the device. This value SHOULD be supported if the implementation supports canceling Jobs at the console.

- 'aborted-by-system': The Job (1) is in the process of being aborted, (2) has been aborted by the system and placed in the 'aborted' state, or (3) has been aborted by the system and placed in the 'pending-held' state, so that a user or Operator can manually try the Job again. This value SHOULD be supported.

- 'unsupported-compression': The Job was aborted by the system because the Printer determined, while attempting to decompress the Document data, that the compression algorithm is actually not among those supported by the Printer. This value MUST be supported, since "compression" is a REQUIRED operation attribute.

- 'compression-error': The Job was aborted by the system because the Printer encountered an error in the Document data while decompressing it. If the Printer posts this reason, the Document data has already passed any tests that would have led to the 'unsupported-compression' "job-state-reasons" value.

- 'unsupported-document-format': The Job was aborted by the system because the Document data's "document-format" attribute is not among those supported by the Printer. If the Client specifies "document-format" as 'application/octet-stream', the Printer MAY abort the Job and post this reason even though the "document-format" value is among the values of the Printer's "document-format-supported" Printer attribute but not among the auto-sensed Document formats. This value MUST be supported, since "document-format" is a REQUIRED operation attribute.

- 'document-format-error': The Job was aborted by the system because the Printer encountered an error in the Document data while processing it. If the Printer posts this reason, the Document data has already passed any tests that would have led to the 'unsupported-document-format' "job-state-reasons" value.

- 'processing-to-stop-point': The requester has issued a Cancel-Job operation or the Printer has aborted the Job, but the Printer is still performing some actions on the Job until a specified stop point occurs or Job termination/cleanup is completed.

  If the implementation requires some measurable time to cancel the Job in the 'processing' or 'processing-stopped' Job state, the Printer MUST use this value to indicate that the Printer is still performing some actions on the Job while the Job remains in the 'processing' or 'processing-stopped' state. Once at the stop point, the Printer moves the Job from the 'processing' state to the 'canceled' or 'aborted' Job state.

- 'service-off-line': The Printer is offline and accepting no Jobs. All 'pending' Jobs are put into the 'pending-held' state. This situation could be true if the service's or Document transform's input is impaired or broken.

- 'job-completed-successfully': The Job completed successfully. This value SHOULD be supported.

- 'job-completed-with-warnings': The Job completed with warnings. This value SHOULD be supported if the implementation detects warnings.

- 'job-completed-with-errors': The Job completed with errors (and possibly warnings too). This value SHOULD be supported if the implementation detects errors.

- 'job-restartable': This Job is retained (see Section 5.3.7.2) and is currently able to be restarted using the Restart-Job (see Section 4.3.7) or Resubmit-Job [PWG5100.11] operation. If 'job-restartable' is a value of the Job's "job-state-reasons" attribute, then the Printer MUST accept a Restart-Job operation for that Job. This value SHOULD be supported if the Restart-Job operation is supported.

- 'queued-in-device': The Job has been forwarded to a device or print system that is unable to send back status. The Printer sets the Job's "job-state" attribute to 'completed' and adds the 'queued-in-device' value to the Job's "job-state-reasons" attribute to indicate that the Printer has no additional information about the Job and never will have any better information. See Section 5.3.7.1.

5.3.9. job-state-message (text(MAX))

This RECOMMENDED attribute specifies information about the "job-state" and "job-state-reasons" attributes in human-readable text. If the Printer supports this attribute, the Printer MUST be able to generate this message in any of the natural languages identified by the Printer's "generated-natural-language-supported" attribute (see the "attributes-natural-language" operation attribute specified in Section 4.1.4.1).

The value SHOULD NOT contain additional information not contained in the values of the "job-state" and "job-state-reasons" attributes, such as interpreter error information. Otherwise, application programs might attempt to parse the (localized) text. For such additional information, such as interpreter errors for application program consumption or specific Document access errors, new attributes with 'keyword' values need to be developed and registered.

5.3.10. job-detailed-status-messages (1setOf text(MAX))

This attribute specifies additional detailed and technical information about the Job. The Printer SHOULD localize the message, unless such localization would obscure the technical meaning of the message. Clients MUST NOT attempt to parse the value of this attribute. See "job-document-access-errors" (Section 5.3.11) for additional errors that a program can process.

5.3.11. job-document-access-errors (1setOf text(MAX))

This attribute provides additional information about each Document access error for this Job encountered by the Printer after it returned a response to the Print-URI or Send-URI operation and subsequently attempted to access document(s) supplied in the Print-URI or Send-URI operation. For errors in the protocol that is identified by the URI scheme in the "document-uri" operation attribute, such as 'http:' or 'ftp:', the error code is returned in parentheses, followed by the URI. For example:

(404) http://www.example.com/filename.pdf

Most Internet protocols use decimal error codes (unlike IPP), so the ASCII error code representation is in decimal.

5.3.12. number-of-documents (integer(0:MAX))

This attribute indicates the number of Documents in the Job, i.e., the number of Send-Document, Send-URI, Print-Job, or Print-URI operations that the Printer has accepted for this Job, regardless of whether the Document data has reached the Printer.

Implementations supporting the RECOMMENDED Create-Job/Send-Document/ Send-URI operations SHOULD support this attribute so that Clients can query the number of Documents in each Job.

5.3.13. output-device-assigned (name(127))

This attribute identifies the Output Device to which the Printer has assigned this Job. If an Output Device implements an embedded Printer, the Printer SHOULD set this attribute. If a print server implements a Printer, the value MAY be empty (zero-length string) or not returned until the Printer assigns an Output Device to the Job. This attribute is particularly useful when a single Printer supports multiple devices (so-called "fan-out" -- see Section 3.1).

5.3.14. Event Time Job Status Attributes

This section defines the Job Status attributes that indicate the time at which certain events occur for a Job. If the Job event has not yet occurred, then the Printer MUST return the 'no-value' out-of-band value (see the beginning of Section 5.1). The "time-at-xxx (integer)" attributes represent time as an 'integer' representing the number of seconds since the device was powered up (informally called "time ticks"). The "date-time-at-xxx (dateTime)" attributes represent time as 'dateTime' representing date and time (including an offset from UTC).

In order to populate these attributes, the Printer copies the value(s) of the following Printer Status attributes at the time the event occurs:

1. the value in the Printer's "printer-up-time" attribute for the "time-at-xxx (integer)" attributes.

2. the value in the Printer's "printer-current-time" attribute for the "date-time-at-xxx (dateTime)" attributes.

If the Printer resets its "printer-up-time" attribute to 1 on power-up (see Section 5.4.29) and has persistent Jobs, then it MUST change all of those Jobs' "time-at-xxx (integer)" (time tick) Job attributes whose events have occurred either to:

1. 0 to indicate that the event happened before the most recent power-up, or

2. the negative of the number of seconds before the most recent power-up that the event took place, if the Printer knows the exact number of seconds.

If a Client queries a "time-at-xxx (integer)" time tick Job attribute and finds the value to be 0 or negative, the Client MUST assume that the event occurred in some life other than the Printer's current life.

Note: A Printer does not change the values of any "date-time-at-xxx (dateTime)" Job attributes on power-up.

5.3.14.1. time-at-creation (integer(MIN:MAX))

This REQUIRED attribute indicates the time at which the Job was created.

5.3.14.2. time-at-processing (integer(MIN:MAX))

This REQUIRED attribute indicates the time at which the Job first began processing after the Job Creation request or the most recent Restart-Job operation. The out-of-band 'no-value' value is returned if the Job has not yet been in the 'processing' state (see the beginning of Section 5.1).

5.3.14.3. time-at-completed (integer(MIN:MAX))

This REQUIRED attribute indicates the time at which the Job entered a Terminating State ('completed', 'canceled', or 'aborted'). The out-of-band 'no-value' value is returned if the Job has not yet completed, been canceled, or aborted (see the beginning of Section 5.1).

5.3.14.4. job-printer-up-time (integer(1:MAX))

This REQUIRED Job Status attribute indicates the amount of time (in seconds) that the Printer implementation has been up and running. This attribute is an alias for the "printer-up-time" Printer Status attribute (see Section 5.4.29).

A Client MAY request this attribute in a Get-Job-Attributes or Get-Jobs request and use the value returned in combination with other requested Event Time Job Status attributes in order to display time attributes to a user. The difference between this attribute and the 'integer' value of a "time-at-xxx" attribute is the number of seconds ago that the "time-at-xxx" event occurred. A Client can compute the wall-clock time at which the "time-at-xxx" event occurred by subtracting this difference from the Client's wall-clock time.

5.3.14.5. date-time-at-creation (dateTime|unknown)

This RECOMMENDED attribute indicates the date and time at which the Job was created.

5.3.14.6. date-time-at-processing (dateTime|unknown|no-value)

This RECOMMENDED attribute indicates the date and time at which the Job first began processing after the Job Creation request or the most recent Restart-Job operation.

5.3.14.7. date-time-at-completed (dateTime|unknown|no-value)

This RECOMMENDED attribute indicates the date and time at which the Job entered a Terminating State ('completed', 'canceled', or 'aborted').

5.3.15. number-of-intervening-jobs (integer(0:MAX))

This attribute indicates the number of Jobs that are "ahead" of this Job in the relative chronological order of expected time to complete (i.e., the current scheduled order). For efficiency, it is only necessary to calculate this value when an operation is performed that requests this attribute.

5.3.16. job-message-from-operator (text(127))

This attribute provides a message from an Operator, Administrator, or "intelligent" process to indicate to the End User the reasons for modification or other management action taken on a Job.

5.3.17. Job Size Attributes

This subsection defines Job attributes that describe the size of the Job. These attributes are not intended to be counters; they are intended to be useful routing and scheduling information if known. For these attributes, the Printer can try to compute the value if it is not supplied in the Job Creation request. Even if the Client does supply a value for these three attributes in the Job Creation request, the Printer MAY choose to change the value if the Printer is able to compute a value that is more accurate than the Client-supplied value. The Printer can determine the correct value for these attributes either right at Job submission time or at any later point in time.

5.3.17.1. job-k-octets (integer(0:MAX))

This attribute specifies the total size of the Document(s) in K octets, i.e., in units of 1024 octets requested to be processed in the Job. The value MUST be rounded up, so that a Job between 1 and 1024 octets MUST be indicated as being 1, 1025 to 2048 MUST be 2, etc.

This value MUST NOT include the multiplicative factors contributed by the number of copies specified by the "copies" attribute, independent of whether the device can process multiple copies without making multiple passes over the Job or Document data and independent of whether the output is collated or not. Thus, the value is independent of the implementation and indicates the size of the Document(s) measured in K octets independent of the number of copies.

This value also MUST NOT include the multiplicative factor due to a copies instruction embedded in the Document data. If the Document data actually includes replications of the Document data, this value will include such replication. In other words, this value is always the size of the source Document data, rather than a measure of the hardcopy output to be produced.

5.3.17.2. job-impressions (integer(0:MAX))

This RECOMMENDED attribute specifies the total size in number of Impressions of the Document(s) being submitted (see the definition of "Impression" in Section 2.3.4).

As with "job-k-octets", this value MUST NOT include the multiplicative factors contributed by the number of copies specified by the "copies" attribute, independent of whether the device can process multiple copies without making multiple passes over the Job or Document data and independent of whether the output is collated or not. Thus, the value is independent of the implementation and reflects the size of the Document(s) measured in Impressions independent of the number of copies.

As with "job-k-octets", this value also MUST NOT include the multiplicative factor due to a copies instruction embedded in the Document data. If the Document data actually includes replications of the Document data, this value will include such replication. In other words, this value is always the number of Impressions in the source Document data, rather than a measure of the number of Impressions to be produced by the Job.

5.3.17.3. job-media-sheets (integer(1:MAX))

This RECOMMENDED attribute specifies the total number of Media Sheets to be produced for this Job.

Unlike the "job-k-octets" and the "job-impressions" attributes, this value MUST include the multiplicative factors contributed by the number of copies specified by the "copies" attribute and a 'number of copies' instruction embedded in the Document data, if any. This difference allows the Administrator to control the lower and upper bounds of both (1) the size of the Document(s) with "job-k-octets-supported" and "job-impressions-supported" and (2) the size of the Job with "job-media-sheets-supported".

5.3.18. Job Progress Attributes

This subsection defines Job attributes that describe the progress of the Job. These attributes are intended to be counters. That is, the values for a Job that has not started processing MUST be 0. When the Job's "job-state" is 'processing' or 'processing-stopped', this value is intended to contain the amount of the Job that has been processed to the time at which the attributes are requested. When the Job enters the 'completed', 'canceled', or 'aborted' states, these values are the final values for the Job.

5.3.18.1. job-k-octets-processed (integer(0:MAX))

This attribute specifies the total number of octets processed in K octets, i.e., in units of 1024 octets so far. The value MUST be rounded up, so that a Job between 1 and 1024 octets inclusive MUST be indicated as being 1, 1025 to 2048 inclusive MUST be 2, etc.

For implementations where multiple copies are produced by the interpreter with only a single pass over the data, the final value MUST be equal to the value of the "job-k-octets" attribute. For implementations where multiple copies are produced by the interpreter by processing the data for each copy, the final value MUST be a multiple of the value of the "job-k-octets" attribute.

5.3.18.2. job-impressions-completed (integer(0:MAX))

This RECOMMENDED attribute specifies the number of Impressions completed for the Job so far. For printing devices, the Impressions completed includes interpreting, marking, and stacking the output.

5.3.18.3. job-media-sheets-completed (integer(0:MAX))

This RECOMMENDED Job attribute specifies the number of Media Sheets that have been marked and stacked for the entire Job so far, whether those sheets have been processed on one side or on both.

5.3.19. attributes-charset (charset)

This REQUIRED attribute is populated using the value in the Client-supplied "attributes-charset" attribute in the Job Creation request. It identifies the charset (coded character set and encoding method) used by any Job attributes with attribute syntaxes 'text' and 'name' that were supplied by the Client in the Job Creation request. See Section 4.1.4 for a complete description of the "attributes-charset" operation attribute.

This attribute does not indicate the charset in which the 'text' and 'name' values are stored internally in the Job. The internal charset is implementation defined. The Printer MUST convert from whatever the internal charset is to that being requested in an operation as specified in Section 4.1.4.

5.3.20. attributes-natural-language (naturalLanguage)

This REQUIRED attribute is populated using the value in the Client-supplied "attributes-natural-language" attribute in the Job Creation request. It identifies the natural language used for any Job attributes with attribute syntaxes 'text' and 'name' that were supplied by the Client in the Job Creation request. See Section 4.1.4 for a complete description of the "attributes-natural-language" operation attribute. See Sections 5.1.2.2 and 5.1.3.2 for how a Natural Language Override can be supplied explicitly for each 'text' and 'name' attribute value that differs from the value identified by the "attributes-natural-language" attribute.

5.4. Printer Description and Status Attributes

These attributes form the attribute group called "printer-description". Tables 16 and 17 summarize these attributes, their syntax, and whether they are REQUIRED for a Printer to support. If they are not indicated as REQUIRED, they are OPTIONAL. The maximum size in octets for 'text' and 'name' attributes is indicated in parentheses.

Note: How these attributes are set by an Administrator is outside the scope of this document.

Attribute	Syntax	REQUIRED?
charset-configured	charset	REQUIRED
charset-supported	1setOf charset	REQUIRED
color-supported	boolean	RECOMMENDED
compression-supported	1setOf type2 keyword	REQUIRED
document-format-default	mimeMediaType	REQUIRED
document-format-supported	1setOf mimeMediaType	REQUIRED

generated-natural-language-supported	1setOf naturalLanguage	REQUIRED
ipp-versions-supported	1setOf type2 keyword	REQUIRED
job-impressions-supported	rangeOfInteger(0:MAX)	RECOMMENDED
job-k-octets-supported	rangeOfInteger(0:MAX)	
job-media-sheets-supported	rangeOfInteger(1:MAX)	
multiple-document-jobs-supported	boolean	RECOMMENDED
multiple-operation-time-out	integer(1:MAX)	RECOMMENDED
natural-language-configured	naturalLanguage	REQUIRED
operations-supported	1setOf type2 enum	REQUIRED
pdl-override-supported	type2 keyword	REQUIRED
printer-driver-installer	uri	
printer-info	text(127)	RECOMMENDED
printer-location	text(127)	RECOMMENDED
printer-make-and-model	text(127)	RECOMMENDED
printer-message-from-operator	text(127)	
printer-more-info-manufacturer	uri	
printer-name	name(127)	REQUIRED
reference-uri-schemes-supported	1setOf uriScheme	

Table 16: Printer Description Attributes (READ-WRITE)

Attribute	Syntax	REQUIRED?
pages-per-minute-color	integer(0:MAX)	RECOMMENDED
pages-per-minute	integer(0:MAX)	RECOMMENDED
printer-current-time	dateTime|unknown	RECOMMENDED
printer-is-accepting-jobs	boolean	REQUIRED
printer-more-info	uri	RECOMMENDED
printer-state	type1 enum	REQUIRED
printer-state-message	text(MAX)	RECOMMENDED
printer-state-reasons	1setOf type2 keyword	REQUIRED
printer-up-time	integer(1:MAX)	REQUIRED
printer-uri-supported	1setOf uri	REQUIRED
queued-job-count	integer(0:MAX)	REQUIRED
uri-authentication-supported	1setOf type2 keyword	REQUIRED
uri-security-supported	1setOf type2 keyword	REQUIRED

Table 17: Printer Status Attributes (READ-ONLY)

5.4.1. printer-uri-supported (1setOf uri)

This REQUIRED Printer attribute contains one or more URIs for the Printer. It MAY contain more than one URI for the Printer. An Administrator determines a Printer's URIs and configures this attribute to contain those URIs by some means outside the scope of this IPP/1.1 document. The precise format of the URIs is implementation dependent and depends on the protocol. See Sections 5.4.2 and 5.4.3 for a description of the "uri-authentication-supported" and "uri-security-supported" attributes, both of which are the REQUIRED companion attributes to this "printer-uri-supported" attribute. See Sections 3.4 ("Object Identity") and 9.2 ("URIs in Operation, Job, and Printer Attributes") for more information.

5.4.2. uri-authentication-supported (1setOf type2 keyword)

This REQUIRED Printer attribute MUST have the same cardinality (contain the same number of values) as the "printer-uri-supported" attribute. This attribute identifies the Client Authentication mechanism associated with each URI listed in the "printer-uri-supported" attribute. The Printer uses the specified mechanism to identify the authenticated user (see Section 9.3). The "i-th" value in "uri-authentication-supported" corresponds to the "i-th" value in "printer-uri-supported", and it describes the authentication mechanisms used by the Printer when accessed via that URI. See [RFC8010] for more details on Client Authentication.

The following standard 'keyword' values are defined:

- 'none': There is no authentication mechanism associated with the URI. The Printer assumes that the authenticated user is 'anonymous'.

- 'requesting-user-name': When a Client performs an operation whose target is the associated URI, the Printer assumes that the authenticated user is specified by the "requesting-user-name" operation attribute (see Section 9.3). If the "requesting-user-name" attribute is absent in a request, the Printer assumes that the authenticated user is 'anonymous'.

- 'basic': When a Client performs an operation whose target is the associated URI, the Printer challenges the Client with HTTP Basic authentication [RFC7617]. The Printer assumes that the authenticated user is the name received via the Basic authentication mechanism.

- 'digest': When a Client performs an operation whose target is the associated URI, the Printer challenges the Client with HTTP Digest authentication [RFC7616]. The Printer assumes that the authenticated user is the name received via the Digest authentication mechanism.

- 'certificate': When a Client performs an operation whose target is the associated URI, the Printer expects the Client to provide an X.509 certificate. The Printer assumes that the authenticated user is one of the textual names (Common Name or Subject Alternate Names) contained within the certificate.

5.4.3. uri-security-supported (1setOf type2 keyword)

This REQUIRED Printer attribute MUST have the same cardinality (contain the same number of values) as the "printer-uri-supported" attribute. This attribute identifies the security mechanisms used for each URI listed in the "printer-uri-supported" attribute. The "i-th" value in "uri-security-supported" corresponds to the "i-th" value in "printer-uri-supported", and it describes the security mechanisms used for accessing the Printer via that URI. See [RFC8010] for more details on security mechanisms.

The following standard 'keyword' values are defined:

- 'none': There are no secure communication channel protocols in use for the given URI.

- 'tls': TLS [RFC5246] [RFC7525] is the secure communications channel protocol in use for the given URI.

This attribute is orthogonal to the definition of a Client Authentication mechanism. Specifically, 'none' does not exclude Client Authentication. See Section 5.4.2.

Consider the following example. For a single Printer, an Administrator configures the "printer-uri-supported", "uri-authentication-supported", and "uri-security-supported" attributes as follows:

> "printer-uri-supported": 'ipp://printer.example.com/ipp/print/open-use-printer', 'ipp://printer.example.com/ipp/print/restricted-use-printer', 'ipps://printer.example.com/ipp/print/private-printer'
>
> "uri-authentication-supported": 'none', 'digest', 'basic'
>
> "uri-security-supported": 'none', 'none', 'tls'

In this case, one Printer has three URIs.

- For the first URI, 'ipp://printer.example.com/ipp/print/open-use-printer', the value 'none' in "uri-security-supported" indicates that there is no secure channel protocol configured to run under HTTP. The value of 'none' in "uri-authentication-supported" indicates that all users are 'anonymous'. There will be no challenge, and the Printer will ignore "requesting-user-name".

- For the second URI, 'ipp://printer.example.com/ipp/print/restricted-use-printer', the value 'none' in "uri-security-supported" indicates that there is no secure channel protocol configured to run under HTTP. The value of 'digest' in "uri-authentication-supported" indicates that the Printer will issue a challenge and that the Printer will use the name supplied by the Digest mechanism to determine the authenticated user (see Section 9.3).

- For the third URI, 'ipps://printer.example.com/ipp/print/private-printer', the value 'tls' in "uri-security-supported" indicates that TLS is being used to secure the channel. The Client SHOULD be prepared to use TLS framing to negotiate an acceptable ciphersuite to use while communicating with the Printer. In this case, the name implies the use of a secure communications channel, but the fact is made explicit by the presence of the 'tls' value in "uri-security-supported". The Client does not need to resort to understanding which security mechanisms it must use by following naming conventions or by parsing the URI to determine which security mechanisms are implied. The value of 'basic' in "uri-authentication-supported" indicates that the Printer will issue a challenge and that the Printer will use the name supplied by the Basic mechanism to determine the authenticated user (see Section 9.3). Because this challenge occurs in a TLS session, the channel is secure.

Some Printers will be configured to support only one channel (either configured to use TLS access or not) and only one authentication mechanism. Such Printers only have one URI listed in the "printer-uri-supported" attribute. No matter the configuration of the Printer (whether it has only one URI or more than one URI), a Client MUST supply only one URI in the target "printer-uri" operation attribute.

5.4.4. printer-name (name(127))

This REQUIRED Printer attribute contains the name of the Printer. It is a name that is more End User friendly than a URI. An Administrator determines a Printer's name and sets this attribute to that name. This name can be the last part of the Printer's URI, or it can be unrelated. In non-US-English locales, a name can contain characters that are not allowed in a URI.

5.4.5. printer-location (text(127))

This RECOMMENDED Printer attribute identifies the location of the device. This could include things like 'in Room 123A, second floor of building XYZ'.

5.4.6. printer-info (text(127))

This RECOMMENDED Printer attribute provides descriptive information about this Printer. This could include things like 'This printer can be used for printing color transparencies for HR presentations', or 'Out of courtesy for others, please print only small (1-5 page) jobs at this printer', or even 'This printer is going away on July 1; please find a new printer'.

5.4.7. printer-more-info (uri)

This RECOMMENDED Printer attribute contains a URI used to obtain more information about this specific Printer. For example, this could be an HTTP URI referencing an HTML page accessible to a web browser. The information obtained from this URI is intended for End User consumption. Features outside the scope of IPP can be accessed from this URI. The information is intended to be specific to this Printer instance and site-specific services, e.g., Job pricing, services offered, and End User assistance. The device manufacturer can initially populate this attribute.

5.4.8. printer-driver-installer (uri)

This Printer attribute contains a URI to use to locate the driver installer for this Printer. This attribute is intended for consumption by automata. The mechanics of Printer driver installation are outside the scope of this document. The device manufacturer can initially populate this attribute.

5.4.9. printer-make-and-model (text(127))

This RECOMMENDED Printer attribute identifies the make and model of the device. The device manufacturer can initially populate this attribute.

5.4.10. printer-more-info-manufacturer (uri)

This Printer attribute contains a URI used to obtain more information about this type of device. The information obtained from this URI is intended for End User consumption. Features outside the scope of IPP can be accessed from this URI (e.g., latest firmware, upgrades, Printer drivers, optional features available, details on color support). The information is intended to be germane to this Printer without regard to site-specific modifications or services. The device manufacturer can initially populate this attribute.

5.4.11. printer-state (type1 enum)

This REQUIRED Printer attribute identifies the current state of the device. The "printer-state reasons" attribute augments the "printer-state" attribute to give more detailed information about the Printer in the given Printer state.

A Printer updates this attribute continually if asynchronous event notification [RFC3995] is supported.

Standard enum values are defined in Table 18. Values of "printer-state-reasons", such as 'spool-area-full' and 'stopped-partly', MAY be used to provide further information.

Value	Symbolic Name and Description
'3'	'idle': Indicates that new Jobs can start processing without waiting.
'4'	'processing': Indicates that Jobs are processing; new Jobs will wait before processing.
'5'	'stopped': Indicates that no Jobs can be processed and intervention is required.

Table 18: "printer-state" Enum Values

5.4.12. printer-state-reasons (1setOf type2 keyword)

This REQUIRED Printer attribute supplies additional detail about the device's state. Some of the value definitions indicate conformance requirements; the rest are OPTIONAL.

Each 'keyword' value MAY have a suffix to indicate its level of severity. The three levels are 'report' (least severe), 'warning', and 'error' (most severe):

- '-report': This suffix indicates that the reason is a "report". An implementation can choose to omit some or all reports. Some reports specify finer granularity about the Printer state; others serve as a precursor to a warning. A report MUST contain nothing that could affect the printed output. Reports correspond to the 'other' value for the prtAlertSeverityLevel property in the Printer MIB [RFC3805].

- ’-warning’: This suffix indicates that the reason is a "warning". An implementation can choose to omit some or all warnings. Warnings serve as a precursor to an error. A warning MUST contain nothing that prevents a Job from completing, though in some cases the output can be of lower quality. Warnings correspond to the ’warning’ value for the prtAlertSeverityLevel property in the Printer MIB [RFC3805].

- ’-error’: This suffix indicates that the reason is an "error". An implementation MUST include all errors. If this attribute contains one or more errors, the Printer MUST be in the ’stopped’ state. Errors correspond to the ’critical’ value for the prtAlertSeverityLevel property in the Printer MIB [RFC3805].

If the implementation does not add any one of the three suffixes and the value is not ’none’, Clients can assume that the reason is an "error" if the Printer is in the ’stopped’ state and a "warning" if the Printer is in any other state.

If a Printer controls more than one Output Device, each value of this attribute MAY apply to one or more of the Output Devices. An error on one Output Device that does not stop the Printer as a whole MAY appear as a warning in the Printer’s "printer-state-reasons" attribute. If "printer-state" for such a Printer has a value of ’stopped’, then there MUST be an error reason among the values in the "printer-state-reasons" attribute.

The following standard ’keyword’ values are defined:

- ’none’: There are no reasons. This state reason is semantically equivalent to "printer-state-reasons" without any value and MUST be used, since the ’1setOf’ attribute syntax requires at least one value.

- ’other’: The device has detected a condition other than one listed in this document.

- ’connecting-to-device’: The Printer has scheduled a Job on the Output Device and is in the process of connecting to a shared network Output Device (and might not be able to actually start printing the Job for an arbitrarily long time, depending on the usage of the Output Device by other servers on the network).

- ’cover-open’: One or more covers on the device are open, equivalent to a prtCoverStatus [RFC3805] of 3 (coverOpen).

- ’developer-empty: The device is out of developer.

- 'developer-low': The device is low on developer.
- 'door-open': One or more doors on the device are open, equivalent to a prtCoverStatus [RFC3805] of 3 (coverOpen).
- 'fuser-over-temp': The fuser temperature is above normal, equivalent to a prtMarkerStatus [RFC3805] of 19 (the sum of "Unavailable because Broken" (3) and "Critical Alerts" (16)).
- 'fuser-under-temp': The fuser temperature is below normal, equivalent to a prtMarkerStatus [RFC3805] of 19 (the sum of "Unavailable because Broken" (3) and "Critical Alerts" (16)).
- 'input-tray-missing': One or more input trays are not in the device, equivalent to a prtInputStatus [RFC3805] of 19 (the sum of "Unavailable because Broken" (3) and "Critical Alerts" (16)).
- 'interlock-open': One or more interlock devices on the Printer are unlocked, equivalent to a prtCoverStatus [RFC3805] of 5 (interlockOpen).
- 'interpreter-resource-unavailable': An interpreter resource is unavailable (i.e., font, form).
- 'marker-supply-empty: The device is out of at least one marker supply, e.g., toner, ink, ribbon.
- 'marker-supply-low': The device is low on at least one marker supply, e.g., toner, ink, ribbon.
- 'marker-waste-almost-full': The device marker supply waste receptacle is almost full.
- 'marker-waste-full': The device marker supply waste receptacle is full.
- 'media-empty': At least one input tray is empty, equivalent to a prtInputStatus [RFC3805] of 19 (the sum of "Unavailable because Broken" (3) and "Critical Alerts" (16)).
- 'media-jam': The device has a media jam, equivalent to a prtInputStatus [RFC3805] of 19 (the sum of "Unavailable because Broken" (3) and "Critical Alerts" (16)).
- 'media-low': At least one input tray is low on media, equivalent to a prtInputStatus [RFC3805] of 8 (Non-Critical Alerts).

- 'media-needed': A tray has run out of media, equivalent to a prtInputStatus [RFC3805] value of 17 (the sum of "Unavailable and OnRequest" (1) and "Critical Alerts" (16)).

- 'moving-to-paused': Someone has paused the Printer using the Pause-Printer operation (see Section 4.2.7) or other means, but the device(s) is taking an appreciable time to stop. Later, when all output has stopped, "printer-state" becomes 'stopped', and the 'paused' value replaces the 'moving-to-paused' value in the "printer-state-reasons" attribute. This value MUST be supported if the Pause-Printer operation is supported and the implementation takes significant time to pause a device in certain circumstances.

- 'opc-life-over': The optical photo conductor is no longer functioning, equivalent to a prtMarkerStatus [RFC3805] of 19 (the sum of "Unavailable because Broken" (3) and "Critical Alerts" (16)).

- 'opc-near-eol': The optical photo conductor is near its end of life, equivalent to a prtMarkerStatus [RFC3805] of 8 (Non-Critical Alerts).

- 'output-area-almost-full': One or more output areas are almost full, e.g., tray, stacker, collator, equivalent to a prtOutputStatus [RFC3805] of 8 (Non-Critical Alerts).

- 'output-area-full': One or more output areas are full, e.g., tray, stacker, collator, equivalent to a prtInputStatus [RFC3805] of 19 (the sum of "Unavailable because Broken" (3) and "Critical Alerts" (16)).

- 'output-tray-missing': One or more output trays are not in the device, equivalent to a prtOutputStatus [RFC3805] of 19 (the sum of "Unavailable because Broken" (3) and "Critical Alerts" (16)).

- 'paused': Someone has paused the Printer using the Pause-Printer operation (see Section 4.2.7) or other means, and the Printer's "printer-state" is 'stopped'. In this state, a Printer MUST NOT produce printed output, but it MUST perform other operations requested by a Client. If a Printer had been printing a Job when the Printer was paused, the Printer MUST resume printing that Job when the Printer is no longer paused and leave no evidence in the printed output of such a pause. This value MUST be supported if the Pause-Printer operation is supported.

- 'shutdown': Someone has removed a Printer from service, and the device can be powered down or physically removed. In this state, a Printer MUST NOT produce printed output, and unless the Printer is realized by a print server that is still active, the Printer MUST perform no other operations requested by a Client, including returning this value. If a Printer had been printing a Job when it was shut down, the Printer MAY resume printing that Job when the Printer is restarted. If the Printer resumes printing such a Job, it can leave evidence in the printed output of such a shutdown, e.g., the part printed before the shutdown can be printed a second time after the shutdown.

- 'spool-area-full': The limit of persistent storage allocated for spooling has been reached. The Printer is temporarily unable to accept more Jobs. The Printer will remove this value when it is able to accept more Jobs. This value SHOULD be used by a non-spooling Printer that only accepts one or a small number of Jobs at a time or by a spooling Printer that has filled the spool space.

- 'stopped-partly': When a Printer controls more than one Output Device, this reason indicates that one or more Output Devices are stopped. If the reason is a report, fewer than half of the Output Devices are stopped. If the reason is a warning, fewer than all of the Output Devices are stopped.

- 'stopping': The Printer is in the process of stopping the device and will be stopped in a while. When the device is stopped, the Printer will change the Printer's state to 'stopped'. The 'stopping-warning' reason is never an error, even for a Printer with a single Output Device. When an Output Device ceases accepting Jobs, the Printer will have this reason while the Output Device completes printing.

- 'timed-out': The server was able to connect to the Output Device (or is always connected) but was unable to get a response from the Output Device.

- 'toner-empty': The device is out of toner.

- 'toner-low': The device is low on toner.

5.4.13. printer-state-message (text(MAX))

This RECOMMENDED Printer attribute specifies information about the "printer-state" and "printer-state-reasons" attributes in human-readable text. If the Printer supports this attribute, the Printer MUST be able to generate this message in any of the natural languages identified by the Printer's "generated-natural-language-supported" attribute (see the "attributes-natural-language" operation attribute specified in Section 4.1.4.1).

5.4.14. ipp-versions-supported (1setOf type2 keyword)

This REQUIRED attribute identifies the IPP version(s) that this Printer supports, including major and minor versions, i.e., the version numbers for which this Printer implementation meets the conformance requirements. For version number validation, the Printer matches the (2-octet binary) "version-number" parameter supplied by the Client in each request (see Sections 4.1.1 and 4.1.8) with the (US-ASCII) 'keyword' values of this attribute.

The following standard 'keyword' values are defined in this document:

- '1.0': Meets the conformance requirements of IPP version 1.0 as specified in RFC 2566 [RFC2566] and RFC 2565 [RFC2565], including any extensions registered according to Section 7 and any extension defined in this version or any future version of the IPP Model and Semantics document (this document) or the IPP Encoding and Transport document [RFC8010] following the rules, if any, when the "version-number" parameter is '1.0'.

- '1.1': Meets the conformance requirements of IPP version 1.1 as specified in this document and [RFC8010], including any extensions registered according to Section 7 and any extension defined in any future versions of this document or [RFC8010] following the rules, if any, when the "version-number" parameter is '1.1'.

Additional values are defined in "IPP Version 2.0, 2.1, and 2.2" [PWG5100.12].

5.4.15. operations-supported (1setOf type2 enum)

This REQUIRED Printer attribute specifies the set of supported operations for this Printer and contained Jobs.

This attribute is encoded as any other enum attribute syntax according to [RFC8010] as 32 bits. However, all 32-bit enum values for this attribute MUST NOT exceed 0x00007fff, since these same values are also passed in two octets in the "operation-id" field (see

Section 4.1.1) in each Protocol request with the two high-order octets omitted in order to indicate the operation being performed [RFC8010].

Table 19 lists the "operations-supported" attribute and "operation-id" parameter (see Section 4.1.2) enum values that are defined in this document.

Value	Operation Name
0x0000	reserved, not used
0x0001	reserved, not used
0x0002	Print-Job
0x0003	Print-URI
0x0004	Validate-Job
0x0005	Create-Job
0x0006	Send-Document
0x0007	Send-URI
0x0008	Cancel-Job
0x0009	Get-Job-Attributes
0x000a	Get-Jobs
0x000b	Get-Printer-Attributes
0x000c	Hold-Job
0x000d	Release-Job
0x000e	Restart-Job
0x000f	reserved for a future operation

0x0010	Pause-Printer
0x0011	Resume-Printer
0x0012	Purge-Jobs
0x0013-0x3fff	additional registered operations (see the IANA IPP registry and Section 7.8)
0x4000-0x7fff	reserved for vendor extensions (see Section 7.8)

Table 19: "operations-supported" Enum Values

5.4.16. multiple-document-jobs-supported (boolean)

This RECOMMENDED Printer attribute indicates whether the Printer supports more than one Document per Job, i.e., more than one Send-Document operation with Document data and/or Send-URI operations. If the Printer supports the Create-Job and Send-Document operations (see Sections 4.2.4 and 4.3.1), it MUST support this attribute.

5.4.17. charset-configured (charset)

This REQUIRED Printer attribute identifies the charset that the Printer has been configured to represent 'text' and 'name' Printer attributes that are set by the Operator, Administrator, or manufacturer, i.e., for "printer-name" (name), "printer-location" (text), "printer-info" (text), and "printer-make-and-model" (text). Therefore, the value of the Printer's "charset-configured" attribute MUST also be among the values of the Printer's "charset-supported" attribute.

5.4.18. charset-supported (1setOf charset)

This REQUIRED Printer attribute identifies the set of charsets that the Printer and contained Jobs support in attributes with attribute syntaxes 'text' and 'name'. At least the value 'utf-8' MUST be present, since IPP objects MUST support the UTF-8 [RFC3629] charset. If a Printer supports a charset, it means that for all attributes of syntaxes 'text' and 'name' the Printer MUST (1) accept the charset in requests and (2) return the charset in responses as needed.

If more charsets than UTF-8 are supported, the Printer MUST perform charset conversion between the charsets as described in Section 4.1.4.2.

5.4.19. natural-language-configured (naturalLanguage)

This REQUIRED Printer attribute identifies the natural language that the Printer has been configured to represent 'text' and 'name' Printer attributes that are set by the Operator, Administrator, or manufacturer, i.e., for "printer-name" (name), "printer-location" (text), "printer-info" (text), and "printer-make-and-model" (text). When returning these Printer attributes, the Printer MAY return them in the configured natural language specified by this attribute, instead of the natural language requested by the Client in the "attributes-natural-language" operation attribute. See Section 4.1.4.1 for the specification of the OPTIONAL support for multiple natural languages. Therefore, the value of the Printer's "natural-language-configured" attribute MUST also be among the values of the Printer's "generated-natural-language-supported" attribute.

5.4.20. generated-natural-language-supported (1setOf naturalLanguage)

This REQUIRED Printer attribute identifies the natural language(s) that the Printer and contained Jobs support in attributes with attribute syntaxes 'text' and 'name'. The natural language(s) supported depends on implementation and/or configuration. Unlike charsets, Printers MUST accept requests with any natural language or any Natural Language Override whether the natural language is supported or not.

If a Printer supports a natural language, it means that for any of the attributes for which the Printer or Job generates messages, i.e., for the "job-state-message" and "printer-state-message" attributes and operation messages (see Section 4.1.5) in operation responses, the Printer and Job MUST be able to generate messages in any of the Printer's supported natural languages. See Sections 4.1.4, 5.1.2, and 5.1.3 for the definitions of 'text' and 'name' attributes in operation requests and responses.

Note: A Printer that supports multiple natural languages often has separate catalogs of messages, one for each natural language supported.

5.4.21. document-format-default (mimeMediaType)

This REQUIRED Printer attribute identifies the Document format that the Printer has been configured to assume if the Client does not supply a "document-format" operation attribute in any of the operation requests that supply Document data. The standard values for this attribute are Internet media types (sometimes called "MIME media types"). For further details, see the description of the 'mimeMediaType' attribute syntax in Section 5.1.10.

5.4.22. document-format-supported (1setOf mimeMediaType)

This REQUIRED Printer attribute identifies the set of Document formats that the Printer and contained Jobs can support. For further details, see the description of the 'mimeMediaType' attribute syntax in Section 5.1.10.

5.4.23. printer-is-accepting-jobs (boolean)

This REQUIRED Printer attribute indicates whether the Printer is currently able to accept Jobs, i.e., is accepting Print-Job, Print-URI, and Create-Job requests. If the value is 'true', the Printer is accepting Jobs. If the value is 'false', the Printer is currently rejecting any Jobs submitted to it. In this case, the Printer returns the 'server-error-not-accepting-jobs' status-code.

This value is independent of the "printer-state" and "printer-state-reasons" attributes because its value does not affect the current Job; rather, it affects future Jobs. This attribute, when 'false', causes the Printer to reject Jobs even when "printer-state" is 'idle' or, when 'true', causes the Printer to accept Jobs even when "printer-state" is 'stopped'.

5.4.24. queued-job-count (integer(0:MAX))

This REQUIRED Printer attribute contains a count of the number of Jobs that are either 'pending', 'processing', 'pending-held', or 'processing-stopped' and is set by the Printer.

5.4.25. printer-message-from-operator (text(127))

This Printer attribute provides a message from an Operator, Administrator, or "intelligent" process to indicate to the End User information or status of the Printer, such as why it is unavailable or when it is expected to be available.

5.4.26. color-supported (boolean)

This RECOMMENDED Printer attribute identifies whether the device is capable of any type of color printing at all, including highlight color. All Document instructions having to do with color are embedded within the Document PDL, although IPP attributes can affect the rendering of those colors.

Note: End Users are able to determine the nature and details of the color support by querying the "printer-more-info-manufacturer" Printer attribute.

5.4.27. reference-uri-schemes-supported (1setOf uriScheme)

This Printer attribute specifies which URI schemes are supported for use in the "document-uri" operation attribute of the Print-URI or Send-URI operations. If a Printer supports these OPTIONAL operations, it MUST support the "reference-uri-schemes-supported" Printer attribute with at least the following URI scheme value:

- 'ftp': The Printer will use an FTP 'get' operation as defined in [RFC959] using FTP URLs as defined by [RFC3986].

The Printer MAY support other URI schemes (see Section 5.1.7).

5.4.28. pdl-override-supported (type2 keyword)

This REQUIRED Printer attribute expresses the ability of a particular Printer implementation to override Document data instructions with IPP attributes. The following 'keyword' values are defined in this document:

- 'attempted': This value indicates that the Printer attempts to make the IPP attribute values take precedence over embedded instructions in the Document data; however, there is no guarantee.

- 'not-attempted': This value indicates that the Printer makes no attempt to make the IPP attribute values take precedence over embedded instructions in the Document data.

Appendix C contains a full description of how this attribute interacts with and affects other IPP attributes, especially the "ipp-attribute-fidelity" attribute.

5.4.29. printer-up-time (integer(1:MAX))

This REQUIRED Printer attribute indicates the amount of time (in seconds) that this Printer instance has been up and running. The value is a monotonically increasing value starting from 1 when the Printer is started up (initialized, booted, etc.). This value is used to populate the Event Time Job Status attributes "time-at-creation", "time-at-processing", and "time-at-completed" (see Section 5.3.14).

If the Printer goes down at some value 'n' and comes back up, the implementation MAY:

1. know how long it has been down and resume at some value greater than 'n', or

2. restart from 1.

In other words, if the device or devices that the Printer is representing are restarted or power-cycled, the Printer MAY continue counting this value or MAY reset this value to 1, depending on implementation. However, if the Printer software ceases running and restarts without knowing the last value for "printer-up-time", the implementation MUST reset this value to 1. If this value is reset and the Printer has persistent Jobs, the Printer MUST reset the "time-at-xxx (integer)" Event Time Job Status attributes according to Section 5.3.14. An implementation MAY use both implementation alternatives, depending on warm versus cold start, respectively.

5.4.30. printer-current-time (dateTime|unknown)

This RECOMMENDED Printer attribute indicates the current date and time. This value is used to populate the Event Time Job Status attributes "date-time-at-creation", "date-time-at-processing", and "date-time-at-completed" (see Section 5.3.14).

This value is obtained on a "best effort" basis and in practice does not have to be precise in order to be useful. A Printer implementation sets the value of this attribute by obtaining the date and time via some implementation-dependent means, such as getting the value from a network time server, initialization at time of manufacture, or setting by an Administrator. See [RFC3196] and [PWG5100.19] for examples. If an implementation supports this attribute and the implementation knows that it has not yet been set, then the implementation MUST return the value of this attribute using the out-of-band 'unknown', meaning the value is not yet known. See the beginning of Section 5.1.

The time zone of this attribute might not be the time zone used by people located near the Printer or device. The Client MUST NOT expect the time zone of any received 'dateTime' value to be in the time zone of the Client or in the time zone of the people located near the Printer.

The Client SHOULD display any dateTime attributes to the user in the Client's local time by converting the 'dateTime' value returned by the server to the time zone of the Client, rather than using the time zone returned by the Printer in attributes that use the 'dateTime' attribute syntax.

Note: Prior versions of this document incorrectly specified the use of the 'no-value' out-of-band value when the current date and time had not been set. The correct out-of-band value is 'unknown', since there is always an intrinsic current date and time.

5.4.31. multiple-operation-time-out (integer(1:MAX))

This RECOMMENDED Printer attribute identifies the minimum time (in seconds) that the Printer waits for additional Send-Document or Send-URI operations to follow a still-open Job before taking any recovery actions, such as the ones indicated in Section 4.3.1. If the Printer supports the Create-Job and Send-Document operations (see Sections 4.2.4 and 4.3.1), it MUST support this attribute.

Printers SHOULD use a value between '60' and '240' (seconds). An implementation MAY allow an Administrator to set this attribute by means not defined in this document. If so, the Administrator MAY be able to set values outside this range.

5.4.32. compression-supported (1setOf type2 keyword)

This REQUIRED Printer attribute identifies the set of supported compression algorithms for Document data. Compression only applies to the Document data; compression does not apply to the encoding of the IPP operation itself. The supported values are used to validate the Client-supplied "compression" operation attributes in Print-Job and Send-Document requests.

Standard 'keyword' values defined in this document are:

- 'none': no compression is used.
- 'deflate': ZIP inflate/deflate compression technology described in RFC 1951 [RFC1951].
- 'gzip': GNU zip compression technology described in RFC 1952 [RFC1952].
- 'compress': UNIX compression technology described in RFC 1977 [RFC1977].

5.4.33. job-k-octets-supported (rangeOfInteger(0:MAX))

This Printer attribute specifies the upper and lower bounds of total sizes of Jobs in K octets, i.e., in units of 1024 octets. The supported values are used to validate the Client-supplied "job-k-octets" operation attribute in Job Creation requests. The corresponding Job Description attribute "job-k-octets" is defined in Section 5.3.17.1.

5.4.34. job-impressions-supported (rangeOfInteger(0:MAX))

This RECOMMENDED Printer attribute specifies the upper and lower bounds for the number of Impressions per Job. The supported values are used to validate the Client-supplied "job-impressions" operation attribute in Job Creation requests. The corresponding Job Description attribute "job-impressions" is defined in Section 5.3.17.2.

5.4.35. job-media-sheets-supported (rangeOfInteger(1:MAX))

This Printer attribute specifies the upper and lower bounds for the number of Media Sheets per Job. The supported values are used to validate the Client-supplied "job-media-sheets" operation attribute in Job Creation requests. The corresponding Job attribute "job-media-sheets" is defined in Section 5.3.17.3.

5.4.36. pages-per-minute (integer(0:MAX))

This RECOMMENDED Printer attribute specifies the nominal number of pages per minute to the nearest whole number that can be generated by this Printer (e.g., simplex, black-and-white). This attribute is informative, not a service guarantee. Generally, it is the value used in the marketing literature to describe the speed of the device.

A value of 0 indicates a device that takes more than two minutes to process a page.

5.4.37. pages-per-minute-color (integer(0:MAX))

This RECOMMENDED Printer attribute specifies the nominal number of pages per minute to the nearest whole number that can be generated by this Printer when printing color (e.g., simplex, color). For purposes of this attribute, the meaning of "color" is the same as that for the "color-supported" attribute; namely, the device is capable of any type of color printing at all, including highlight color. This attribute is informative, not a service guarantee. Generally, it is the value used in the marketing literature to describe the color capabilities of this device.

A value of 0 indicates a device that takes more than two minutes to process a page in color.

If a color device has several color modes, it MAY use the "pages-per-minute" value for this attribute that corresponds to the mode that produces the highest number.

Printers that are black-and-white only MUST NOT support this attribute. If this attribute is present, then the "color-supported" Printer Description attribute MUST be present and have a 'true' value.

The values of the "pages-per-minute" and "pages-per-minute-color" attributes returned by the Get-Printer-Attributes operation MAY be affected by the "document-format" attribute supplied by the Client in the Get-Printer-Attributes request. In other words, the implementation MAY have different speeds, depending on the Document format being processed. See Section 4.2.5.1 ("Get-Printer-Attributes Request").

6. Conformance

This section describes conformance issues and requirements. This document introduces model entities such as objects, operations, attributes, attribute syntaxes, and attribute values. The following sections describe the conformance requirements that apply to these model entities.

6.1. Client Conformance Requirements

This section describes the conformance requirements for a Client (see Section 3.1), whether it be:

1. contained within software controlled by an End User, e.g., activated by the "Print" menu item in an application that sends IPP requests, or

2. the print server component that sends IPP requests to either an Output Device or another "downstream" print server.

A conforming Client supports all REQUIRED operations as defined in this document. For each attribute included in an operation request, a conforming Client MUST supply a value whose type and value syntax conforms to the requirements specified in Sections 4 and 5 of this document. A conforming Client MAY supply any Standards Track extensions and/or vendor extensions in an operation request, as long as the extensions meet the requirements in Section 7.

While this document does not define conformance requirements for the user interfaces provided by IPP Clients or their applications, best practices for user interfaces are defined in [PWG5100.19].

A Client MUST be able to accept any of the attribute syntaxes defined in Section 5.1, including their full range, that can be returned to it in a response from a Printer. In particular, for each attribute that the Client supports whose attribute syntax is 'text', the Client MUST accept and process both the 'textWithoutLanguage' and 'textWithLanguage' forms. Similarly, for each attribute that the Client supports whose attribute syntax is 'name', the Client MUST accept and process both the 'nameWithoutLanguage' and 'nameWithLanguage' forms. For presentation purposes, truncation of long attribute values is not recommended. A recommended approach would be for the Client implementation to allow the user to scroll through long attribute values.

A response MAY contain attribute groups, attributes, attribute syntaxes, values, and status-code values that the Client does not expect. Therefore, a Client implementation MUST gracefully handle such responses and not refuse to interoperate with a conforming Printer that is returning Standards Track extensions or vendor extensions, including attribute groups, attributes, attribute syntaxes, attribute values, status-code values, and out-of-band attribute values that conform to Section 7. Clients can choose to ignore any parameters, attribute groups, attributes, attribute syntaxes, or values that they do not understand.

While a Client is sending data to a Printer, it SHOULD do its best to prevent a channel from being closed by a lower layer when the channel is blocked (i.e., flow-controlled off) for whatever reason, e.g., 'out of paper' or 'Job ahead hasn't freed up enough memory'. However, the layer that launched the print submission (e.g., an End User) MAY close the channel in order to cancel the Job. When a Client closes a channel, a Printer MAY print all or part of the received portion of the Document. See the Encoding and Transport document [RFC8010] for more details.

A Client MUST support Client Authentication as defined in [RFC8010]. A Client SHOULD support Operation Privacy and Server Authentication as defined in [RFC8010]. See also Section 9 of this document.

6.2. IPP Object Conformance Requirements

This section specifies the conformance requirements for conforming implementations of IPP objects (see Section 3). These requirements apply to an IPP object whether it is:

1) an (embedded) device component that accepts IPP requests and controls the device, or

2) a component of a print server that accepts IPP requests (where the print server controls one or more networked devices using IPP or other protocols).

6.2.1. Objects

Conforming implementations MUST implement all of the model objects as defined in this document in the indicated sections:

Section 3.1 - Printer Object

Section 3.2 - Job Object

6.2.2. Operations

Conforming IPP object implementations MUST implement all of the REQUIRED model operations, including REQUIRED responses, as defined in this document in the indicated sections. Table 20 lists the operations for a Printer, while Table 21 lists the operations for a Job.

Operation	Conformance
Print-Job (Section 4.2.1)	REQUIRED
Print-URI (Section 4.2.2)	OPTIONAL
Validate-Job (Section 4.2.3)	REQUIRED
Create-Job (Section 4.2.4)	RECOMMENDED
Get-Printer-Attributes (Section 4.2.5)	REQUIRED
Get-Jobs (Section 4.2.6)	REQUIRED
Pause-Printer (Section 4.2.7)	OPTIONAL
Resume-Printer (Section 4.2.8)	OPTIONAL
Purge-Jobs (Section 4.2.9)	SHOULD NOT

Table 20: Conformance Requirements for Printer Operations

Operation	Conformance
Send-Document (Section 4.3.1)	RECOMMENDED
Send-URI (Section 4.3.2)	RECOMMENDED
Cancel-Job (Section 4.3.3)	REQUIRED
Get-Job-Attributes (Section 4.3.4)	REQUIRED
Hold-Job (Section 4.3.5)	OPTIONAL
Release-Job (Section 4.3.6)	OPTIONAL
Restart-Job (Section 4.3.7)	SHOULD NOT

Table 21: Conformance Requirements for Job Operations

Conforming IPP objects MUST support all REQUIRED operation attributes and all values of such attributes if so indicated in the description. Conforming IPP objects MUST ignore all unsupported or unknown operation attributes or Operation Attributes groups received in a request but MUST reject a request that contains a supported operation attribute that contains an unsupported value.

Conforming IPP objects MAY return operation responses that contain attribute groups, attribute names, attribute syntaxes, attribute values, and status-code values that are extensions to this specification. The additional attribute groups MAY occur in any order.

The following section on object attributes specifies the support required for object attributes.

6.2.3. IPP Object Attributes

Conforming IPP objects MUST support all of the REQUIRED object attributes, as defined in this document in the indicated sections.

If an object supports an attribute, it MUST support only those values specified in this document or through the extension mechanism described in Section 6.2.5. It MAY support any non-empty subset of these values. That is, it MUST support at least one of the specified values and at most all of them.

6.2.4. Versions

IPP/1.1 Clients MUST meet the conformance requirements for Clients specified in this document and [RFC8010]. IPP/1.1 Clients MUST be capable of sending requests containing a "version-number" parameter with a value of '1.1'.

IPP/1.1 Printer and Job objects MUST meet the conformance requirements for IPP objects specified in this document and [RFC8010]. IPP/1.1 objects MUST accept requests containing a "version-number" parameter with a '1.1' value or reject the request if the operation is not supported.

It is beyond the scope of this specification to mandate conformance with other IPP versions. However, IPP was deliberately designed to make supporting different versions easy. IPP/1.1 Printer implementations MUST:

- decode and process any well-formed IPP/1.1 request, and
- respond appropriately with a response containing the same "version-number" parameter value used by the Client in the request.

IPP/1.1 Client implementations MUST:

- decode and process any well-formed IPP/1.1 response.

IPP Clients SHOULD try supplying alternate version numbers if they receive a 'server-error-version-not-supported' error in a response.

6.2.5. Extensions

A conforming IPP object MAY support Standards Track extensions and vendor extensions, as long as the extensions meet the requirements specified in Section 7.

For each attribute included in an operation response, a conforming IPP object MUST return a value whose type and value syntax conforms to the requirements specified in Sections 4 and 5 of this document.

6.2.6. Attribute Syntaxes

An IPP object MUST be able to accept any of the attribute syntaxes defined in Section 5.1, including their full range, in any operation in which a Client can supply attributes or the Administrator can configure attributes (by means outside the scope of this IPP/1.1 document). In particular, for each attribute that the IPP object supports whose attribute syntax is 'text', the IPP object MUST accept and process both the 'textWithoutLanguage' and 'textWithLanguage' forms. Similarly, for each attribute that the IPP object supports whose attribute syntax is 'name', the IPP object MUST accept and process both the 'nameWithoutLanguage' and 'nameWithLanguage' forms. Furthermore, an IPP object MUST return attributes to the Client in operation responses that conform to the syntaxes specified in Section 5.1, including their full range if supplied previously by a Client.

6.2.7. Security

An IPP Printer implementation SHOULD contain support for Client Authentication as defined in the IPP/1.1 Encoding and Transport document [RFC8010]. A Printer implementation MAY allow an Administrator to configure the Printer so that all, some, or none of the users are authenticated. See also Section 9 of this document.

An IPP Printer implementation SHOULD contain support for Operation Privacy and Server Authentication as defined in [RFC8010]. A Printer implementation MAY allow an Administrator to configure the degree of support for Operation Privacy and Server Authentication. See also Section 9 of this document.

Security MUST NOT be compromised when a Client supplies a lower "version-number" parameter in a request. For example, if a Printer conforming to IPP/1.1 accepts version '1.0' requests and is configured to enforce Digest Authentication, it MUST do the same for a version '1.0' request.

6.3. Charset and Natural Language Requirements

All Clients and IPP objects MUST support the 'utf-8' charset as defined in Section 5.1.8.

IPP objects MUST be able to accept any Client request that correctly uses the "attributes-natural-language" operation attribute or the Natural Language Override mechanism on any individual attribute whether or not the natural language is supported by the IPP object. If an IPP object supports a natural language, then it MUST be able to translate (perhaps by table lookup) all generated 'text' or 'name' attribute values into one of the supported languages (see Section 4.1.4).

7. IANA Considerations

This section describes the procedures for defining Standards Track and vendor extensions to this document. This affects the following subregistries of the IANA IPP registry:

1. Objects
2. Attributes
3. Keyword Attribute Values
4. Enum Attribute Values
5. Attribute Group Tags
6. Out-of-Band Attribute Value Tags
7. Attribute Syntaxes
8. Operations
9. Status-Code Values

Extensions registered for use with IPP are OPTIONAL for Client and IPP object conformance to the IPP/1.1 Model and Semantics document (this document).

These extension procedures are aligned with the guidelines as set forth in "Guidelines for Writing an IANA Considerations Section in RFCs" [RFC5226]. Appendix A describes how to propose new registrations for consideration. IANA will reject registration proposals that leave out required information or do not follow the appropriate format described in Appendix A. The IPP/1.1 Model and Semantics document can also be extended by an appropriate Standards Track document that specifies any of the above extensions.

The IANA policy (using terms defined in [RFC5226]) for all extensions is Specification Required, Expert Review, or First Come First Served as documented in the following subsections. Registrations submitted to IANA are forwarded to the IPP Designated Expert(s) who reviews the proposal on a mailing list that the Designated Expert(s) keeps for this purpose. Initially, that list is the mailing list used by the PWG IPP WG:

ipp@pwg.org

The IPP Designated Expert(s) is appointed by the IESG Area Director responsible for IPP, according to [RFC5226].

In addition, the IANA-PRINTER-MIB [RFC3805] has been updated to reference this document; the current version is available from <http://www.iana.org>.

7.1. Object Extensions

The IANA policy (using terms defined in [RFC5226]) for object extensions was formerly Expert Review; this document changes the policy to Specification Required.

7.2. Attribute Extensibility

Since attribute names are type2 keywords (see Section 5.1.4), the IANA policy (using terms defined in [RFC5226]) for attribute extensions is Expert Review.

For vendor attribute extensions, implementors SHOULD use keywords with a suitable distinguishing prefix such as 'smiNNN-' where NNN is an SMI Private Enterprise Number (PEN) [IANA-PEN]. For example, if the company Example Corp. had obtained the SMI PEN 32473, then a vendor attribute 'foo' would be 'smi32473-foo'.

> Note: Prior versions of this document recommended using a fully qualified domain name [RFC1035] as the prefix (e.g., 'example.com-foo'), and many IPP implementations have also used reversed domain names (e.g., 'com.example-foo'). Domain names have proven problematic due to the length of some domain names, parallel use of country-specific domain names (e.g., 'example.co.jp-foo'), and changes in ownership of domain names.

If a new Printer attribute is defined and its values can be affected by a specific Document format, its specification needs to contain the following sentence:

> "The value of this attribute returned in a Get-Printer-Attributes response MAY depend on the "document-format" attribute supplied (see Section 4.2.5.1) of the IPP/1.1 Model and Semantics document."

If the specification does not, then its value in the Get-Printer-Attributes response MUST NOT depend on the "document-format" attribute supplied in the request.

When a new Job Template attribute is registered, the value of the Printer attributes MAY vary with "document-format" supplied in the request without the specification having to indicate so.

7.3. Keyword Extensibility

The IANA policy (using terms defined in [RFC5226]) for type1 keyword extensions is Specification Required. The IANA policy for type2 keyword extensions is Expert Review. The IANA policy for vendor keyword extensions is First Come First Served. Only attributes using the type1 and type2 keyword syntax can be registered in the IANA IPP registry.

> Note: The type1 or type2 prefix on the basic attribute syntax is provided only to communicate the IANA policy required for registration and is not represented in IPP messages. Both type1 and type2 'keyword' values are represented using the same 'keyword' value tag.

For type1 and type2 keywords, the proposer includes the name of the keyword in the registration proposal, and the name is part of the technical review.

For vendor keyword extensions, implementors SHOULD either:

a. follow attribute-specific guidance such as the guidance defined in [PWG5101.1], or

b. use keywords with a suitable distinguishing prefix, such as 'smiNNN-' where NNN is an SMI Private Enterprise Number (PEN) [IANA-PEN].

For example, if the company Example Corp. had obtained the SMI PEN 32473, then a vendor keyword 'foo' would be 'smi32473-foo'.

> Note: Prior versions of this document recommended using a fully qualified domain name [RFC1035] as the prefix (e.g., 'example.com-foo'), and many IPP implementations have also used reversed domain names (e.g., 'com.example-foo'). Domain names have proven problematic due to the length of some domain names, parallel use of country-specific domain names (e.g., 'example.co.jp-foo'), and changes in ownership of domain names.

When a type2 keyword extension is approved, the IPP Designated Expert(s) becomes the point of contact for any future maintenance that might be required for that registration.

7.4. Enum Extensibility

The IANA policy (using terms defined in [RFC5226]) for type1 enum extensions is Specification Required. The IANA policy for type2 enum extensions is Expert Review. The IANA policy for vendor enum extensions is First Come First Served. Only attributes using the type1 and type2 enum syntax can be registered in the IANA IPP registry.

> Note: The type1 or type2 prefix on the basic attribute syntax is provided only to communicate the IANA policy required for registration and is not represented in IPP messages. Both type1 and type2 enum values are represented using the same enum value tag.

For vendor enum extensions, implementors MUST use values in the reserved integer range, which is 0x40000000 to 0x7fffffff. Implementors SHOULD consult with the IPP Designated Expert(s) to reserve vendor extension value(s) for their usage.

When a type1 or type2 enum extension is approved, the IPP Designated Expert(s), in consultation with IANA, assigns the next available enum number for each enum value.

When a type2 enum extension is approved, the IPP Designated Expert(s) becomes the point of contact for any future maintenance that might be required for that registration.

7.5. Attribute Group Extensibility

The IANA policy (using terms defined in [RFC5226]) for attribute group extensions was formerly Expert Review; this document changes the policy to Specification Required.

For attribute groups, the IPP Designated Expert(s), in consultation with IANA, assigns the next attribute group tag code in the appropriate range as specified in [RFC8010].

7.6. Out-of-Band Attribute Value Extensibility

The IANA policy (using terms defined in [RFC5226]) for out-of-band attribute value extensions was formerly Expert Review; this document changes the policy to Specification Required.

For out-of-band attribute value tags, the IPP Designated Expert(s), in consultation with IANA, assigns the next out-of-band attribute value tag code in the appropriate range as specified in [RFC8010].

7.7. Attribute Syntax Extensibility

The IANA policy (using terms defined in [RFC5226]) for attribute syntax extensions was formerly Expert Review; this document changes the policy to Specification Required. The IANA policy for vendor attribute syntax extensions (tags 0x40000000 to 0x7fffffff) is First Come First Served. Only attribute syntaxes in the range of 0x00000000 to 0x3fffffff can be registered in the IANA IPP registry.

For vendor attribute syntax extensions, implementors MUST use values in the reserved integer range, which is 0x40000000 to 0x7fffffff. Implementors SHOULD consult with the IPP Designated Expert(s) to reserve vendor extension value(s) for their usage.

For registered attribute syntaxes, the IPP Designated Expert(s), in consultation with IANA, assigns the next attribute syntax tag in the appropriate range as specified in [RFC8010].

7.8. Operation Extensibility

The IANA policy (using terms defined in [RFC5226]) for operation extensions is Expert Review. The IANA policy for vendor operation extensions (values 0x4000 to 0x7fff) is First Come First Served. Only operation codes in the range of 0x0000 to 0x3fff can be registered in the IANA IPP registry.

For vendor operation extensions, implementors MUST use values in the reserved integer range, which is 0x4000 to 0x7fff. Implementors SHOULD consult with the IPP Designated Expert(s) to reserve vendor extension value(s) for their usage.

For registered operation extensions, the IPP Designated Expert(s), in consultation with IANA, assigns the next "operation-id" code as specified in Section 5.4.15.

7.9. Status-Code Extensibility

The IANA policy (using terms defined in [RFC5226]) for status-code extensions is Expert Review. The IANA policy for vendor status-code extensions (codes 0x0n80 to 0x0nff, for n = 0 to 5) is First Come First Served. Only status-code values in the range of 0x0n00 to 0x0n7f can be registered in the IANA IPP registry.

The status-code values are allocated in ranges as specified in Appendix B for each status-code class:

"informational" - Request received, continuing process

"successful" - The action was successfully received, understood, and accepted

"redirection" - Further action is taken in order to complete the request

"client-error" - The request contains bad syntax or cannot be fulfilled

"server-error" - The IPP object failed to fulfill an apparently valid request

For vendor operation status-code extensions, implementors MUST use the top of each range (0x0n80 to 0x0nff) as specified in Appendix B. Implementors SHOULD consult with the IPP Designated Expert(s) to reserve vendor extension value(s) for their usage.

For registered operation status-code values, the IPP Designated Expert(s), in consultation with IANA, assigns the next status-code in the appropriate class range as specified in Appendix B.

8. Internationalization Considerations

Some of the attributes have values that are text strings and names that are intended for human understanding rather than machine understanding (see the 'text' and 'name' attribute syntaxes in Sections 5.1.2 and 5.1.3).

In each operation request, the Client

- identifies the charset and natural language of the request that affects each supplied 'text' and 'name' attribute value, and

- requests the charset and natural language for attributes returned by the IPP object in operation responses (as described in Section 4.1.4.1).

In addition, the Client MAY separately and individually identify the Natural Language Override of a supplied 'text' or 'name' attribute using the 'textWithLanguage' and 'nameWithLanguage' techniques described in Sections 5.1.2.2 and 5.1.3.2, respectively.

All IPP objects MUST support the UTF-8 [RFC3629] charset in all 'text' and 'name' attributes supported. If an IPP object supports more than the UTF-8 charset, the object MUST convert between them in order to return the requested charset to the Client according to Section 4.1.4.2. If an IPP object supports more than one natural language, the object SHOULD return 'text' and 'name' values in the natural language requested where those values are generated by the Printer (see Section 4.1.4.1).

For Printers that support multiple charsets and/or multiple natural languages in 'text' and 'name' attributes, different Jobs might have been submitted in differing charsets and/or natural languages. All responses MUST be returned in the charset requested by the Client. However, the Get-Jobs operation uses the 'textWithLanguage' and 'nameWithLanguage' mechanisms to identify the differing natural languages with each Job attribute returned.

The Printer also has configured charset and natural language attributes. The Client can query the Printer to determine the list of charsets and natural languages supported by the Printer and what the Printer's configured values are. See the "charset-configured", "charset-supported", "natural-language-configured", and "generated-natural-language-supported" Printer Description attributes for more details.

The "charset-supported" attribute identifies the supported charsets. If a charset is supported, the IPP object MUST be capable of converting to and from that charset into any other supported charset. In many cases, an IPP object will support only one charset, and it MUST be the UTF-8 charset.

The "charset-configured" attribute identifies the one supported charset that is the native charset, given the current configuration of the IPP object (Administrator defined).

The "generated-natural-language-supported" attribute identifies the set of supported natural languages for generated messages; it is not related to the set of natural languages that MUST be accepted for Client-supplied 'text' and 'name' attributes. For Client-supplied 'text' and 'name' attributes, an IPP object MUST accept ALL supplied natural languages. For example, if a Client supplies a Job name that is in 'fr-ca' but the Printer only generates 'en-us', the Printer object MUST still accept the Job name value.

The "natural-language-configured" attribute identifies the one supported natural language for generated messages that is the native natural language, given the current configuration of the IPP object (Administrator defined).

Attributes of types 'text' and 'name' are populated from different sources. These attributes can be categorized into the following groups (depending on the source of the attribute):

1. Some attributes are supplied by the Client (e.g., the Client-supplied "job-name", "document-name", and "requesting-user-name" operation attributes along with the corresponding Job's "job-name" and "job-originating-user-name" attributes). The IPP object MUST accept these attributes in any natural language no matter what the set of supported languages for generated messages.

2. Some attributes are supplied by the Administrator (e.g., the Printer's "printer-name" and "printer-location" attributes). These can also be in any natural language. If the natural language for these attributes is different than what a Client requests, then they MUST be reported using the Natural Language Override mechanism.

3. Some attributes are supplied by the device manufacturer (e.g., the Printer's "printer-make-and-model" attribute). These can also be in any natural language. If the natural language for these attributes is different than what a Client requests, then they MUST be reported using the Natural Language Override mechanism.

4. Some attributes are supplied by the Operator (e.g., the Job's "job-message-from-operator" attribute). These can also be in any natural language. If the natural language for these attributes is different than what a Client requests, then they MUST be reported using the Natural Language Override mechanism.

5. Some attributes are generated by the IPP object (e.g., the Job's "job-state-message" attribute, the Printer's "printer-state-message" attribute, and the "status-message" operation attribute). These attributes can only be in one of the "generated-natural-language-supported" natural languages. If a Client requests some natural language for these attributes other than one of the supported values, the IPP object SHOULD respond using the value of the "natural-language-configured" attribute (using the Natural Language Override mechanism if needed).

The 'text' and 'name' attributes specified in this version of this document (additional ones will be registered according to the procedures in Section 7) are shown in Table 22.

Attributes	Source
Operation Attributes:	
job-name (name)	Client
document-name (name)	Client
requesting-user-name (name)	Client
status-message (text)	Job or Printer
detailed-status-message (text)	Job or Printer (note 1)
document-access-error (text)	Job or Printer (note 1)
Job Template Attributes:	
job-hold-until (keyword | name)	Client matches Administrator-configured
job-hold-until-default (keyword | name)	Client matches Administrator-configured
job-hold-until-supported (keyword | name)	Client matches Administrator-configured

job-sheets (keyword | name)	Client matches Administrator-configured
job-sheets-default (keyword | name)	Client matches Administrator-configured
job-sheets-supported (keyword | name)	Client matches Administrator-configured
media (keyword | name)	Client matches Administrator-configured
media-default (keyword | name)	Client matches Administrator-configured
media-supported (keyword | name)	Client matches Administrator-configured
media-ready (keyword | name)	Client matches Administrator-configured
Job Description Attributes:	
job-name (name)	Client or Printer
job-originating-user-name (name)	Printer
job-state-message (text)	Job or Printer
output-device-assigned (name(127))	Administrator
job-message-from-operator (text(127))	Operator
job-detailed-status-messages (1setOf text)	Job or Printer (note 1)
job-document-access-errors (1setOf text)	Job or Printer (note 1)
Printer Description Attributes:	
printer-name (name(127))	Administrator
printer-location (text(127))	Administrator
printer-info (text(127))	Administrator
printer-make-and-model (text(127))	Administrator or manufacturer
printer-state-message (text)	Printer
printer-message-from-operator (text(127))	Operator

Table 22: ’text’ and ’name’ Attributes

Note 1: Neither the Printer nor the Client localizes these message attributes, since they are intended for use by the Administrator or other experienced technical persons.

9. Security Considerations

It is difficult to anticipate the security risks that might exist in any given IPP environment. For example, if IPP is used within a given small business over a private LAN with physical security, the risks of exposing Document data can be low enough that the business will choose not to use encryption on that data. However, if the connection between the Client and the IPP object is over a public network, the Client can protect the content of the information during transmission through the network with encryption.

Furthermore, the value of the information being printed can vary from one IPP environment to the next. Printing payroll checks, for example, would have a different value than printing public information from a file. There is also the possibility of denial-of-service attacks, but denial-of-service attacks against printing resources are not well understood, and there are no published precedents regarding this scenario.

Once the authenticated identity of the requester has been supplied to the IPP object, the object uses that identity to enforce any authorization policy that might be in place. For example, one site's policy might be that only the Job owner is allowed to cancel a Job. The details and mechanisms to set up a particular access control policy are not part of this document and are typically established via some other type of administrative or access control framework. However, there are operation status-code values that allow an IPP server to return information back to a Client about any potential access control violations for an IPP object.

During a Job Creation request, the Client's identity is recorded in the Job object in an implementation-defined attribute. This information can be used to verify a Client's identity for subsequent operations on that Job object in order to enforce any access control policy that might be in effect. See Section 9.3 below for more details. This and other information stored in the Job object can also be considered personal or sensitive in nature and can be filtered out as part of a configured privacy policy (Section 9.4).

Since the security levels or the specific threats that an Administrator can be concerned with cannot be anticipated, IPP implementations MUST be capable of operating with different security mechanisms and security policies as required by the individual installation. Security policies might vary from very strong to very weak, or to none at all, and corresponding security mechanisms will be required.

9.1. Security Scenarios

The following sections describe specific security attacks for IPP environments. Where examples are provided, they are illustrative of the environment and not an exhaustive set.

9.1.1. Client and Server in the Same Security Domain

This environment is typical of internal networks where traditional office workers print the output of personal productivity applications on shared workgroup Printers, or where batch applications print their output on large production Printers. Although the identity of the user has been authenticated and can be trusted in this environment, a user might want to protect the content of a Document against such attacks as eavesdropping, replaying, or tampering by using a secure transport such as TLS [RFC5246].

9.1.2. Client and Server in Different Security Domains

Examples of this environment include printing a Document created by the Client on a publicly available Printer, such as at a commercial print shop, or printing a Document remotely on a business associate's Printer. This latter operation is functionally equivalent to sending the Document to the business associate as a facsimile. Printing sensitive information on a Printer in a different security domain requires strong security measures. In this environment, authentication of the Printer is required as well as protection against unauthorized use of print resources. Since the Document crosses security domains, protection against eavesdropping and Document tampering is also required. It will also be important in this environment to protect Printers against "spamming" and malicious Document content -- authentication and Document data pre-scanning can be used to minimize those threats.

9.1.3. Print by Reference

When the Document is not stored on the Client, printing can be done by reference. That is, the print request can contain a reference, or pointer, to the Document instead of the actual Document itself -- see Sections 4.2.2 and 4.3.2. Standard methods currently do not exist for remote entities to "assume" the credentials of a Client for forwarding requests to a third party. It is anticipated that print by reference will be used to access "public" Documents. Note that sophisticated methods for authenticating "proxies" are beyond the scope of this IPP/1.1 document. Because Printers typically process Jobs serially, print by reference is not seen as a serious denial-of-service threat to the referenced servers.

9.2. URIs in Operation, Job, and Printer Attributes

The "printer-uri-supported" attribute contains the Printer's URI(s). Its companion attribute, "uri-security-supported", identifies the security mechanism used for each URI listed in the "printer-uri-supported" attribute. For each Printer operation request, a Client MUST supply only one URI in the "printer-uri" operation attribute. In other words, even though the Printer supports more than one URI, the Client only interacts with the Printer using one of its URIs. This duality is not needed for Job objects, since Printers will act as the "factory" for Job objects and a given Printer will, depending on the Printer's security configuration, generate the correct URI for new Job objects.

9.3. URIs for Each Authentication Mechanism

Each URI has an authentication mechanism associated with it. If the URI is the "i-th" element of "printer-uri-supported", then the authentication mechanism is the "i-th" element of "uri-authentication-supported". For a list of possible authentication mechanisms, see Section 5.4.2.

The Printer uses an authentication mechanism to determine the name of the user performing an operation. This user is called the "authenticated user". The credibility of authentication depends on the mechanism that the Printer uses to obtain the user's name. When the authentication mechanism is 'none', all authenticated users are 'anonymous'.

During Job Creation requests, the Printer initializes the value of the "job-originating-user-name" attribute (see Section 5.3.6) to be the authenticated user. The authenticated user in this case is called the "Job owner".

If an implementation can be configured to support more than one authentication mechanism (see Section 5.4.2), then it MUST implement rules for determining equality of authenticated user names that have been authenticated via different authentication mechanisms. One possible policy is that identical names that are authenticated via different mechanisms are different. For example, a user can cancel his Job only if he uses the same authentication mechanism for both Cancel-Job and Print-Job. Another policy is that identical names that are authenticated via different mechanisms are the same if the authentication mechanism for the later operation is not less strong than the authentication mechanism for the earlier Job Creation operation. For example, a user can cancel his Job only if he uses the same or stronger authentication mechanism for Cancel-Job and Print-Job. With this second policy, a Job submitted via

'requesting-user-name' authentication could be canceled via 'digest' authentication. With the first policy, the Job could not be canceled in this way.

A Client is able to determine the authentication mechanism used to create a Job. It is the "i-th" value of the Printer's "uri-authentication-supported" attribute (see Section 5.4.2), where "i" is the index of the element of the Printer's "printer-uri-supported" attribute (see Section 5.4.1) equal to the Job's "job-printer-uri" attribute (see Section 5.3.3).

9.4. Restricted Queries

In many IPP operations, a Client supplies a list of attributes to be returned in the response. For security reasons, an IPP object can be configured not to return all attributes (or all values) that a Client requests. The Job attributes returned MAY depend on whether the requesting user is the same as the user that submitted the Job. The IPP object MAY even return none of the requested attributes. In such cases, the status returned is the same as if the object had returned all requested attributes. The Client cannot tell by such a response whether the requested attribute was present or absent in the object.

9.5. Operations Performed by Operators and Administrators

For the three Printer operations Pause-Printer, Resume-Printer, and Purge-Jobs (see Sections 4.2.7, 4.2.8, and 4.2.9), the requesting user is intended to be an Operator or Administrator of the Printer (see Section 1). Otherwise, the IPP Printer MUST reject the operation and return 'client-error-forbidden', 'client-error-not-authenticated', or 'client-error-not-authorized' as appropriate. For operations on Jobs, the requesting user is intended to be the Job owner or can be an Operator or Administrator of the Printer. The means for authorizing an Operator or Administrator of the Printer are not specified in this document.

9.6. Queries on Jobs Submitted Using Non-IPP Protocols

If the device that an IPP Printer is representing is able to accept Jobs using other Job submission protocols in addition to IPP, such an implementation SHOULD at least allow such "foreign" Jobs to be queried using Get-Jobs returning "job-id" and "job-uri" as 'unknown'. Such an implementation MAY support all of the same IPP Job attributes as for IPP Jobs. The IPP object returns the 'unknown' out-of-band value for any requested attribute of a foreign Job that is supported for IPP Jobs but not for foreign Jobs.

IPP Printers SHOULD also generate "job-id" and "job-uri" values for such foreign Jobs, if possible, so that they can be targets of other IPP operations, such as Get-Job-Attributes and Cancel-Job. Such an implementation also needs to deal with the problem of authentication of such foreign Jobs. One approach would be to treat all such foreign Jobs as belonging to users other than the user of the IPP Client. Another approach would be for the foreign Job to belong to 'anonymous' -- then only authenticated Operators or Administrators of the IPP Printer could query the foreign Jobs with an IPP request. Alternatively, if the security policy is to allow users to query other users' Jobs, then the foreign Jobs would also be visible to an End User IPP Client using Get-Jobs and Get-Job-Attributes.

10. Changes since RFC 2911

The following changes have been made since RFC 2911:

- Errata ID 364: Fixed range of "redirection" status-code values (to 0x03xx).
- Errata ID 694: Fixed range of vendor status-code values (0x0n80 to 0x0nff).
- Errata ID 3072: Reworded multiple-document-handling definition, since it also applies to Jobs with a single Document and is the only interoperable way to request uncollated copies.
- Errata ID 3365: Fixed bad 'nameWithLanguage' maximum length by referencing the 'nameWithoutLanguage' section (i.e., Section 5.1.3.1).
- Errata ID 4173: Fixed range of vendor operation codes (0x4000 to 0x7fff).
- Updated obsoleted RFC references.
- Changed the IPP/1.1 Implementor's Guide reference to RFC 3196.
- Updated Create-Job, Send-Document, and Send-URI to RECOMMENDED.
- Incorporated 'collection' attribute content from RFC 3382.
- Obsoleted all attributes and values defined in RFC 3381, as they do not interact well with the "finishings" attribute and have never been widely implemented.
- Deprecated the Purge-Jobs and Restart-Job operations, which destroy accounting information.

- Dropped type3 registration procedures.

- Changed the vendor attribute and keyword naming recommendations to use SMI Private Enterprise Numbers ("smiNNN-foo") instead of domain names.

- Split READ-ONLY Job Description and Printer Description attributes into Job Status and Printer Status attributes to match the current IANA IPP registry organization.

- Referenced all IETF and PWG IPP standards.

- Updated OPTIONAL operations, attributes, and values to RECOMMENDED for consistency with IPP 2.0, IPP Everywhere, and the IPP Implementor’s Guide v2.0.

- Removed the appendix on media names. Readers are directed to "PWG Media Standardized Names 2.0 (MSN2)" [PWG5101.1].

11. References

11.1. Normative References

[ASME-Y14.1M]
ASME Y14.1M-2012, "Metric Drawing Sheet Size and Format", March 2013.

[ISO10175] ISO/IEC 10175, "Information technology -- Text and office systems -- Document Printing Application (DPA) -- Part 1: Abstract service definition and procedures", September 1996.

[ISO10646] ISO/IEC 10646:2014, JTC1/SC2, "Information technology -- Universal Coded Character Set (UCS)", September 2014.

[ISO8859-1]
ISO/IEC 8859-1:1998, "Information technology -- 8-bit single-byte coded graphic character sets -- Part 1: Latin alphabet No. 1", April 1998.

[PWG5100.1]
Sweet, M., "IPP Finishings 2.0 (FIN)", December 2014, <http://ftp.pwg.org/pub/pwg/candidates/cs-ippfinishings20-20141219-5100.1.pdf>.

[PWG5100.11]
Hastings, T. and D. Fullman, "Internet Printing Protocol (IPP): Job and Printer Extensions -- Set 2 (JPS2)", October 2010, <http://ftp.pwg.org/pub/pwg/candidates/cs-ippjobprinterext10-20101030-5100.11.pdf>.

[PWG5100.12]
Sweet, M. and I. McDonald, "IPP Version 2.0, 2.1, and 2.2", October 2015, <http://ftp.pwg.org/pub/pwg/standards/std-ipp20-20151030-5100.12.pdf>.

[PWG5100.13]
Sweet, M., McDonald, I., and P. Zehler, "IPP: Job and Printer Extensions -- Set 3 (JPS3)", July 2012, <http://ftp.pwg.org/pub/pwg/candidates/cs-ippjobprinterext3v10-20120727-5100.13.pdf>.

[PWG5100.14]
Sweet, M., McDonald, I., Mitchell, A., and J. Hutchings, "IPP Everywhere", January 2013, <http://ftp.pwg.org/pub/pwg/candidates/cs-ippeve10-20130128-5100.14.pdf>.

[PWG5100.15]
Sweet, M., "IPP FaxOut Service", June 2014, <http://ftp.pwg.org/pub/pwg/candidates/cs-ippfaxout10-20140618-5100.15.pdf>.

[PWG5100.16]
Sweet, M., "IPP Transaction-Based Printing Extensions", November 2013, <http://ftp.pwg.org/pub/pwg/candidates/cs-ipptrans10-20131108-5100.16.pdf>.

[PWG5100.17]
Zehler, P. and M. Sweet, "IPP Scan Service (SCAN)", September 2014, <http://ftp.pwg.org/pub/pwg/candidates/cs-ippscan10-20140918-5100.17.pdf>.

[PWG5100.18]
Sweet, M. and I. McDonald, "IPP Shared Infrastructure Extensions (INFRA)", June 2015, <http://ftp.pwg.org/pub/pwg/candidates/cs-ippinfra10-20150619-5100.18.pdf>.

[PWG5100.19]
Kennedy, S., "IPP Implementor’s Guide v2.0 (IG)", August 2015, <http://ftp.pwg.org/pub/pwg/candidates/cs-ippig20-20150821-5100.19.pdf>.

[PWG5100.2]
Hastings, T. and R. Bergman, "Internet Printing Protocol (IPP): "output-bin" attribute extension", February 2001, <http://ftp.pwg.org/pub/pwg/candidates/cs-ippoutputbin10-20010207-5100.2.pdf>.

[PWG5100.3]
Ocke, K. and T. Hastings, "Internet Printing Protocol (IPP): Production Printing Attributes -- Set1", February 2001, <http://ftp.pwg.org/pub/pwg/candidates/cs-ippprodprint10-20010212-5100.3.pdf>.

[PWG5100.5]
Carney, D., Hastings, T., and P. Zehler, "Standard for The Internet Printing Protocol (IPP): Document Object", October 2003, <http://ftp.pwg.org/pub/pwg/candidates/cs-ippdocobject10-20031031-5100.5.pdf>.

[PWG5100.6]
Zehler, P., Herriot, R., and K. Ocke, "Standard for The Internet Printing Protocol (IPP): Page Overrides", October 2003, <http://ftp.pwg.org/pub/pwg/candidates/cs-ipppageoverride10-20031031-5100.6.pdf>.

[PWG5100.7]
Hastings, T. and P. Zehler, "Standard for The Internet Printing Protocol (IPP): Job Extensions", October 2003, <http://ftp.pwg.org/pub/pwg/candidates/cs-ippjobext10-20031031-5100.7.pdf>.

[PWG5100.8]
Carney, D. and H. Lewis, "Standard for Internet Printing Protocol (IPP): "-actual" attributes", March 2003, <http://ftp.pwg.org/pub/pwg/candidates/cs-ippactuals10-20030313-5100.8.pdf>.

[PWG5100.9]
McDonald, I. and C. Whittle, "Internet Printing Protocol (IPP): Printer State Extensions v1.0", July 2009, <http://ftp.pwg.org/pub/pwg/candidates/cs-ippstate10-20090731-5100.9.pdf>.

[PWG5101.1]
Sweet, M., Bergman, R., and T. Hastings, "PWG Media Standardized Names 2.0 (MSN2)", March 2013, <http://ftp.pwg.org/pub/pwg/candidates/cs-pwgmsn20-20130328-5101.1.pdf>.

[RFC20] Cerf, V., "ASCII format for network interchange", STD 80, RFC 20, DOI 10.17487/RFC0020, October 1969, <http://www.rfc-editor.org/info/rfc20>.

[RFC793] Postel, J., "Transmission Control Protocol", STD 7, RFC 793, DOI 10.17487/RFC0793, September 1981, <http://www.rfc-editor.org/info/rfc793>.

[RFC1035] Mockapetris, P., "Domain names - implementation and specification", STD 13, RFC 1035, DOI 10.17487/RFC1035, November 1987, <http://www.rfc-editor.org/info/rfc1035>.

[RFC1951] Deutsch, P., "DEFLATE Compressed Data Format Specification version 1.3", RFC 1951, DOI 10.17487/RFC1951, May 1996, <http://www.rfc-editor.org/info/rfc1951>.

[RFC1952] Deutsch, P., "GZIP file format specification version 4.3", RFC 1952, DOI 10.17487/RFC1952, May 1996, <http://www.rfc-editor.org/info/rfc1952>.

[RFC1977] Schryver, V., "PPP BSD Compression Protocol", RFC 1977, DOI 10.17487/RFC1977, August 1996, <http://www.rfc-editor.org/info/rfc1977>.

[RFC2046] Freed, N. and N. Borenstein, "Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types", RFC 2046, DOI 10.17487/RFC2046, November 1996, <http://www.rfc-editor.org/info/rfc2046>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

[RFC2818] Rescorla, E., "HTTP Over TLS", RFC 2818, DOI 10.17487/RFC2818, May 2000, <http://www.rfc-editor.org/info/rfc2818>.

[RFC3196] Hastings, T., Manros, C., Zehler, P., Kugler, C., and H. Holst, "Internet Printing Protocol/1.1: Implementor’s Guide", RFC 3196, DOI 10.17487/RFC3196, November 2001, <http://www.rfc-editor.org/info/rfc3196>.

[RFC3380] Hastings, T., Herriot, R., Kugler, C., and H. Lewis, "Internet Printing Protocol (IPP): Job and Printer Set Operations", RFC 3380, DOI 10.17487/RFC3380, September 2002, <http://www.rfc-editor.org/info/rfc3380>.

[RFC3510] Herriot, R. and I. McDonald, "Internet Printing Protocol/1.1: IPP URL Scheme", RFC 3510, DOI 10.17487/RFC3510, April 2003, <http://www.rfc-editor.org/info/rfc3510>.

[RFC3629] Yergeau, F., "UTF-8, a transformation format of ISO 10646", STD 63, RFC 3629, DOI 10.17487/RFC3629, November 2003, <http://www.rfc-editor.org/info/rfc3629>.

[RFC3805] Bergman, R., Lewis, H., and I. McDonald, "Printer MIB v2", RFC 3805, DOI 10.17487/RFC3805, June 2004, <http://www.rfc-editor.org/info/rfc3805>.

[RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, DOI 10.17487/RFC3986, January 2005, <http://www.rfc-editor.org/info/rfc3986>.

[RFC3995] Herriot, R. and T. Hastings, "Internet Printing Protocol (IPP): Event Notifications and Subscriptions", RFC 3995, DOI 10.17487/RFC3995, March 2005, <http://www.rfc-editor.org/info/rfc3995>.

[RFC3996] Herriot, R., Hastings, T., and H. Lewis, "Internet Printing Protocol (IPP): The 'ippget' Delivery Method for Event Notifications", RFC 3996, DOI 10.17487/RFC3996, March 2005, <http://www.rfc-editor.org/info/rfc3996>.

[RFC3998] Kugler, C., Lewis, H., and T. Hastings, Ed., "Internet Printing Protocol (IPP): Job and Printer Administrative Operations", RFC 3998, DOI 10.17487/RFC3998, March 2005, <http://www.rfc-editor.org/info/rfc3998>.

[RFC5051] Crispin, M., "i;unicode-casemap - Simple Unicode Collation Algorithm", RFC 5051, DOI 10.17487/RFC5051, October 2007, <http://www.rfc-editor.org/info/rfc5051>.

[RFC5234] Crocker, D., Ed., and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, DOI 10.17487/RFC5234, January 2008, <http://www.rfc-editor.org/info/rfc5234>.

[RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, DOI 10.17487/RFC5246, August 2008, <http://www.rfc-editor.org/info/rfc5246>.

[RFC5646] Phillips, A., Ed., and M. Davis, Ed., "Tags for Identifying Languages", BCP 47, RFC 5646, DOI 10.17487/RFC5646, September 2009, <http://www.rfc-editor.org/info/rfc5646>.

[RFC6838] Freed, N., Klensin, J., and T. Hansen, "Media Type Specifications and Registration Procedures", BCP 13, RFC 6838, DOI 10.17487/RFC6838, January 2013, <http://www.rfc-editor.org/info/rfc6838>.

[RFC7230] Fielding, R., Ed., and J. Reschke, Ed., "Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing", RFC 7230, DOI 10.17487/RFC7230, June 2014, <http://www.rfc-editor.org/info/rfc7230>.

[RFC7472] McDonald, I. and M. Sweet, "Internet Printing Protocol (IPP) over HTTPS Transport Binding and the 'ipps' URI Scheme", RFC 7472, DOI 10.17487/RFC7472, March 2015, <http://www.rfc-editor.org/info/rfc7472>.

[RFC7612] Fleming, P. and I. McDonald, "Lightweight Directory Access Protocol (LDAP): Schema for Printer Services", RFC 7612, DOI 10.17487/RFC7612, June 2015, <http://www.rfc-editor.org/info/rfc7612>.

[RFC7616] Shekh-Yusef, R., Ed., Ahrens, D., and S. Bremer, "HTTP Digest Access Authentication", RFC 7616, DOI 10.17487/RFC7616, September 2015, <http://www.rfc-editor.org/info/rfc7616>.

[RFC7617] Reschke, J., "The 'Basic' HTTP Authentication Scheme", RFC 7617, DOI 10.17487/RFC7617, September 2015, <http://www.rfc-editor.org/info/rfc7617>.

[RFC8010] Sweet, M. and I. McDonald, "Internet Printing Protocol/1.1: Encoding and Transport", RFC 8010, DOI 10.17487/RFC8010, January 2017, <http://www.rfc-editor.org/info/rfc8010>.

11.2. Informative References

[HTPP] Barnett, J., Carter, K., and R. deBry, "Internet Print Protocol Proposal: HTPP -- Hypertext Print Protocol (HTPP/1.0 Initial Draft)", October 1996, <ftp://ftp.pwg.org/pub/pwg/ipp/historic/htpp/overview.ps.gz>.

[IANA-CS] IANA, "Registry of Coded Character Sets", <http://www.iana.org/assignments/character-sets/>.

[IANA-MT] IANA, "Media Types", <http://www.iana.org/assignments/media-types/>.

[IANA-PEN] IANA, "Private Enterprise Numbers", <http://www.iana.org/assignments/enterprise-numbers/>.

[ISO32000] "Document management -- Portable document format -- Part 1: PDF 1.7", July 2008, <http://www.adobe.com/devnet/acrobat/pdfs/PDF32000_2008.pdf>.

[LDPA] Isaacson, S., Taylor, D., MacKay, M., Zehler, P., Hastings, T., and C. Manros, "LDPA - Lightweight Document Printing Application", Proposed Internet-Draft, October 1996, <ftp://ftp.pwg.org/pub/pwg/ipp/historic/ldpa/ldpa8.pdf.gz>.

[P1387.4] Kirk, M., "POSIX Systems Administration - Part 4: Printing Interfaces, POSIX 1387.4 D8", 1998.

[PSIS] Herriot, R., Ed., "X/Open: A Printing System Interoperability Specification (PSIS)", August 1995.

[PWG-IPP-WG] IEEE-ISTO Printer Working Group, "Internet Printing Protocol Workgroup", <http://www.pwg.org/ipp>.

[RFC959] Postel, J. and J. Reynolds, "File Transfer Protocol", STD 9, RFC 959, DOI 10.17487/RFC0959, October 1985, <http://www.rfc-editor.org/info/rfc959>.

[RFC1179] McLaughlin, L., "Line printer daemon protocol", RFC 1179, DOI 10.17487/RFC1179, August 1990, <http://www.rfc-editor.org/info/rfc1179>.

[RFC1738] Berners-Lee, T., Masinter, L., and M. McCahill, "Uniform Resource Locators (URL)", RFC 1738, DOI 10.17487/RFC1738, December 1994, <http://www.rfc-editor.org/info/rfc1738>.

[RFC2565] Herriot, R., Ed., Butler, S., Moore, P., and R. Turner, "Internet Printing Protocol/1.0: Encoding and Transport", RFC 2565, DOI 10.17487/RFC2565, April 1999, <http://www.rfc-editor.org/info/rfc2565>.

[RFC2566] deBry, R., Hastings, T., Herriot, R., Isaacson, S., and P. Powell, "Internet Printing Protocol/1.0: Model and Semantics", RFC 2566, DOI 10.17487/RFC2566, April 1999, <http://www.rfc-editor.org/info/rfc2566>.

[RFC2567] Wright, F., "Design Goals for an Internet Printing Protocol", RFC 2567, DOI 10.17487/RFC2567, April 1999, <http://www.rfc-editor.org/info/rfc2567>.

[RFC2568] Zilles, S., "Rationale for the Structure of the Model and Protocol for the Internet Printing Protocol", RFC 2568, DOI 10.17487/RFC2568, April 1999, <http://www.rfc-editor.org/info/rfc2568>.

[RFC2569] Herriot, R., Ed., Hastings, T., Jacobs, N., and J. Martin, "Mapping between LPD and IPP Protocols", RFC 2569, DOI 10.17487/RFC2569, April 1999, <http://www.rfc-editor.org/info/rfc2569>.

[RFC2579] McCloghrie, K., Ed., Perkins, D., Ed., and J. Schoenwaelder, Ed., "Textual Conventions for SMIv2", STD 58, RFC 2579, DOI 10.17487/RFC2579, April 1999, <http://www.rfc-editor.org/info/rfc2579>.

[RFC2978] Freed, N. and J. Postel, "IANA Charset Registration Procedures", BCP 19, RFC 2978, DOI 10.17487/RFC2978, October 2000, <http://www.rfc-editor.org/info/rfc2978>.

[RFC3239] Kugler, C., Lewis, H., and T. Hastings, "Internet Printing Protocol (IPP): Requirements for Job, Printer, and Device Administrative Operations", RFC 3239, DOI 10.17487/RFC3239, February 2002, <http://www.rfc-editor.org/info/rfc3239>.

[RFC3997] Hastings, T., Ed., deBry, R., and H. Lewis, "Internet Printing Protocol (IPP): Requirements for IPP Notifications", RFC 3997, DOI 10.17487/RFC3997, March 2005, <http://www.rfc-editor.org/info/rfc3997>.

[RFC4122] Leach, P., Mealling, M., and R. Salz, "A Universally Unique IDentifier (UUID) URN Namespace", RFC 4122, DOI 10.17487/RFC4122, July 2005, <http://www.rfc-editor.org/info/rfc4122>.

[RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 5226, DOI 10.17487/RFC5226, May 2008, <http://www.rfc-editor.org/info/rfc5226>.

[RFC6068] Duerst, M., Masinter, L., and J. Zawinski, "The 'mailto' URI Scheme", RFC 6068, DOI 10.17487/RFC6068, October 2010, <http://www.rfc-editor.org/info/rfc6068>.

[RFC7525] Sheffer, Y., Holz, R., and P. Saint-Andre, "Recommendations for Secure Use of Transport Layer Security (TLS) and Datagram Transport Layer Security (DTLS)", BCP 195, RFC 7525, DOI 10.17487/RFC7525, May 2015, <http://www.rfc-editor.org/info/rfc7525>.

[SWP] Moore, P. and S. Butler, "Simple Web Printing (SWP/1.0)", May 1997, <ftp://ftp.pwg.org/pub/pwg/ipp/new_PRO/swp9705.pdf>.

Appendix A. Formats for IPP Registration Proposals

In order to propose an IPP extension for registration, the proposer must submit an application to IANA by email to "iana@iana.org" or by filling out the appropriate form on the IANA web pages (http://www.iana.org). This section specifies the required information and the formats for proposing registrations of extensions to IPP as provided in Section 7 for:

1. attributes
2. type2 ’keyword’ attribute values
3. type2 ’enum’ attribute values
4. operations
5. status-code values

A.1. Attribute Registration

Type of registration: attribute

Proposed keyword name of this attribute:

Types of attributes (Document Description, Document Status, Document Template, Event Notifications, Job Description, Job Status, Job Template, Operation, Printer Description, Printer Status, Subscription Description, Subscription Status, Subscription Template):

Operations to be used if the attribute is an operation attribute:

Object (Document, Job, Printer, Subscription, etc. if bound to an object):

Attribute syntax(es) (include ’1setOf’ and range; see Section 5.2):

If attribute syntax is ’keyword’ or ’enum’, is it type1 or type2?

If this is a Printer attribute, MAY the value returned depend on "document-format"? (See Section 7.2.)

If this is a Job Template attribute, how does its specification depend on the value of the "multiple-document-handling" attribute?

Specification of this attribute (follow the style of Section 5.2):

Name of proposer:

Email address of proposer:

Note: For attributes, the IPP Designated Expert will be the point of contact and change controller for the approved registration specification, if any maintenance of the registration specification is needed.

A.2. type2 'keyword' Attribute Value Registration

Type of registration: type2 keyword attribute value

Name of attribute to which this keyword specification is to be added:

Proposed keyword name of this 'keyword' value:

Specification of this 'keyword' value (follow the style of Section 5.1.4):

Name of proposer:

Email address of proposer:

Note: For type2 keywords, the Designated Expert will be the point of contact and change controller for the approved registration specification, if any maintenance of the registration specification is needed.

A.3. type2 'enum' Attribute Value Registration

Type of registration: type2 enum attribute value

Name of attribute to which this enum specification is to be added:

Keyword symbolic name of this enum value:

Numeric value (to be assigned by the IPP Designated Expert in consultation with IANA):

Specification of this enum value (follow the style of Section 5.1.5):

Name of proposer:

Email address of proposer:

Note: For type2 enums, the Designated Expert will be the point of contact and change controller for the approved registration specification, if any maintenance of the registration specification is needed.

A.4. Operation Registration

Type of registration: operation

Proposed name of this operation:

Numeric "operation-id" value according to Section 5.4.15 (to be assigned by the IPP Designated Expert in consultation with IANA):

Object Target (Document, Job, Printer, Subscription, etc. that operation is upon):

Specification of this operation (follow the style of Section 4):

Name of proposer:

Email address of proposer:

Note: For operations, the IPP Designated Expert will be the point of contact and change controller for the approved registration specification, if any maintenance of the registration specification is needed.

A.5. Status-Code Registration

Type of registration: status-code

Keyword symbolic name of this status-code value:

Numeric value (to be assigned by the IPP Designated Expert in consultation with IANA):

Operations that this status-code can be used with:

Specification of this status-code (follow the style of Appendix B):

Name of proposer:

Email address of proposer:

Note: For status-code values, the Designated Expert will be the point of contact and change controller for the approved registration specification, if any maintenance of the registration specification is needed.

Appendix B. Status-Code Values and Suggested Status-Code Messages

This section defines status-code enum keywords and values that are used to provide semantic information on the results of an operation request. Each operation response MUST include a status-code. The response MAY also contain a status message that provides a short textual description of the status. The status-code is intended for use by automata, and the status message is intended for the human End User.

The prefix of the status keyword defines the class of response as follows:

"informational" - Request received, continuing process

"successful" - The action was successfully received, understood, and accepted

"redirection" - Further action is taken in order to complete the request

"client-error" - The request contains bad syntax or cannot be fulfilled

"server-error" - The IPP object failed to fulfill an apparently valid request

As with type2 enums, IPP status-code values are extensible. Regardless of whether all status-code values are recognized, IPP Clients MUST understand the class of any status-code, as indicated by the prefix, and treat any unrecognized response as being equivalent to the first status-code of that class, with the exception that an unrecognized response MUST NOT be cached. For example, if an unrecognized status-code of 'client-error-xxx-yyy' is received by the Client, it can safely assume that there was something wrong with its request and treat the response as if it had received a 'client-error-bad-request' status-code. The name of the enum is the suggested status message for US English.

See [PWG5100.19] for guidelines on presenting status messages to End Users.

The status-code values range from 0x0000 to 0x7fff. The value ranges for each status-code class are as follows:

"successful" - 0x0000 to 0x00ff

"informational" - 0x0100 to 0x01ff

"redirection" - 0x0300 to 0x03ff

"client-error" - 0x0400 to 0x04ff

"server-error" - 0x0500 to 0x05ff

The top half (128 values) of each range (0x0n80 to 0x0nff, for n = 0 to 5) is reserved for vendor use within each status-code class. Values 0x0600 to 0x7fff are reserved for future assignment by Standards Track documents and MUST NOT be used.

B.1. Status-Code Values

Each status-code is described below. Appendix B.2 contains a table that indicates which status-code values apply to which operations. The Implementor's Guides [RFC3196] [PWG5100.19] provide guidance for processing IPP attributes for all operations, including status-code values.

B.1.1. Informational

This class of status-code values indicates a provisional response and is to be used for informational purposes only.

There are no values defined in this document for this class of status-code values.

B.1.2. Successful Status-Code Values

This class of status-code values indicates that the Client's request was successfully received, understood, and accepted.

B.1.2.1. successful-ok (0x0000)

The request has succeeded, and no request attributes were substituted or ignored. In the case of a response to a Job Creation request, the 'successful-ok' status-code indicates that the request was successfully received and validated, and that the Job object has been created; it does not indicate that the Job has been processed. The transition of the Job object into the 'completed' state is the only indicator that the Job has been printed.

B.1.2.2. successful-ok-ignored-or-substituted-attributes (0x0001)

The request has succeeded, but some supplied (1) attributes were ignored or (2) unsupported values were substituted with supported values or were ignored in order to perform the operation without rejecting it. Unsupported attributes, attribute syntaxes, or values MUST be returned in the Unsupported Attributes group of the response for all operations. There is an exception to this rule for the query operations Get-Printer-Attributes, Get-Jobs, and Get-Job-Attributes for the "requested-attributes" operation attribute only. When the supplied values of the "requested-attributes" operation attribute are requesting attributes that are not supported, the IPP object SHOULD return the "requested-attributes" operation attribute in the Unsupported Attributes group of the response (with the unsupported values only). See Sections 4.1.7 and 4.2.1.2.

B.1.2.3. successful-ok-conflicting-attributes (0x0002)

The request has succeeded, but some supplied attribute values conflicted with the values of other supplied attributes. Either (1) these conflicting values were substituted with (supported) values or (2) the attributes were removed in order to process the Job without rejecting it. Attributes or values that conflict with other attributes and have been substituted or ignored MUST be returned in the Unsupported Attributes group of the response for all operations as supplied by the Client. See Sections 4.1.7 and 4.2.1.2.

B.1.3. Redirection Status-Code Values

This class of status-code values indicates that further action needs to be taken to fulfill the request.

There are no values defined in this document for this class of status-code values.

B.1.4. Client Error Status-Code Values

This class of status-code values is intended for cases in which the Client seems to have erred. The IPP object SHOULD return a message containing an explanation of the error situation and whether it is a temporary or permanent condition.

B.1.4.1. client-error-bad-request (0x0400)

The request could not be understood by the IPP object due to malformed syntax (such as the value of a fixed-length attribute whose length does not match the prescribed length for that attribute -- see the Implementor's Guides [RFC3196] [PWG5100.19]). The IPP application SHOULD NOT repeat the request without modifications.

B.1.4.2. client-error-forbidden (0x0401)

The IPP object understood the request but is refusing to fulfill it. Additional authentication information or authorization credentials will not help, and the request SHOULD NOT be repeated. This status-code is commonly used when the IPP object does not wish to reveal exactly why the request has been refused or when no other response is applicable.

B.1.4.3. client-error-not-authenticated (0x0402)

The request requires user authentication. The IPP Client can repeat the request with suitable authentication information. If the request already included authentication information, then this status-code indicates that authorization has been refused for those credentials. If this response contains the same challenge as the prior response and the user agent has already attempted authentication at least once, then the response message can contain relevant diagnostic information. This status-code reveals more information than 'client-error-forbidden'.

B.1.4.4. client-error-not-authorized (0x0403)

The requester is not authorized to perform the request. Additional authentication information or authorization credentials will not help, and the request SHOULD NOT be repeated. This status-code is used when the IPP object wishes to reveal that the authentication information is understandable; however, the requester is explicitly not authorized to perform the request. This status-code reveals more information than 'client-error-forbidden' and 'client-error-not-authenticated'.

B.1.4.5. client-error-not-possible (0x0404)

This status-code is used when the request is for something that cannot happen. For example, there might be a request to cancel a Job that has already been canceled or aborted by the system. The IPP Client SHOULD NOT repeat the request.

B.1.4.6. client-error-timeout (0x0405)

The Client did not produce a request within the time that the IPP object was prepared to wait. For example, a Client issued a Create-Job operation and then, after a long period of time, issued a Send-Document operation; this error status-code was returned in response to the Send-Document request (see Section 4.3.1). The IPP object might have been forced to clean up resources that had been held for the waiting additional Documents. The IPP object was forced to close the Job, since the Client took too long. The Client SHOULD NOT repeat the request without modifications.

B.1.4.7. client-error-not-found (0x0406)

The IPP object has not found anything matching the request URI. No indication is given of whether the condition is temporary or permanent. For example, a Client with an old reference to a Job (a URI) tries to cancel the Job; however, in the meantime the Job might have been completed and all record of it at the Printer has been deleted. This status-code, 'client-error-not-found', is returned indicating that the referenced Job cannot be found. This error status-code is also used when a Client supplies a URI as a reference to the Document data in either a Print-URI or Send-URI operation but the Document cannot be found.

In practice, an IPP application should avoid a "not found" situation by first querying and presenting a list of valid Printer URIs and Job URIs to the End User.

B.1.4.8. client-error-gone (0x0407)

The requested object is no longer available, and no forwarding address is known. This condition should be considered permanent. Clients with link-editing capabilities should delete references to the request URI after user approval. If the IPP object does not know or has no facility to determine whether or not the condition is permanent, the status-code 'client-error-not-found' should be used instead.

This response is primarily intended to assist the task of maintenance by notifying the recipient that the resource is intentionally unavailable and that the IPP object Administrator desires that remote links to that resource be removed. It is not necessary to mark all permanently unavailable resources as "gone" or to keep the mark for any length of time -- that is left to the discretion of the IPP object Administrator and/or Printer implementation.

B.1.4.9. client-error-request-entity-too-large (0x0408)

The IPP object is refusing to process a request because the request entity is larger than the IPP object is willing or able to process. An IPP Printer returns this status-code when it limits the size of Print Jobs and it receives a Print Job that exceeds that limit or when the attributes are so many that their encoding causes the request entity to exceed IPP object capacity.

B.1.4.10. client-error-request-value-too-long (0x0409)

The IPP object is refusing to service the request because one or more of the Client-supplied attributes have a variable-length value that is longer than the maximum length specified for that attribute. The IPP object might not have sufficient resources (memory, buffers, etc.) to process (even temporarily), interpret, and/or ignore a value larger than the maximum length. Another use of this error code is when the IPP object supports the processing of a large value that is less than the maximum length, but during the processing of the request as a whole, the object can pass the value onto some other system component that is not able to accept the large value. For more details, see the Implementor's Guides [RFC3196] [PWG5100.19].

Note: For attribute values that are URIs, this rare condition is only likely to occur when a Client has improperly submitted a request with long query information (e.g., an IPP application allows an End User to enter an invalid URI), when the Client has descended into a URI "black hole" of redirection (e.g., a redirected URI prefix that points to a suffix of itself), or when the IPP object is under attack by a Client attempting to exploit security holes present in some IPP objects using fixed-length buffers for reading or manipulating the request URI.

B.1.4.11. client-error-document-format-not-supported (0x040a)

The IPP object is refusing to service the request because the Document data is in a format, as specified in the "document-format" operation attribute, that is not supported by the Printer. This error is returned independent of the Client-supplied "ipp-attribute-fidelity" attribute. The Printer MUST return this status-code, even if there are other Job Template attributes that are not supported as well, since this error is a bigger problem than with Job Template attributes. See Sections 4.1.6.1, 4.1.7, and 4.2.1.1.

B.1.4.12. client-error-attributes-or-values-not-supported (0x040b)

In a Job Creation request, if the Printer does not support one or more attributes, attribute syntaxes, or attribute values supplied in the request and the Client supplied the "ipp-attribute-fidelity" operation attribute with the 'true' value, the Printer MUST return this status-code. The Printer MUST also return in the Unsupported Attributes group all the attributes and/or values supplied by the Client that are not supported. See Section 4.1.7. Examples would be if the request indicates 'iso-a4' media but that media type is not supported by the Printer, or if the Client supplies a Job Template attribute and the attribute itself is not even supported by the Printer. If the "ipp-attribute-fidelity" attribute is 'false', the Printer MUST ignore or substitute values for unsupported Job Template attributes and values rather than reject the request and return this status-code.

For any operation where a Client requests attributes (such as a Get-Jobs, Get-Printer-Attributes, or Get-Job-Attributes operation), if the IPP object does not support one or more of the requested attributes, the IPP object simply ignores the unsupported requested attributes and processes the request as if they had not been supplied, rather than returning this status-code. In this case, the IPP object MUST return the 'successful-ok-ignored-or-substituted-attributes' status-code and SHOULD return the unsupported attributes as values of the "requested-attributes" operation attribute in the Unsupported Attributes group (see Appendix B.1.2.2).

B.1.4.13. client-error-uri-scheme-not-supported (0x040c)

The scheme of the Client-supplied URI in a Print-URI or a Send-URI operation is not supported. See Sections 4.1.6.1 and 4.1.7.

B.1.4.14. client-error-charset-not-supported (0x040d)

For any operation, if the IPP Printer does not support the charset supplied by the Client in the "attributes-charset" operation attribute, the Printer MUST reject the operation and return this status-code, and any 'text' or 'name' attributes using the 'utf-8' charset (Section 4.1.4.1). See Sections 4.1.6.1 and 4.1.7.

B.1.4.15. client-error-conflicting-attributes (0x040e)

The request is rejected because some attribute values conflicted with the values of other attributes that this document does not permit to be substituted or ignored. The Printer MUST also return in the Unsupported Attributes group the conflicting attributes supplied by the Client. See Sections 4.1.7 and 4.2.1.2.

B.1.4.16. client-error-compression-not-supported (0x040f)

The IPP object is refusing to service the request because the Document data, as specified in the "compression" operation attribute, is compressed in a way that is not supported by the Printer. This error is returned independent of the Client-supplied "ipp-attribute-fidelity" attribute. The Printer MUST return this status-code, even if there are other Job Template attributes that are not supported as well, since this error is a bigger problem than with Job Template attributes. See Sections 4.1.6.1, 4.1.7, and 4.2.1.1.

B.1.4.17. client-error-compression-error (0x0410)

The IPP object is refusing to service the request because the Document data cannot be decompressed when using the algorithm specified by the "compression" operation attribute. This error is returned independent of the Client-supplied "ipp-attribute-fidelity" attribute. The Printer MUST return this status-code, even if there are Job Template attributes that are not supported as well, since this error is a bigger problem than with Job Template attributes. See Sections 4.1.7 and 4.2.1.1.

B.1.4.18. client-error-document-format-error (0x0411)

The IPP object is refusing to service the request because the Printer encountered an error in the Document data while interpreting it. This error is returned independent of the Client-supplied "ipp-attribute-fidelity" attribute. The Printer MUST return this status-code, even if there are Job Template attributes that are not supported as well, since this error is a bigger problem than with Job Template attributes. See Sections 4.1.7 and 4.2.1.1.

B.1.4.19. client-error-document-access-error (0x0412)

The IPP object is refusing to service the Print-URI or Send-URI request because the Printer encountered an access error while attempting to validate the accessibility of, or access to, the Document data specified in the "document-uri" operation attribute. The Printer MAY also return a specific Document access error code using the "document-access-error" operation attribute (see

Section 4.1.6.4). This error is returned independent of the Client-supplied "ipp-attribute-fidelity" attribute. The Printer MUST return this status-code, even if there are Job Template attributes that are not supported as well, since this error is a bigger problem than with Job Template attributes. See Sections 4.1.6.1 and 4.1.7.

B.1.5. Server Error Status-Code Values

This class of status-code values indicates cases in which the IPP object is aware that it has erred or is incapable of performing the request. The IPP object SHOULD include a message containing an explanation of the error situation, and whether it is a temporary or permanent condition.

B.1.5.1. server-error-internal-error (0x0500)

The IPP object encountered an unexpected condition that prevented it from fulfilling the request. This error status-code differs from 'server-error-temporary-error' in that it implies a more permanent type of internal error. It also differs from 'server-error-device-error' in that it implies an unexpected condition (unlike a paper-jam or out-of-toner problem, which is undesirable but expected). This error status-code indicates that intervention by a knowledgeable human is probably required.

B.1.5.2. server-error-operation-not-supported (0x0501)

The IPP object does not support the functionality required to fulfill the request. This is the appropriate response when the IPP object does not recognize an operation or is not capable of supporting it. See Sections 4.1.6.1 and 4.1.7.

B.1.5.3. server-error-service-unavailable (0x0502)

The IPP object is currently unable to handle the request due to temporary overloading or due to maintenance of the IPP object. The implication is that this is a temporary condition that will be alleviated after some delay. If known, the length of the delay can be indicated in the message. If no delay is given, the IPP application should handle the response as it would for a 'server-error-temporary-error' response. If the condition is more permanent, the 'client-error-gone' or 'client-error-not-found' error status-code could be used.

B.1.5.4. server-error-version-not-supported (0x0503)

The IPP object does not support or refuses to support the IPP version that was supplied as the value of the "version-number" operation parameter in the request. The IPP object is indicating that it is unable or unwilling to complete the request using the same major and minor version number as supplied in the request, other than with this error message. The error response SHOULD contain a "status-message" attribute (see Section 4.1.6.2) describing why that version is not supported and what other versions are supported by that IPP object. See Sections 4.1.6.1, 4.1.7, and 4.1.8.

The error response MUST identify in the "version-number" operation parameter the closest version number that the IPP object does support. For example, if a Client supplies version '1.0' and an IPP/1.1 object supports version '1.0', then it responds with version '1.0' in all responses to such a request. If the IPP/1.1 object does not support version '1.0', then it should accept the request and respond with version '1.1' or can reject the request and respond with this error code and version '1.1'. If a Client supplies version '1.2', the IPP/1.1 object should accept the request and return version '1.1' or can reject the request and respond with this error code and version '1.1'. See Sections 4.1.8 and 5.3.14.

B.1.5.5. server-error-device-error (0x0504)

A Printer error, such as a paper jam, occurs while the IPP object processes a Print or send operation. The response contains the true Job status (the values of the "job-state" and "job-state-reasons" attributes). Additional information can be returned in the OPTIONAL "job-state-message" attribute value or in the OPTIONAL status message that describes the error in more detail. This error status-code is only returned in situations where the Printer is unable to accept the Job Creation request because of such a device error. For example, if the Printer is unable to spool and can only accept one Job at a time, the reason it might reject a Job Creation request is that the Printer currently has a paper jam. In many cases, however, where the Printer can accept the request even though the Printer has some error condition, the 'successful-ok' status-code will be returned. In such a case, the Client would look at the returned Job object attributes or later query the Printer to determine its state and state reasons.

B.1.5.6. server-error-temporary-error (0x0505)

A temporary error such as a buffer-full write error, a memory overflow (i.e., the Document data exceeds the memory of the Printer), or a disk-full condition, occurs while the IPP Printer processes an operation. The Client MAY try the unmodified request again at some later point in time with an expectation that the temporary internal error condition has been cleared. Alternatively, as an implementation option, a Printer MAY delay the response until the temporary condition is cleared so that no error is returned.

B.1.5.7. server-error-not-accepting-jobs (0x0506)

This is a temporary error indicating that the Printer is not currently accepting Jobs because the Administrator has set the value of the Printer's "printer-is-accepting-jobs" attribute to 'false' (by means outside the scope of this IPP/1.1 document).

B.1.5.8. server-error-busy (0x0507)

This is a temporary error indicating that the Printer is too busy processing Jobs and/or other requests. The Client SHOULD try the unmodified request again at some later point in time with an expectation that the temporary busy condition will have been cleared.

B.1.5.9. server-error-job-canceled (0x0508)

This is an error indicating that the Job has been canceled by an Operator or the system while the Client was transmitting the data to the IPP Printer. If a "job-id" attribute and a "job-uri" attribute had been created, then they are returned in the Print-Job, Send-Document, or Send-URI response as usual; otherwise, no "job-id" and "job-uri" attributes are returned in the response.

B.1.5.10. server-error-multiple-document-jobs-not-supported (0x0509)

The IPP object does not support multiple Documents per Job, and a Client attempted to supply Document data with a second Send-Document or Send-URI operation.

B.2. Status-Code Values for IPP Operations

PJ = Print-Job, PU = Print-URI, CJ = Create-Job, SD = Send-Document, SU = Send-URI, V = Validate-Job, GA = Get-Job-Attributes and Get-Printer-Attributes, GJ = Get-Jobs, C = Cancel-Job

	IPP Operations								
IPP Status Keyword	PJ	PU	CJ	SD	SU	V	GA	GJ	C
successful-ok	x	x	x	x	x	x	x	x	x
successful-ok-ignored-or-substituted-attributes	x	x	x	x	x	x	x	x	x
successful-ok-conflicting-attributes	x	x	x	x	x	x	x	x	x
client-error-bad-request	x	x	x	x	x	x	x	x	x
client-error-forbidden	x	x	x	x	x	x	x	x	x
client-error-not-authenticated	x	x	x	x	x	x	x	x	x
client-error-not-authorized	x	x	x	x	x	x	x	x	x
client-error-not-possible	x	x	x	x	x	x	x	x	x
client-error-timeout				x	x				
client-error-not-found	x	x	x	x	x	x	x	x	x
client-error-gone	x	x	x	x	x	x	x	x	x
client-error-request-entity-too-large	x	x	x	x	x	x	x	x	x
client-error-request-value-too-long	x	x	x	x	x	x	x	x	x
client-error-document-format-not-supported	x	x		x	x	x	x		
client-error-attributes-or-values-not-supported	x	x	x	x	x	x	x	x	x
client-error-uri-scheme-not-supported		x			x				
client-error-charset-not-supported	x	x	x	x	x	x	x	x	x
client-error-conflicting-attributes	x	x	x	x	x	x	x	x	x
client-error-compression-not-supported	x	x		x	x	x			
client-error-compression-error	x	x		x	x				
client-error-document-format-error	x	x		x	x				
client-error-document-access-error		x			x				
server-error-internal-error	x	x	x	x	x	x	x	x	x
server-error-operation-not-supported		x	x	x	x				
server-error-service-unavailable	x	x	x	x	x	x	x	x	x
server-error-version-not-supported	x	x	x	x	x	x	x	x	x
server-error-device-error	x	x	x	x	x				
server-error-temporary-error	x	x	x	x	x				
server-error-not-accepting-jobs	x	x	x			x			
server-error-busy	x	x	x	x	x	x	x	x	x
server-error-job-canceled	x			x	x				
server-error-multiple-document-jobs-not-supported				x	x				

HJ = Hold-Job, RJ = Release-Job, RS = Restart-Job,
PP = Pause-Printer, RP = Resume-Printer, PJ = Purge-Jobs

	IPP Operations (cont.)					
IPP Status Keyword	HJ	RJ	RS	PP	RP	PJ
successful-ok	x	x	x	x	x	x
successful-ok-ignored-or-substituted-attributes	x	x	x	x	x	x
successful-ok-conflicting-attributes	x	x	x	x	x	x
client-error-bad-request	x	x	x	x	x	x
client-error-forbidden	x	x	x	x	x	x
client-error-not-authenticated	x	x	x	x	x	x
client-error-not-authorized	x	x	x	x	x	x
client-error-not-possible	x	x	x	x	x	x
client-error-timeout						
client-error-not-found	x	x	x	x	x	x
client-error-gone	x	x	x	x	x	x
client-error-request-entity-too-large	x	x	x	x	x	x
client-error-request-value-too-long	x	x	x	x	x	x
client-error-document-format-not-supported						
client-error-attributes-or-values-not-supported	x	x	x	x	x	x
client-error-uri-scheme-not-supported						
client-error-charset-not-supported	x	x	x	x	x	x
client-error-conflicting-attributes	x	x	x	x	x	x
client-error-compression-not-supported						
client-error-compression-error						
client-error-document-format-error						
client-error-document-access-error						
server-error-internal-error	x	x	x	x	x	x
server-error-operation-not-supported	x	x	x	x	x	x
server-error-service-unavailable	x	x	x	x	x	x
server-error-version-not-supported	x	x	x	x	x	x
server-error-device-error						
server-error-temporary-error	x	x	x	x	x	x
server-error-not-accepting-jobs						
server-error-busy	x	x	x	x	x	x
server-error-job-canceled						
server-error-multiple-document-jobs-not-supported						

Appendix C. Processing IPP Attributes

When submitting a Print Job to a Printer, the IPP Model allows a Client to supply operation and Job Template attributes along with the Document data. These Job Template attributes in the Job Creation request affect the rendering, production, and finishing of the Documents in the Job. Similar types of instructions can also be contained in the Document data itself. In addition, the Printer has a set of attributes that describe what rendering and finishing processes are supported by that Printer. This model, which allows for flexibility and power, also introduces the potential that Client-supplied attributes can conflict with either:

- what the implementation is capable of realizing (i.e., what the Printer supports), or
- the instructions embedded within the Document data itself.

The following sections describe how these two types of conflicts are handled in the IPP Model.

C.1. Fidelity

If there is a conflict between what the Client requests and what a Printer supports, the Client can request one of two possible conflict-handling mechanisms:

1) either reject the Job, since the Job cannot be processed exactly as specified, or
2) allow the Printer to make any changes necessary to proceed with processing the Job the best it can.

In the first case, the Client is indicating the following to the Printer: "Print the Job exactly as specified with no exceptions, and if that can’t be done, don’t even bother printing the Job at all." In the second case, the Client is indicating the following to the Printer: "It is more important to make sure the Job is printed rather than be processed exactly as specified; just make sure the Job is printed even if some Client-supplied attributes need to be changed or ignored."

The IPP Model accounts for this situation by introducing an "ipp-attribute-fidelity" attribute.

In a Job Creation request, "ipp-attribute-fidelity" is a boolean operation attribute that MAY be supplied by the Client. The value 'true' indicates that total fidelity to Client-supplied Job Template attributes and values is required. The Client is requesting that the Job be printed exactly as specified, and if that is not possible, then the Job MUST be rejected rather than processed incorrectly. The value 'false' indicates that a reasonable attempt to print the Job is acceptable. If a Printer does not support some of the Client-supplied Job Template attributes or values, the Printer MUST ignore or replace them with supported values. The Printer can choose to substitute the default value associated with that attribute or use some other supported value that is similar to the unsupported requested value. For example, if a Client supplies a "media" value of 'na_letter_8.5x11in', the Printer can choose to substitute 'iso_a4_210x297mm' rather than a default value of 'na_personal_3.625x6.5in'. If the Client does not supply the "ipp-attribute-fidelity" attribute, the Printer assumes a value of 'false'.

Each Printer implementation MUST support both types of "fidelity" printing (that is, whether the Client supplies a value of 'true' or 'false'):

- If the Client supplies 'false' or does not supply the attribute, the Printer MUST always accept the request by ignoring unsupported Job Template attributes and by substituting unsupported values of supported Job Template attributes with supported values.

- If the Client supplies 'true', the Printer MUST reject the request if the Client supplies unsupported Job Template attributes.

Since a Client can always query a Printer to find out exactly what is and is not supported, "ipp-attribute-fidelity" set to 'false' is useful when:

1) The End User uses a command line interface to request attributes that might not be supported.

2) In a GUI context, if the End User expects the Job might be moved to another Printer and prefers a suboptimal result to nothing at all.

3) The End User just wants something reasonable in lieu of nothing at all.

C.2. Page Description Language (PDL) Override

If there is a conflict between the value of an IPP Job Template attribute and a corresponding instruction in the Document data, the value of the IPP attribute SHOULD take precedence over the Document instruction. Consider the case where a previously formatted file of Document data is sent to an IPP Printer. In this case, if the Client supplies any attributes at Job submission time, the Client desires that those attributes override the embedded instructions. Consider the case where a previously formatted Document has embedded in it commands to load 'iso-a4' media. However, the Document is passed to an End User that only has access to a Printer with 'na-letter' media loaded. That End User most likely wants to submit that Document to an IPP Printer with the "media" Job Template attribute set to 'na-letter'. Attributes supplied at Job submission time should take precedence over the embedded PDL instructions. However, until companies that supply Document data interpreters allow a way for external IPP attributes to take precedence over embedded Job production instructions, a Printer might not be able to support the semantics that IPP attributes override the embedded instructions.

The IPP Model accounts for this situation by introducing a "pdl-override-supported" attribute that describes the Printer's capabilities to override instructions embedded in the PDL data stream. The value of the "pdl-override-supported" attribute is configured by means outside the scope of this IPP/1.1 document.

This REQUIRED Printer attribute takes on the following values:

- 'attempted': This value indicates that the Printer attempts to make the IPP attribute values take precedence over embedded instructions in the Document data; however, there is no guarantee.

- 'not-attempted': This value indicates that the Printer makes no attempt to make the IPP attribute values take precedence over embedded instructions in the Document data.

At Job processing time, an implementation that supports the value of 'attempted' might do one of several different actions:

1) Generate an Output-Device-specific command sequence to realize the feature represented by the IPP attribute value.

2) Parse the Document data itself and replace the conflicting embedded instruction with a new embedded instruction that matches the intent of the IPP attribute value.

3) Indicate to the Printer that external supplied attributes take precedence over embedded instructions and then pass the external IPP attribute values to the Document data interpreter.

4) Anything else that allows for the semantics that IPP attributes override embedded Document data instructions.

Since 'attempted' does not offer any type of guarantee, even though a given Printer might not do a very "good" job of attempting to ensure that IPP attributes take a higher precedence over instructions embedded in the Document data, it would still be a conforming implementation.

At Job processing time, an implementation that supports the value of 'not-attempted' might do one of the following actions:

1) Simply prepend the Document data with the PDL instruction that corresponds to the Client-supplied PDL attribute, such that if the Document data also has the same PDL instruction it will override what the Printer prepended. In other words, this implementation is using the same implementation semantics for the Client-supplied IPP attributes as for the Printer defaults.

2) Parse the Document data and replace the conflicting embedded instruction with a new embedded instruction that approximates, but does not match, the semantic intent of the IPP attribute value.

Note: The "ipp-attribute-fidelity" attribute applies to the Printer's ability to either accept or reject other unsupported Job Template attributes. In other words, if "ipp-attribute-fidelity" is set to 'true', a Job is accepted if and only if the Client-supplied Job Template attributes and values are supported by the Printer. Whether these attributes actually affect the processing of the Job when the Document data contains embedded instructions depends on the ability of the Printer to override the instructions embedded in the Document data with the semantics of the IPP attributes. If the Document data attributes can be overridden ("pdl-override-supported" set to 'attempted'), the Printer makes an attempt to use the IPP attributes when processing the Job. If the Document data attributes cannot be overridden ("pdl-override-supported" set to 'not-attempted'), the Printer makes no attempt to override the embedded Document data instructions with the IPP attributes when processing the Job, and hence, the IPP attributes can fail to affect the Job processing and output when the corresponding instruction is embedded in the Document data.

C.3. Using Job Template Attributes during Document Processing

The Printer uses some of the Job's Job Template attributes during the processing of the Document data associated with that Job. These include, but are not limited to, "orientation-requested", "number-up", "sides", "media", and "copies". The processing of each Document in a Job object MUST follow the steps below. These steps are intended only to identify when and how attributes are to be used in processing Document data; any alternative steps that accomplish the same effect can be used to implement this specification document.

1. Using the Client-supplied "document-format" attribute or some form of Document format detection algorithm (if the value of "document-format" is not specific enough), determine whether the Document data has already been formatted for printing. If the Document data has been formatted, then go to step 2. Otherwise, the Document data MUST be formatted. The formatting detection algorithm is implementation defined and is not specified by this document. The formatting of the Document data uses the "orientation-requested" attribute to determine how the formatted print data should be placed on an Input Page; see Section 5.2.10 for details.

2. The Document data is a set of Input Pages in a known media type. The "page-ranges" attribute is used to select, as specified in Section 5.2.7, a sub-sequence of the pages in the print-stream that are to be processed and imaged.

3. The input for this step is a sequence of Input Pages. This step is controlled by the "number-up" attribute. If the value of "number-up" is N, then during the processing of the Input Pages each N Input Pages are positioned, as specified in Section 5.2.9, to create a single Impression. If a given Document does not have N more Input Pages, then the completion of the Impression is controlled by the "multiple-document-handling" attribute as described in Section 5.2.4; when the value of this attribute is 'single-document' or 'single-document-new-sheet', the Input Pages of Document data from subsequent Documents are used to complete the Impression.

The size (scaling), position (translation), and rotation of the Input Pages on the Impression are implementation defined. Note that during this process the Input Pages can be rendered to a form suitable for placing on the Impression; this rendering is controlled by the values of the "printer-resolution" and "print-quality" attributes as described in Sections 5.2.12 and 5.2.13. In the case where N = 1, the Impression is nearly the same as the Input Page; the differences

would only be in the size, position, and rotation of the Input Page and/or any decoration, such as a frame for the page, that is added by the implementation.

1. The collection of Impressions is placed, in sequence, onto sides of the Media Sheets. This placement is controlled by the "sides" attribute and the orientation of the Input Page, as described in Section 5.2.8. The orientation of the Input Pages affects the orientation of the Impression; for example, if "number-up" equals 2, then, typically, two portrait Input Pages become one landscape Impression. Note that the placement of Impressions onto Media Sheets is also controlled by the "multiple-document-handling" attribute as described in Section 5.2.4.

2. The "copies" and "multiple-document-handling" attributes are used to determine how many copies of each Media Sheet are printed and in what order. See Sections 5.2.4 and 5.2.5 for details.

3. When the correct number of copies are created, the Media Sheets are finished according to the values of the "finishings" attribute as described in Section 5.2.6. Note that sometimes finishing processes can require manual intervention to perform the finishing processes on the copies, especially uncollated copies. This document allows any or all of the processing steps to be performed automatically or manually, at the discretion of the Printer.

Appendix D. Generic Directory Schema

This section defines a generic schema for an entry in a directory service. Implementations of this schema are defined by "Lightweight Directory Access Protocol (LDAP): Schema for Printer Services" [RFC7612] and "IPP Everywhere" [PWG5100.14]. A directory service is a means by which service users can locate service providers. In IPP environments, this means that IPP Printers can be registered (either automatically or with the help of an Administrator) as entries of type Printer in the directory using an implementation-specific mechanism such as entry attributes, entry type fields, specific branches, etc. Directory Clients can search or browse for entries of type Printer. Clients use the directory service to find entries based on naming, organizational contexts, or filtered searches on attribute values of entries. For example, a Client can find all Printers in the "Local Department" context. Authentication and authorization are also often part of a directory service so that an Administrator can place limits on End Users so that they are only allowed to find entries to which they have certain access rights. IPP itself does not require any specific directory service protocol or provider.

Note: Some directory implementations allow for the notion of "aliasing". That is, one directory entry object can appear as multiple directory entry objects with different names for each object. In each case, each alias refers to the same directory entry object, which refers to a single IPP Printer.

The generic schema is a subset of IPP Printer Job Template and Printer Description attributes (Sections 5.2 and 5.4). These attributes are identified as either RECOMMENDED or OPTIONAL for the directory entry itself. This conformance labeling is NOT the same conformance labeling applied to the attributes of IPP Printer objects. The conformance labeling in this appendix is intended to apply to directory templates and to IPP Printer implementations that subscribe by adding one or more entries to a directory. RECOMMENDED attributes SHOULD be associated with each directory entry. OPTIONAL attributes MAY be associated with the directory entry (if known or supported). In addition, all directory entry attributes SHOULD reflect the current attribute values for the corresponding Printer.

As much as possible, the names of attributes in directory schema and entries SHOULD be the same as the IPP Printer attribute names as shown.

In order to bridge between the directory service and the IPP Printer, one of the RECOMMENDED directory entry attributes is the Printer's "printer-uri-supported" attribute. The directory Client queries the "printer-uri-supported" attribute (or its equivalent) in the directory entry, and then the IPP Client addresses the IPP Printer using one of its URIs. The "uri-security-supported" attribute identifies the protocol (if any) used to secure a channel.

The attributes in Table 23 define the generic schema for directory entries of type Printer.

Attribute	Conformance	Section
charset-supported	OPTIONAL	Section 5.4.18
color-supported	RECOMMENDED	Section 5.4.26
compression-supported	RECOMMENDED	Section 5.4.32
document-format-supported	RECOMMENDED	Section 5.4.22
finishings-supported	OPTIONAL	Section 5.2.6

generated-natural-language-supported	OPTIONAL	Section 5.4.20
ipp-versions-supported	RECOMMENDED	Section 5.4.14
media-supported	RECOMMENDED	Section 5.2.11
multiple-document-jobs-supported	OPTIONAL	Section 5.4.16
number-up-supported	OPTIONAL	Section 5.2.9
pages-per-minute-color	OPTIONAL	Section 5.4.37
pages-per-minute	OPTIONAL	Section 5.4.36
print-quality-supported	OPTIONAL	Section 5.2.13
printer-info	OPTIONAL	Section 5.4.6
printer-location	RECOMMENDED	Section 5.4.5
printer-make-and-model	RECOMMENDED	Section 5.4.9
printer-more-info	OPTIONAL	Section 5.4.7
printer-name	RECOMMENDED	Section 5.4.4
printer-resolution-supported	OPTIONAL	Section 5.2.12
printer-uri-supported	RECOMMENDED	Section 5.4.1
sides-supported	RECOMMENDED	Section 5.2.8
uri-authentication-supported	RECOMMENDED	Section 5.4.2
uri-security-supported	RECOMMENDED	Section 5.4.3

Table 23: Attributes in Directory Entries

Acknowledgements

The authors would like to acknowledge the following individuals for their contributions to the original IPP/1.1 specifications:

Roger deBry, Tom Hastings (original RFC 2911 editor), Robert Herriot, Scott A. Isaacson, Kirk Ocke, Patrick Powell, and Peter Zehler

Authors' Addresses

Michael Sweet
Apple Inc.
1 Infinite Loop
MS 111-HOMC
Cupertino, CA 95014
United States of America

Email: msweet@apple.com

Ira McDonald
High North, Inc.
PO Box 221
Grand Marais, MI 49839
United States of America

Phone: +1 906-494-2434
Email: blueroofmusic@gmail.com