INTERNET-DRAFT -- BakeOff2 issue 1-36 resolutions are flagged as Issue nn and can be searched for
draft-ietf-ipp-model-v11-02~~11~~.txt

R. deBry
Utah Valley State College
T. Hastings (editor)
Xerox Corporation
R. Herriot
~~Sun Microsystems~~Xerox Corporation
S. Isaacson
Novell, Inc.
P. Powell
Astart Technologies

~~November 16,~~May 10, 1998

Internet Printing ~~Protocol/1.0:~~Protocol/1.1: Model and Semantics

Status of this Memo

This document is an Internet-Draft and is in full conformance with all provisions of Section 10 of [RFC2026]. Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress".

~~To learn the current status of any Internet-Draft, please check the "1id-abstracts.txt" listing contained in the~~The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt

The list of Internet-Draft Shadow Directories ~~on ftp.is.co.za (Africa), nic.nordu.net (Europe), munnari.oz.au (Pacific Rim), ftp.ietf.org (US East Coast), or ftp.isi.edu (US West Coast).~~can be accessed as http://www.ietf.org/shadow.html.

Abstract

This document is one of a set of documents, which together describe all aspects of a new Internet Printing Protocol (IPP). IPP is an application level protocol that can be used for distributed printing using Internet tools and technologies. This document describes a simplified model consisting of abstract objects, their attributes, and their operations that is independent of encoding and transport. The model consists of a Printer and a Job object. A Job optionally supports multiple documents. IPP ~~1.0~~1.1 semantics allow end-users and operators to query printer capabilities, submit print jobs, inquire about the status of print jobs and printers, cancel, hold, release, and ~~and cancel~~restart print jobs. IPP 1.1 semantics

allow operators to pause, resume, and purge (jobs from) Printer objects. This document also addresses security, internationalization, and directory issues.

The full set of IPP documents includes:

Design Goals for an Internet Printing Protocol ~~[IPP-REQ]~~[RFC2567]
Rationale for the Structure and Model and Protocol for the Internet Printing Protocol ~~[IPP-RAT]~~[RFC2568]
Internet Printing ~~Protocol/1.0:~~Protocol/1.1: Model and Semantics (this document)
Internet Printing ~~Protocol/1.0:~~Protocol/1.1: Encoding and Transport [IPP-PRO]
Internet Printing ~~Protocol/1.0:~~Protocol/1.1: Implementer's Guide [IPP-IIG]
Mapping between LPD and IPP Protocols ~~[IPP LPD]~~[RFC2569]

The "Design Goals for an Internet Printing Protocol" document takes a broad look at distributed printing functionality, and it enumerates real-life scenarios that help to clarify the features that need to be included in a printing protocol for the Internet. It identifies requirements for three types of users: end users, operators, and administrators. It calls out a subset of end user requirements that are satisfied in IPP/1.0. Operator and administrator requirements are out of scope for version 1.0. A few OPTIONAL operator operations have been added to IPP/1.1.

The "Rationale for the Structure and Model and Protocol for the Internet Printing Protocol" document describes IPP from a high level view, defines a roadmap for the various documents that form the suite of IPP specifications, and gives background and rationale for the IETF working group's major decisions.

The "Internet Printing ~~Protocol/1.0:~~Protocol/1.1: Encoding and Transport" document is a formal mapping of the abstract operations and attributes defined in the model document onto HTTP/1.1. It defines the encoding rules for a new Internet MIME media type called "application/ipp". This document also defines the rules for transporting over HTTP a message body whose Content-Type is "application/ipp". This document defines a new scheme named 'ipp' for identifying IPP printers and jobs. Finally, this document defines interoperability rules for supporting IPP/1.0 clients. Issue 33

The "Internet Printing ~~Protocol/1.0:~~Protocol/1.1: Implementer's Guide" document gives insight and advice to implementers of IPP clients and IPP objects. It is intended to help them understand ~~IPP/1.0~~IPP/1.1 and some of the considerations that may assist them in the design of their client and/or IPP object implementations. For example, a typical order of processing requests is given, including error checking. Motivation for some of the specification decisions is also included.

The "Mapping between LPD and IPP Protocols" document gives some advice to implementers of gateways between IPP and LPD (Line Printer Daemon) implementations.

Table of Contents

1. Introduction

The Internet Printing Protocol (IPP) is an application level protocol that can be used for distributed printing using Internet tools and technologies. IPP version ~~1.0 (IPP/1.0)~~1.1 (IPP/1.1) focuses only on end user functionality. This document is just one of a suite of documents that fully define IPP. The full set of IPP documents includes:

Design Goals for an Internet Printing Protocol ~~[IPP-REQ]~~[RFC2567]
Rationale for the Structure and Model and Protocol for the Internet Printing Protocol ~~[IPP-RAT]~~[RFC2568]
Internet Printing ~~Protocol/1.0:~~Protocol/1.1: Model and Semantics (this document)
Internet Printing ~~Protocol/1.0:~~Protocol/1.1: Encoding and Transport [IPP-PRO]
Internet Printing ~~Protocol/1.0:~~Protocol/1.1: Implementer's Guide [IPP-IIG]
Mapping between LPD and IPP Protocols ~~[IPP-LPD]~~[RFC2569]

Anyone reading these documents for the first time is strongly encouraged to read the IPP documents in the above order.

This document is laid out as follows:

- The rest of Section 1 is an introduction to the IPP simplified model for distributed printing.
- Section 2 introduces the object types covered in the model with their basic behaviors, attributes, and interactions.
- Section 3 defines the operations included in ~~IPP/1.0.~~IPP/1.1. IPP operations are synchronous, therefore, for each operation, there is a both request and a response.
- Section 4 defines the attributes (and their syntaxes) that are used in the model.
- Sections 5 - 6 summarizes the implementation conformance requirements for objects that support the protocol and IANA considerations, respectively.
- Sections 7 - 12 cover the Internationalization and Security considerations as well as References, Intellectual Property Notice, Copyright Notice, Author contact information, and Formats for Registration Proposals.
- Sections 13 - 15 are appendices that cover Terminology, Status Codes and Messages, and "media" keyword values.

> Note: This document uses terms such as "attributes", "keywords", and "support". These terms have special meaning and are defined in the model terminology section 12.2. Capitalized terms, such as MUST, MUST NOT, REQUIRED, SHOULD, SHOULD NOT, MAY, NEED NOT, and OPTIONAL, have special meaning relating to conformance. These terms are defined in section 12.1 on conformance terminology, most of which is taken from RFC 2119 [RFC2119].

- Section 16 is an appendix that helps to clarify the effects of interactions between related attributes and their values.
- Section 17 is an appendix that enumerates the subset of Printer attributes that form a generic directory schema. These attributes are useful when registering a Printer so that a client can find the Printer not just by name, but by filtered searches as well.

- Section 18 is an appendix~~that provides a Change History~~ summarizing the ~~clarification and changes that might affect an implementation since the June 30, 1998 draft.~~additions and changes from the IPP/1.0 "Model and Semantics" specification [RFC2566] to make this IPP/1.1 document.

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 model and semantics document describes a simple, abstract model for IPP even though the underlying configurations may 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 model document does not include features, interfaces, and relationships that are beyond the scope of the first version of IPP ~~(IPP/1.0). IPP/1.0~~(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 version ~~1.0 (IPP/1.0) focuses only on end user functionality.~~

1.1 (IPP/1.1) focuses primarily on end user functionality with a few additional OPTIONAL operator operations.

The ~~IPP/1.0~~IPP/1.1 model encapsulates the important components of distributed printing into two object types:

- Printer (Section 2.1)
- Job (Section 2.2)

Each object type has an associated set of operations (see section 3) and attributes (see section 4).

It is important, however, to understand that in real system implementations (which lie underneath the abstracted ~~IPP/1.0~~IPP/1.1 model), there are other components of a print service which are not explicitly defined in the ~~IPP/1.0~~IPP/1.1 model. The following figure illustrates where ~~IPP/1.0~~IPP/1.1 fits with respect to these other components.

```
                             +--------------+
                             |  Application |
                   o         +. . . . . . . |
                  \|/        |   Spooler    |
                  / \        +. . . . . . . |   +---------+
                End-User     | Print Driver |---|  File   |
      +-----------+ +-----+  +------+-------+   +----+----+
      |  Browser  | | GUI |         |                |
      +-----+-----+ +--+--+         |                |
            |          |            |                |
            |      +---+------------+---+            |
N   D   S   |      |      IPP Client    |------------+
O   I   E   |      +---------+----------+
T   R   C   |                |
I   E   U   |
F   C   R   -------------- Transport ------------------
I   T   I
C   O   T                    |         --+
A   R   Y           +--------+--------+  |
T   Y               |    IPP Server   |  |
I                   +--------+--------+  |
O                            |           |
N                   +-----------------+  | IPP Printer
                    |  Print Service  |  |
                    +-----------------+  |
                             |         --+
                    +-----------------+
                    | Output Device(s)|
                    +-----------------+
```

An IPP Printer object 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. Printer objects are optionally registered as entries in a directory where end users find and select them based on some sort of filtered and context based searching mechanism (see section 16). 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, 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 object itself rather than with the entry in the directory which only represents the Printer object.

IPP clients implement the IPP protocol on the client side and give end users (or programs running on behalf of end users) the ability to query Printer objects 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. The Printer objects may be embedded in an output device or may be implemented on a host on the network that communicates with an output device.

When a job is submitted to the Printer object and the Printer object validates the attributes in the submission request, the Printer object creates a new Job object. The end user then interacts with this new Job object to query its status and monitor the progress of the job. ~~End users may also cancel the print job~~An end user can also cancel their print jobs by using the Job object's Cancel-Job operation. An end-user can also hold, release, and restart their print jobs using the Job object's OPTIONAL Hold-Job, Release-Job, and Restart-Job operations, if implemented.

A privileged operator or administrator of a Printer object 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 additional privileged operator or administrator of a Printer object can pause, resume, or purge (jobs from) a Printer object using the OPTIONAL Pause-Printer, Resume-Printer, and Purge-Jobs operations, if implemented.

The notification service is out of scope for ~~IPP/1.0,~~this IPP/1.1 specification, but using such a notification service, the end user is able to register for and receive Printer specific and Job specific events. An end user can query the status of Printer objects and can follow the progress of Job objects by polling using the Get-Printer-Attributes, ~~Printer-Attributes,~~ Get-Jobs, and Get-Job-Attributes operations.

2. IPP Objects

The ~~IPP/1.0~~IPP/1.1 model introduces objects of type 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 may 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.
- "OPTIONAL": each object MAY support the attribute.

There is no such similar labeling of attribute values. However, if an implementation supports an attribute, it MUST support at least one of the possible values for that attribute.

2.1 Printer Object

The major component of the ~~IPP/1.0~~IPP/1.1 model is the Printer object. A Printer object implements the server-side of the ~~IPP/1.0~~IPP/1.1 protocol. Using the protocol, end users may query the attributes of the Printer object and submit print jobs to the Printer object. The actual implementation components behind the Printer abstraction may 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 3 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 4.2)
- "printer-description" attributes: These attributes describe the Printer object's identification, state, location, references to other sources of information about the Printer object, etc. (see section 4.4)

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 may or may not be capable of spooling jobs
   3b) The associated output devices may or may not support IPP

The following figures show some examples of how Printer objects can be realized on top of various distributed printing configurations. The embedded case below represents configurations 1 and 2. The hosted and fan-out figures below represent configurations 3a and 3b.

In this document the term "client" refers to a software entity that sends IPP operation request to an IPP Printer object 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 and/or

2. a component of a print server that communicates (using IPP operations) with 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 object MAY be:

1. (embedded) software that controls a device

2. part of a print server that accepts IPP operation requests and, in turn, sends operation requests using (the IPP or other) protocol to one or more networked device(s).

```
Legend:

##### indicates a Printer object which is
      either embedded in an output device or is
      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:
                                          output device
                                        +---------------+
 O   +--------+                         |  ###########  |
/|\  | client |------------IPP------------># Printer #  |
/ \  +--------+                         |  # Object  #  |
                                        |  ###########  |
                                        +---------------+


hosted printer:
                                        +---------------+
 O   +--------+        ###########      |               |
/|\  | client |--IPP--># Printer #-any->| output device |
/ \  +--------+        # Object  #      |               |
                       ###########      +---------------+



                                         +---------------+
fan out:                                 |               |
                                     +-->| output device |
                                 any/    |               |
 O   +--------+      ###########   /     +---------------+
/|\  | client |-IPP-># Printer #--*
/ \  +--------+      # Object  #   \     +---------------+
                     ########### any\    |               |
                                     +-->| output device |
                                         |               |
                                         +---------------+
```

2.2 Job Object

A Job object is used to model a print job. A Job object contains documents. The information required to create a Job object is sent in a create request from the end user via an IPP Client to the Printer object.

The Printer object validates the create request, and if the Printer object accepts the request, the Printer object creates the new Job object. Section 3 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 which are intended to override any Printer object defaults and/or instructions embedded within the document data. (See section 4.2)
- "job-description" attributes: These attributes describe the Job object's identification, state, size, etc. The client supplies some of these attributes, and the Printer object generates others. (See section 4.3)

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 object (typically a Page Description Language - PDL), or
- a reference to such a stream of document data

In ~~IPP/1.0,~~IPP/1.1, a document is not modeled as an IPP object, therefore it has no object identifier or associated attributes. 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.

2.3 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 which "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 object contains zero or more Job objects.

A Job object is contained by exactly one Printer object, however the identical document data associated with a Job object could be sent to either the same or a different Printer object. In this case, a second Job object would be created which would be almost identical to the first Job object, however it would have new (different) Job object identifiers (see section 2.4).

A Job object is either 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 Job objects. If the contained document is just a reference to a stream of document data, other documents (in the same or different Job object(s)) may contain the same reference.

2.4 Object Identity

All Printer and Job objects are identified by a Uniform Resource Identifier (URI) [RFC2396] so that they can be persistently and unambiguously referenced. The notion of a URI is a useful concept, however, until the notion of URI is more stable (i.e., defined more completely and deployed more widely), it is expected that the URIs used for IPP objects will actually be URLs [RFC2396]. 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 Printer objects to either support or not support authentication and/or message privacy using ~~SSL3 [SSL]~~TLS [TLS] (the mechanism for security configuration is outside the scope of ~~IPP/1.0).~~this IPP/1.1 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 8 provides a full description of all security considerations and configurations.

If a Printer object 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 object where each URI represents one of the communication channels to the Printer object. To support this flexibility, the IPP Printer object type defines a multi-valued identification attribute called the "printer-uri-supported" attribute. It MUST contain at least one URI. It MAY contain more than one URI. That is, every Printer object will have at least one URI that identifies at least one communication channel to the Printer object, but it may have more than one URI where each URI identifies a different communication channel to the Printer object. The "printer-uri-supported" attribute has ~~a~~two companion attributes, the "uri-security-supported" attribute and the "uri-~~attribute, that has~~authentication-supported". 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 ~~two~~three attributes are fully described in sections 4.4.1~~ and~~, 4.4.2~~.~~, and 4.4.3. Issue 2

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

Note: ~~IPP/1.0~~IPP/1.1 does not specify how the client obtains the client supplied URI, but it is RECOMMENDED that a Printer object be registered as an entry in a directory service. End-users and programs can then interrogate the directory searching for Printers. Section 16 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 object. The entry in the directory that represents the IPP Printer object includes the possibly many URIs for that Printer object as values in one its attributes.

When a client submits a create request to the Printer object, the Printer object validates the request and creates a new Job object. The Printer object assigns the new Job object a URI which is stored in the "job-uri" Job attribute. This URI is then used by clients as the target for subsequent Job operations. The Printer object generates a Job URI based on its configured security policy and the URI used by the client in the create request.

For example, consider a Printer object that supports both a communication channel secured by the use of SSL3 (using HTTP over SSL3 with an "https" schemed URI) and another open communication channel that is not secured with SSL3 (using a simple "http" schemed URI). If a client were to submit a job using the secure URI, the Printer object would assign the new Job object a secure URI as well. If a client were to submit a job using the open-channel URI, the Printer would assign the new Job object an open-channel URI.

In addition, the Printer object also populates the Job object's "job-printer-uri" attribute. This is a reference back to the Printer object that created the Job object. If a client only has access to a Job object's "job-uri" identifier, the client can query the Job's "job-printer-uri" attribute in order to determine which Printer object created the Job object. If the Printer object supports more than one URI, the Printer object 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.

Allowing Job objects to have URIs allows for flexibility and scalability. For example, in some implementations, the Printer object might create Jobs that are processed in the same local environment as the Printer object itself. In this case, the Job URI might just be a composition of the Printer's URI and some unique component for the Job object, such as the unique 32-bit positive integer mentioned later in this paragraph. In other implementations, the Printer object might be a central clearing-house for validating all Job object creation requests, but the Job object itself might be created in some environment that is remote from the Printer object. In this case, the Job object's URI may have no physical-location relationship at all to the Printer object's URI. Again, the fact that Job objects have URIs allows for flexibility and scalability, however, many existing printing systems have local models or interface constraints that force print jobs to be identified using only a 32-bit positive integer rather than an independent URI. This numeric Job ID is only unique within the context of the Printer object to which the create request was originally submitted. Therefore, in order to allow both types of client access to IPP Job objects (either by Job URI or by numeric Job ID), when the Printer object successfully processes a create request and creates a new Job object, the Printer object MUST generate both a Job URI and a Job ID. The Job ID (stored in the "job-id" attribute) only has meaning in the context of the Printer object to which the create request was originally submitted. This requirement to support both Job URIs and Job IDs allows all types of clients to access Printer objects and Job objects no matter the local constraints imposed on the client implementation.

In addition to identifiers, Printer objects and Job objects have names ("printer-name" and "job-name"). An object name NEED NOT be unique across all instances of all objects. A Printer object's name is chosen and set by an administrator through some mechanism outside the scope of ~~IPP/1.0.~~this IPP/1.1 document. A Job object's name is optionally chosen and supplied by the IPP client submitting the job. If the client does not supply a Job object name, the Printer object generates a name for the new Job object. In all cases, the name only has local meaning.

To summarize:

- Each Printer object is identified with one or more URIs. The Printer’s "printer-uri-supported" attribute contains the URI(s).
- The Printer object’s "uri-security-supported" attribute identifies the communication channel security protocols that may or may not have been configured for the various Printer object URIs (e.g., ~~’ssl3’~~’tls’ or ’none’).
- - The Printer object’s "uri-authentication-supported" attribute identifies the authentication mechanisms that may or may not have been configured for the various Printer object URIs (e.g., ’digest’ or ’none’).

~~-~~Each Job object is identified with a Job URI. The Job’s "job-uri" attribute contains the URI.

- Each Job object is also identified with Job ID 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 object which created the Job object.
- Each Job object has a "job-printer-uri" attribute which contains the URI of the Printer object that was used to create the Job object. This attribute is used to determine the Printer object that created a Job object when given only the URI for the Job object. This linkage is necessary to determine the languages, charsets, and operations which are supported on that Job (the basis for such support comes from the creating Printer object).
- Each Printer object has a name (which is not necessarily unique). The administrator chooses and sets this name through some mechanism outside the scope of ~~IPP/1.0 itself.~~this IPP/1.1 document. The Printer object’s "printer-name" attribute contains the name.
- Each Job object has a name (which is not necessarily unique). The client optionally supplies this name in the create request. If the client does not supply this name, the Printer object generates a name for the Job object. The Job object’s "job-name" attribute contains the name.

3. IPP Operations

IPP objects support operations. An operation consists of a request and a response. When a client communicates with an IPP object, the client issues an operation request to the URI for that object. Operation requests and responses have parameters that identify the operation. Operations also have attributes that affect the run-time 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 object attributes or Job object 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 ~~IPP/1.0~~IPP/1.1 Printer operations are:

Print-Job (section 3.2.1)
Print-URI (section 3.2.2)

Validate-Job (section 3.2.3)
Create-Job (section 3.2.4)
Get-Printer-Attributes (section 3.2.5)
Get-Jobs (section 3.2.6)
Pause-Printer (section 3.3.5)
Resume-Printer (section 3.3.6)
Purge-Jobs (section 3.3.7)

The Job operations are:

Send-Document (section 3.3.1)
Send-URI (section 3.3.2)
Cancel-Job (section 3.3.3)
Get-Job-Attributes (section 3.3.4)
Hold-Job (section 3.3.5)
Release-Job (section 3.3.6)
Restart-Job (section 3.3.7)

The Send-Document and Send-URI Job operations are used to add a new document to an existing multi-document Job object created using the Create-Job operation.

3.1 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.

3.1.1 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~~"Encoding and Transport document [IPP-PRO] 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.

3.1.2 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 4.4.15). 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 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 mappings, the operation responses can come back in any order. In these cases, the "request-id" would be essential.

3.1.3 Attributes

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

- Operation Attributes: These attributes are passed in the operation and affect the IPP object's behavior while processing the operation request and may 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 and 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 OPTIONALLY supplies Job Template Attributes in a create 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 create 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 sections 3.2.1.2 and 15 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 create request, the client supplies a set of Operation and Job Template attributes. If any of these attributes or their values is unsupported by the Printer object, the Printer object returns the set of unsupported attributes in the response. Sections 3.1.7, 3.2.1.2, and 15 give a full description of how Job Template attributes supplied by the client in a create 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 Attribute group is defined for all operation responses for returning unsupported attributes that the client supplied in the request. Issue

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 may be in any order, unless specified otherwise.

Each attribute specification includes the attribute's name followed by the name of its attribute syntax(es) in parenthesizes. In addition, each 'integer' attribute is followed by the allowed range in parentheses, (m:n), for values of that attribute. Each 'text' or 'name' attribute is 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 attributes syntaxes in section 4.1.

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 that support supplying the document data within an operation request are Print-Job and Send-Document. There are no operation responses that include document data.

Note: Some operations are REQUIRED for IPP objects to support; the others are OPTIONAL (see section 5.2.2). 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 Printer and Job 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 returns the 'server-error-operation-not-supported' status code (see section 13.1.5.2). An IPP object is non-conformant if it does not support a REQUIRED operation.

3.1.4 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 4.1). The following sections describe two special Operation Attributes called "attributes-charset" and "attributes-natural-language". These attributes are always part of the Operation Attributes group. For most attribute groups, the order of the attributes within the group is not important. However, for these two attributes within the Operation Attributes group, the order is critical. 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. In other words, these attributes MUST be supplied in every IPP request and response, they MUST come first in the group, and MUST come in the specified order. For job creation operations, the IPP Printer implementation saves these two attributes with the new Job object as Job Description attributes. For the sake of brevity in this document, these operation attribute descriptions are not repeated with every operation request and response, but have a reference back to this section instead.

3.1.4.1 Request Operation Attributes

The client MUST supply and the Printer object MUST support the following REQUIRED operation attributes in every ~~IPP/1.0~~IPP/1.1 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 4.1.1 and 4.1.2 for the specification of the 'text' and 'name' attribute syntaxes.

All clients and IPP objects MUST support the 'utf-8' charset ~~[RFC2044]~~[RFC2279] and MAY support additional charsets provided that they are registered with IANA [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 NEED NOT return any attributes in the Unsupported Attributes Group (See sections 3.1.7 and 3.2.1.2). The Printer object MUST indicate the charset(s) supported as the values of the "charset-supported" Printer attribute (see Section 4.4.18), so that the client can query to determine which charset(s) are supported.

Note to client implementers: Since IPP objects are only required to support the 'utf-8' charset, in order to maximize interoperability with multiple IPP object implementations, a client may want to 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 or ISO-8859-1. Then the client will have to filter out (or charset convert) 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 may want to use that charset in order to avoid charset conversion or data loss.

See the 'charset' attribute syntax description in Section 4.1.7 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 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.

There are no REQUIRED natural languages required for the Printer object to support. However, the Printer object’s "generated-natural-language-supported" attribute identifies the natural languages supported by the Printer object and any contained Job objects for all text strings generated by the IPP object. A client MAY query this attribute to determine which natural language(s) are 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 3.1.6), 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 4.4.19).

For other ’text’ and ’name’ attributes supplied by the client, authentication system, operator, system 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 object’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 4.1.1.2 and 4.1.2.2) 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 object’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’~~'text' or 'name' (see sections 4.1.1 and 4.1.2) 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”~~"attributes-natural-language" operation attribute in the request or response (see sections 4.1.1.1 textWithoutLanguage and 4.1.2.1 nameWithoutLanguage).

- 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 4.1.1.2 textWithLanguage and 4.1.2.2 nameWithLanguage).

For example, the "job-name" attribute MAY be supplied by the client in a create 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 object's "job-name" attribute. Whenever any client queries the Job object'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.

See the 'naturalLanguage' attribute syntax description in section 4.1.8 for the syntax and semantic interpretation of the values of this attribute and for example values.

Clients SHOULD NOT supply 'text' or 'name' attributes that use an illegal combination of natural language and charset. For example, suppose a Printer object 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 object 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 object handles this apparent incompatibility differently depending on the context in which it occurs:

- In a create 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 object or its correct operation. Therefore, the Printer object simply accepts the client supplied value, stores it with the Job object, 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 object responds (as described in section 3.1.4.2) using the Printer's configured natural language rather than the natural language requested by the client.

In either case, the Printer object 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.

3.1.4.2 Response Operation Attributes

The Printer object MUST supply and the client MUST support the following REQUIRED operation attributes in every ~~IPP/1.0~~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 3.1.4.1 above.

If the Printer object supports more than just the 'utf-8' charset, the Printer object MUST be able to code convert 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 MAY occur during the charset conversion depending on the charsets involved. For example, the Printer object may 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 '?', decimal code equivalent, or to the absence of a character, depending on implementation.

Note: Whether an implementation that supports more than one charset stores the data in the charset supplied by the client or code converts 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 NEED NOT return the same value as that supplied by the client in the request. The IPP object MAY return the natural language of the Job object or the Printer's configured natural language as identified by the Printer object'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 4.1.1.2 and 4.1.2.2) 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.

3.1.5 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 object'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 (uri)" operation attribute.

For Job operations, the operation is directed at either:

- The Job object itself using the Job object's URI. In this case, the client identifies the target object by supplying the correct URI in the "job-uri (uri)" operation attribute.
- The Printer object that created the Job object using both the Printer objects URI and the Job object's 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 object's URI in the "printer-uri (uri)" operation attribute and the Job object's Job ID in the "job-id (integer(1:MAX))" operation attribute.

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

The operation target attributes are REQUIRED operation attributes that MUST be included in every operation request. Like the charset and natural language attributes (see section 3.1.4), 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 attribute group, however the specific ordering rules are:

- 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 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: The IPP ~~encoding and transport~~"Encoding and Transport document [IPP-PRO] shows a mapping of IPP onto HTTP/1.1 and defines a new default port number for using IPP over HTTP/1.1.

3.1.6 Operation Status Codes and Messages

Every operation response includes a REQUIRED "status-code" parameter and an OPTIONAL "status-message" operation attribute. The "status-code" provides information on the processing of a request. A "status-message" attribute provides a short textual description of the status of the operation. The status code is intended for use by automata, and the status message is intended for the human end user. The "status-message" is especially useful for a later version of a Printer object 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 a response does include a "status-message" attribute, an IPP client NEED NOT examine or display the message, however it SHOULD do so in some implementation specific manner.

The "status-code" value is a numeric value that has semantic meaning. The "status-code" syntax is similar to a "type2 enum" (see section 4.1 on "Attribute Syntaxes") except that values can range only from 0x0000 to 0x7FFF. Section 13 describes the status codes, assigns the numeric values, and suggests a corresponding status message for each status code. The "status-message" attribute's syntax is "text(255)". A client implementation of IPP SHOULD convert status code values into any localized message that has semantic meaning to the end user.

If the Printer object supports the "status-message" operation attribute, the Printer object MUST be able to generate this message in any of the natural languages identified by the Printer object's "generated-natural-language-supported" attribute (see the "attributes-natural-language" operation attribute specified in section 3.1.4.1). As described in section 3.1.4.1 for any returned 'text' attribute, if there is a choice for generating this message, the Printer object uses the natural language indicated by the value of the "attributes-natural-language" in the client request if supported, otherwise the Printer object uses the value in the Printer object's own "natural-language-configured" attribute. If the Printer object supports the "status-message" operation attribute, it SHOULD use the REQUIRED 'utf-8' charset to return a status message for the following error status codes (see section 13): '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.

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 section 13.

If the client supplies unsupported values for the following parameters or Operation attributes, the Printer object MUST reject the operation, NEED NOT 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

If the client supplies unsupported values for other attributes, or unsupported attributes, the Printer returns the status code defined in the next section on Unsupported Attributes.

3.1.7 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 object 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 “unsupported” attributes. Issue 18, Issue 23, and Issue 27

If the Printer object is not returning any Unsupported Attributes in the response, the Printer object SHOULD omit Group 2 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 object does not support the supplied attribute (no matter what the attribute syntax or value).
2. The Printer object does support the attribute, but does not support some or all of the particular attribute syntaxes or values supplied by the client (i.e., the Printer object does not have those attribute syntaxes or values in its corresponding "xxx-supported" attribute).
3. The Printer object 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 object 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" specification [IPP-PRO]. Its value indicates no support for the attribute itself (see the beginning of section 4.1). Issue 12

In the case of a supported attribute with one or more unsupported attribute syntaxes or values, the Printer object 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 object supports the attribute but only supports a subset of the client-supplied attribute syntaxes or values, the Printer object 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 object 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 sections 3.2.1.2 and 15. See The Implementer’s Guide [IPP-IIG] for an example.

3.1.8 Versions

Each operation request and response carries with it a "version-number" parameter. Each value of the "version-number" 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. If the IPP object does not support that version, the object responds with a status code of ’server-error-version-not-supported’ along with the closest version number that is supported (see section 13.1.5.4).

There is no version negotiation per se. However, if after receiving a ’server-error-version-not-supported’ status code from an IPP object, there is nothing that prevents a client from trying again with a different version number. In order to conform to ~~IPP/1.0, an implementation~~IPP/1.1, an IPP object implementations MUST support version ’1.1’ SHOULD ~~at least~~support version ’1.0’. Issue 33

There is only one notion of "version number" that covers both IPP Model and IPP Protocol changes. Thus the version number MUST change when introducing a new version of the Model and Semantics document [IPP-MOD] or a new version of the ~~Encoding and Transport~~"Encoding and Transport" document [IPP-PRO].

Changes to the major version number indicate structural or syntactic changes that make it impossible for older version of IPP clients and Printer objects to correctly parse and correctly process the new or changed attributes, operations and responses. If the major version number changes, the minor version numbers 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’),~~’1.1’), 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 ~~number.~~not know about. Items that might affect the changing of the major version number include any changes to the Model and Semantics document [IPP-MOD] or the~~Encoding and Transport [IPP-~~ "Encoding and Transport" document [IPP-PRO] itself, such as:

- reordering of ordered attributes or attribute sets
- changes to the syntax of existing attributes
- ~~changing Operation or Job Template attributes from OPTIONAL to REQUIRED and vice versa~~
- ~~adding REQUIRED (for an IPP object to support) operation attributes~~

- adding REQUIRED (for an IPP object to support) operation attribute 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 may not be understood by all IPP objects, but which 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 [IPP-PRO] (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 attribute groups (i.e., those attributes that an IPP object can ignore without confusing clients)
- adding new attribute syntaxes
- adding OPTIONAL operations
- changing Job Description attributes or Printer Description attributes from OPTIONAL to REQUIRED or vice versa.
- adding OPTIONAL attribute syntaxes to an existing attribute. Issue 33

The encoding of the ~~"operation-id",~~ the "version-number"~~, the "status-code", and the "request-id"~~ 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~~"operation-id", the~~ "version-number"~~, and the "request-id"~~). 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~~major~~ version NEED NOT support ALL previous versions. As each new~~major~~ version is defined (through the release of a new specification), that major version will specify which previous~~major~~ versions MUST and which versions SHOULD be supported in compliant implementations. Issue 33

3.1.9 Job Creation Operations

In order to "submit a print job" and create a new Job object, a client issues a create request. A create request is any one of following three operation requests:

- The Print-Job Request: A client that wants to submit a print job with only a single document uses the Print-Job operation. The operation allows for the client to "push" the document data to the Printer object by including the document data in the request itself.

- The Print-URI Request: A client that wants to submit a print job with only a single document (where the Printer object "pulls" the document data instead of the client "pushing" the data to the

Printer object) 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 multiple 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 create Job object). 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 object includes a "last-document" operation attribute set to 'true' indicating that this is the last request.

Throughout this model specification, the term "create 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, for performance reasons, the client SHOULD use the Print-Job operation for all single document jobs. Also, Print-Job is a REQUIRED operation (all implementations MUST support it) whereas Create-Job is an OPTIONAL operation, hence some implementations might not support it.

Job submission time is the point in time when a client issues a create request. The initial state of every Job object is the 'pending', ~~or~~ 'pending-held', or 'processing' state (see section 4.3.7). Issue 13 When the Printer object begins processing the print job~~. At this point in time~~, the Job object's state moves to 'processing'. This is known as job processing time. There are validation checks that must be done at job submission time and others that must be performed at 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 object MUST validate whether or not the supplied attributes, attribute syntaxes, and values are supported by matching them with the Printer object's corresponding "xxx-supported" attributes. See section 3.1.7 for details. [IPP-IIG] presents suggested steps for an IPP object to either accept or reject any request and additional steps for processing create requests.

At job submission time the Printer object NEED NOT perform the validation checks reserved for job processing time such as:

1. Validating 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 object 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 object, especially a non-spooling Printer, MAY accept jobs that it does not have enough space for. In such a situation, a Printer object 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 5.1 on client conformance).

When a Printer object has too little space for starting a new job, it MAY reject a new create request. In this case, a Printer object MUST return a response (in reply to the rejected request) with a status-code of 'server-error-busy' (See section 14.1.5.8) and it MAY close the connection before receiving all bytes of the operation. When receiving a 'server-error-busy' status-code in an operation response, a client MUST be prepared for the Printer object 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 create request until the IPP Printer object accepts the create request. Issue 20

At job processing time, since the Printer object has already responded with a successful status code in the response to the create request, if the Printer object detects an error, the Printer object 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 object's "job-state", "job-state-reasons", 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 outside the scope of ~~IPP/1.0.~~this IPP/1.1 document.

3.2 Printer Operations

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

3.2.1 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 Section 15 for the suggested steps for processing create operations and their Operation and Job Template attributes.

3.2.1.1 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 3.1.4.1. The Printer object MUST copy these values to the corresponding Job Description attributes described in sections 0 and 4.3.18.

Target:
The "printer-uri" (uri) operation attribute which is the target for this operation as described in section 3.1.5.

Requesting User Name:
The "requesting-user-name" (name(MAX)) attribute SHOULD be supplied by the client as described in section 8.3.

"job-name" (name(MAX)):
The client OPTIONALLY supplies this attribute. The Printer object 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 object. 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 object (see Section 4.3.5).

"ipp-attribute-fidelity" (boolean):
The client OPTIONALLY supplies this attribute. The Printer object MUST support this attribute. The value 'true' indicates that total fidelity to client supplied Job Template attributes and values is required, else the Printer object MUST reject the Print-Job request. The value 'false' indicates that a reasonable attempt to print the Job object is acceptable and the Printer object MUST accept the Print-job request. If not supplied, the Printer object assumes the value is 'false'. All Printer objects MUST support both types of job processing. See section 15 for a full description of "ipp-attribute-fidelity" and its relationship to other attributes, especially the Printer object's "pdl-override-supported" attribute.

"document-name" (name(MAX)):
The client OPTIONALLY supplies this attribute. The Printer object 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: 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. ~~IPP/1.0~~IPP/1.1 does not support the concept of document level attributes.

"compression" (type3 keyword)

The client OPTIONALLY supplies this attribute. The Printer object ~~OPTIONALLY supports~~MUST support this attribute and the "compression-supported" attribute (see section 4.4.32). 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 object 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 the attribute and the Printer object supports the attribute, the Printer object uses the corresponding decompression algorithm on the documentdata.~~ If the client supplies this attribute, but the value is not supported by the Printer object, i.e., the value is not one of the values of the Printer object's "compression-supported" attribute, the Printer object MUST~~copy the attribute and its value to the Unsupported Attributes response group,~~ reject the request, and return the ~~'client-error-attributes-or-values-not-supported' status code.~~'client-error-compression-not-supported' status code. See section 3.1.7 for returning unsupported attributes and values.

c) If the client supplies the attribute and the Printer object supports the attribute value, the Printer object uses the corresponding decompression algorithm on the document data.

d) If the decompression algorithm fails before the Printer returns an operation response, the Printer object 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 object 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 (q.v.). Issue 28

"document-format" (mimeMediaType) :

The client OPTIONALLY supplies this attribute. The Printer object MUST support this attribute. The value of this attribute identifies the format of the supplied document data. The following cases exist:

a) If the client does not supply this attribute, the Printer object assumes that the document data is in the format defined by the Printer object'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 object, i.e., the value is not one of the values of the Printer object's "document-format-supported" attribute, the Printer object 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 4.1.9.1), 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 object, the document data, the Printer is capable of interpreting the document data.

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

f) If interpreting of the document data fails after the Printer returns an operation response, the Printer object MUST abort the job and add the 'document-format-error' value to the job's "job-state-reasons" attribute. Issue 11

"document-natural-language" (naturalLanguage):

The client OPTIONALLY supplies this attribute. The Printer object OPTIONALLY supports this attribute. This attribute specifies the natural language of the document for those document-formats that require a specification of the natural language in order to image the document unambiguously. There are no particular values required for the Printer object to support.

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

The client OPTIONALLY supplies this attribute. The Printer object OPTIONALLY supports this attribute and the "job-k-octets-supported" attribute (see section 4.4.33). The client supplied "job-k-octets" operation attribute identifies the total size of the document(s) in K octets being submitted (see section 4.3.15.1 for the complete semantics). If the client supplies the attribute and the Printer object supports the attribute, the value of the attribute is used to populate the Job object's "job-k-octets" Job Description attribute.

Note: For this attribute and the following two attributes ("job-impressions", and "job-media-sheets"), if the client supplies the attribute, but the Printer object does not support the attribute, the Printer object 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 object's "xxx-supported" attribute, the Printer object MUST use the value to populate the Job object'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 object's "xxx-supported" attribute, the Printer object MUST copy the attribute and its value to the Unsupported Attributes response 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 object MAY choose to populate the corresponding Job object attribute depending on whether the Printer object supports the attribute and is able to calculate or discern the correct value.

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

The client OPTIONALLY supplies this attribute. The Printer object OPTIONALLY supports this attribute and the "job-impressions-supported" attribute (see section 4.4.34). The client supplied "job-impressions" operation attribute identifies the total size in number of impressions of the document(s) being submitted (see section 4.3.15.2 for the complete semantics).

See note under "job-k-octets".

"job-media-sheets" (integer(0:MAX))

The client OPTIONALLY supplies this attribute. The Printer object OPTIONALLY supports this attribute and the "job-media-sheets-supported" attribute (see section 4.4.35). The client supplied "job-media-sheets" operation attribute identifies the total number of media sheets to be produced for this job (see section 4.3.15.3 for the complete semantics).

See note under "job-k-octets".

Group 2: Job Template Attributes

The client OPTIONALLY supplies a set of Job Template attributes as defined in section 4.2. 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 object MUST be able to accept an empty group.

Group 3: Document Content

The client MUST supply the document data to be processed.

Note: In addition to the MANDATORY parameters required for every operation request, the simplest Print-Job Request consists of just the "attributes-charset" and "attributes-natural-language" operation attributes; the "printer-uri" target operation attribute; the Document Content and nothing else. In this simple case, the Printer object:

- creates a new Job object (the Job object contains a single document),
- stores a generated Job name in the "job-name" attribute in the natural language and charset requested (see Section 3.1.4.1) (if those are supported, otherwise using the Printer object'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 IPP attribute or embedded instructions in the document data.

3.2.1.2 Print-Job Response

The Printer object MUST return to the client the following sets of attributes as part of the Print-Job Response:

Group 1: Operation Attributes

Status Message:

In addition to the REQUIRED status code returned in every response, the response OPTIONALLY includes a "status-message" (text) operation attribute as described in sections 13 and 3.1.6. If the client supplies unsupported or conflicting Job Template attributes or values, the Printer object MUST reject or accept the Print-Job request depending on the whether the client supplied a 'true' or 'false' value for the "ipp-attribute-fidelity" operation attribute. See the Implementer's Guide [IPP-IIG] for a complete description of the suggested steps for processing a create request.

Natural Language and Character Set:

The "attributes-charset" and "attributes-natural-language" attributes as described in section 3.1.4.2.

Group 2: Unsupported Attributes

~~This is a set of Operation and Job Template attributes supplied by the client (in the request) that are not supported by the Printer object or that conflict with one another (see the Implementer's Guide [IPP-IIG]). If the Printer object is not returning any Unsupported Attributes in the response, the Printer object SHOULD omit Group 2 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 object does not support the supplied attribute (no matter what the attribute syntax or value).~~

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

~~3. The Printer object 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 object returns the client-supplied attribute with a substituted "out-of-band" value of 'unsupported' indicating no support for the attribute itself (see the beginning of section 4.1).~~

~~In the case of a supported attribute with one or more unsupported attribute syntaxes or values, the Printer object 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 object supports the attribute but only supports a subset of the client-supplied attribute syntaxes or values, the Printer object 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 object 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 The Implementer's Guide [IPP-IIG] for an example.~~

See section 3.1.7 for details on returning Unsupported Attributes.

~~In these three cases, the~~The value of the "ipp-attribute-fidelity" supplied by the client does not affect what attributes the Printer object 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 may query the job using the Get-Job-Attributes operation requesting the unsupported attributes that were returned in the create response to see which attributes were ignored (not stored on the Job object) and which attributes were stored with other (substituted) values.

Group 3: Job Object Attributes

"job-uri" (uri):

The Printer object MUST return the Job object's URI by returning the contents of the REQUIRED "job-uri" Job object attribute. The client uses the Job object's URI when directing operations at the Job object. The Printer object always uses its configured security policy when creating the new URI. However, if the Printer object supports more than one URI, the Printer object also uses information about which URI was used in the Print-Job Request to generated the new URI so that the new URI references the correct access channel. In other words, if the Print-Job Request comes in over a secure channel, the Printer object MUST generate a Job URI that uses the secure channel as well.

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

The Printer object MUST return the Job object's Job ID by returning the REQUIRED "job-id" Job object 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 object.

"job-state":

The Printer object MUST return the Job object's REQUIRED "job-state" attribute. The value of this attribute (along with the value of the next attribute: "job-state-reasons") is taken from a "snapshot" of the new Job object at some meaningful point in time (implementation defined) between when the Printer object receives the Print-Job Request and when the Printer object returns the response.

"job-state-reasons":

The Printer object ~~OPTIONALLY returns~~MUST return the Job object's ~~OPTIONAL "job-state-reasons" attribute. If the Printer object 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-reasons" attribute is not supported and will not~~ REQUIRED "job-state-reasons" attribute. .
Issue 30
~~be returned in a subsequent Job object query.~~

"job-state-message":

The Printer object OPTIONALLY returns the Job object's OPTIONAL "job-state-message" attribute. If the Printer object 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 object query.

"number-of-intervening-jobs":

The Printer object OPTIONALLY returns the Job object's OPTIONAL "number-of-intervening-jobs" attribute. If the Printer object 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 object query.

Note: Since any printer state information which 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 specific printer status attributes.

Note: In addition to the MANDATORY parameters required for every operation response, the simplest response consists of the just the "attributes-charset" and "attributes-natural-language" operation attributes and the "job-uri", "job-id", and "job-state" Job Object Attributes. In this simplest case, the status code is ~~"successful-ok"~~'successful-ok' and there is no "status-message" operation attribute.

3.2.2 Print-URI Operation

This OPTIONAL operation is identical to the Print-Job operation (section 3.2.1) 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 object's "referenced-uri-scheme-supported" attribute, the Printer object MUST reject the request and return the 'client-error-uri-scheme-not-supported' status code.

The IPP Printer MAY validate the accessibility of the document as part of the operation or subsequently. If the Printer determines an accessibility problem before returning an operation response, it rejects the request and returns the 'client-error-document-access-error' status code. If the Printer determines this accessibility problem after accepting the request and returning an operation response with one of the successful status codes, the Printer adds the 'document-access-error' value to the job's "job-state-reasons" attribute. See The Implementer's Guide [IPP-IIG] for suggested additional checks. ~~The Printer NEED NOT follow the reference and validate the contents of the reference.~~Issue 35

If the Printer object supports this operation, it MUST support the "reference-uri-schemes-supported" Printer attribute (see section 4.4.27).

It is up to the IPP object to interpret the URI and subsequently "pull" the document from the source referenced by the URI string.

3.2.3 Validate-Job Operation

This REQUIRED operation is similar to the Print-Job operation (section 3.2.1) except that a client supplies no document data and the Printer allocates no resources (i.e., it does not create a new Job object). This operation is used only to verify capabilities of a printer object 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 Print-Job operation (with the document data) would be accepted. The Validate-Job operation also performs the same security negotiation as the Print-Job operation (see section 8), so that a client can check that the client and Printer object security requirements can be met before performing a Print-Job operation.

Note: 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 object returns the same status codes, Operation Attributes (Group 1) and Unsupported Attributes (Group 2) as the Print-Job operation. However, no Job Object Attributes (Group 3) are returned, since no Job object is created.

3.2.4 Create-Job Operation

This OPTIONAL operation is similar to the Print-Job operation (section 3.2.1) 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 OPTIONALLY supplies the "document-name", "document-format", and "document-natural-language" attributes for each document in the multi-document Job object.

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

If the Printer object supports this operation, it MUST support the "multiple-operation-time-out" Printer attribute (see section 4.4.31).

~~In addition to the Print-Job status codes in the following additional error status codes not applicable to Print-Job MAY be returned:~~

If the Printer object supports this operation, then it MUST support the "multiple-document-jobs-supported" Printer Description attribute and indicate whether or not it supports multiple-document jobs. Issue 34

If the Printer object supports this operation and supports multiple documents in a job, then it MUST support the "multiple-document-handling" Job Template job attribute with at least one value (see section 4.2.4) and the associated "multiple-document-handling-default" and "multiple-document-handling-supported" Job Template Printer attributes. Issue 34

After the Create-Job operation has completed, the value of the "job-state" attribute is similar to the "job-state" after a Print-Job, even though no document-data has arrived. A Printer MAY set the 'job-data-insufficient' value of the job's "job-state-reason" 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 may even set the "job-state" to 'processing', even though there is not yet any data to process. Issue 13

3.2.5 Get-Printer-Attributes Operation

This REQUIRED operation allows a client to request the values of the attributes of a Printer object. 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 object returns a corresponding attribute set with the appropriate attribute values filled in.

For Printer objects, the possible names of attribute groups are:

- 'job-template': ~~all~~the subset of the Job Template attributes that apply to a Printer object (the last two columns of the table in Section 4.2) that the implementation supports for Printer objects.
- 'printer-description': the subset of the attributes specified in Section 4.4 that the implementation supports for Printer objects.
- 'all': the special group 'all' that includes all ~~supported attributes.~~attributes that the implementation supports for Printer objects. Issue 23

Since a client MAY request specific attributes or named groups, there is a potential that there is 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 object NEED NOT return 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.

It is NOT REQUIRED that a Printer object support all attributes belonging to a group (since some attributes are OPTIONAL). However, it is REQUIRED that each Printer object support all group names.

3.2.5.1 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 3.1.4.1.

Target:

The "printer-uri" (uri) operation attribute which is the target for this operation as described in section 3.1.5.

Requesting User Name:

The "requesting-user-name" (name(MAX)) attribute SHOULD be supplied by the client as described in section 8.3.

"requested-attributes" (1setOf keyword) :

The client OPTIONALLY supplies a set of attribute names and/or attribute group names in whose values the requester is interested. The Printer object 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 OPTIONALLY supplies this attribute. The Printer object MUST support this attribute. This attribute is useful for a Printer object to determine the set of supported attribute values that relate to the requested document format. The Printer object MUST return the attributes and values that it uses to validate a job on a create or Validate-Job operation in which this document format is supplied. The Printer object 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 object to eliminate the attributes and values that are not supported for a specific document format. For example, a Printer object might have multiple interpreters to support both 'application/postscript' (for PostScript) and 'text/plain' (for text) documents. However, for only one of those interpreters might the Printer object be able to support "number-up" with values of '1', '2', and '4'. For the other interpreter it might be able to only support "number-up" 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 create job operation.

If the Printer object does not distinguish between different sets of supported values for each different document format when validating jobs in the create and Validate-Job operations, it MUST NOT distinguish between different document formats in the Get-Printer-Attributes operation. If the Printer object 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 object attributes:

- Printer attributes that are Job Template attributes ("xxx-default" "xxx-supported", and "xxx-ready" in the Table in Section 4.2),
- "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 object attributes (including "document-format-supported") remain invariant with respect to the client supplied document format (except for new Printer description attribute as registered according to section 6.2).

If the client omits this "document-format" operation attribute, the Printer object MUST respond as if the attribute had been supplied with the value of the Printer object's "document-format-default" attribute. It is recommended that the client always supply a value for "document-format", since the Printer object's "document-format-default" may 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 4.1.9.

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 object's "document-format-supported" attribute, the Printer object MUST reject the operation and return the 'client-error-document-format-not-supported' status code.

3.2.5.2 Get-Printer-Attributes Response

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

Group 1: Operation Attributes

Status Message:

In addition to the REQUIRED status code returned in every response, the response OPTIONALLY includes a "status-message" (text) operation attribute as described in sections 13 and 3.1.6.

Natural Language and Character Set:

The "attributes-charset" and "attributes-natural-language" attributes as described in section 3.1.4.2.

Group 2: Unsupported Attributes

See section 3.1.7 for details on returning Unsupported Attributes.

~~This is a set of Operation attributes supplied by the client (in the request) that are not supported by the Printer object or that conflict with one another (see sections 3.2.1.2 and 16).~~ The response

NEED NOT contain the "requested-attributes" operation attribute with any supplied values (attribute keywords) that were requested by the client but are not supported by the IPP object. If the Printer ~~object is not returning any Unsupported Attributes in the response, the Printer object SHOULD omit Group 2 rather than sending an empty group. However, a client MUST be able to accept an empty group.~~ does include unsupported attributes referenced in "requested-attributes" and such attributes include group names, such as 'all', the unsupported attributes MUST NOT include attributes described in the standard but not supported by the implementation. Issue 23

Group 3: Printer Object Attributes

This is the set of requested attributes and their current values. The Printer object ignores (does not respond with) any requested attribute which is not supported. The Printer object MAY respond with a subset of the supported attributes and values, depending on the security policy in force. However, the Printer object MUST respond with the 'unknown' value for any supported attribute (including all REQUIRED attributes) for which the Printer object does not know the value. Also the Printer object MUST respond with the 'no-value' for any supported attribute (including all REQUIRED attributes) for which the system administrator has not configured a value. See the description of the "out-of-band" values in the beginning of Section 4.1.

3.2.6 Get-Jobs Operation

This REQUIRED operation allows a client to retrieve the list of Job objects belonging to the target Printer object. The client may also supply a list of Job attribute names and/or attribute group names. A group of Job object attributes will be returned for each Job object 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 object (see the description of Job attribute group names in section 3.3.4).

3.2.6.1 Get-Jobs Request

The client submits the Get-Jobs request to a Printer object.

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 3.1.4.1.

Target:

The "printer-uri" (uri) operation attribute which is the target for this operation as described in section 3.1.5.

Requesting User Name:

The "requesting-user-name" (name(MAX)) attribute SHOULD be supplied by the client as described in section 8.3.

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

The client OPTIONALLY supplies this attribute. The Printer object MUST support this attribute. It is an integer value that ~~indicates a limit to the number of Job objects~~determines the maximum number of jobs that a client will receive from the Printer even if "which-jobs" or "my-jobs" 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. There is no mechanism to allow for the next 'M' jobs after the first 'N' jobs. If the client does not supply this attribute, the Printer object responds with all applicable jobs. Issue 8

"requested-attributes" (1setOf keyword):

The client OPTIONALLY supplies this attribute. The Printer object MUST support this attribute. It is a set of Job attribute names and/or attribute groups names in whose values the requester is interested. This set of attributes is returned for each Job object that is returned. The allowed attribute group names are the same as those defined in the Get-Job-Attributes operation in section 3.3.4. 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" (keyword):

The client OPTIONALLY supplies this attribute. The Printer object MUST support this attribute. It indicates which Job objects MUST be returned by the Printer object. The values for this attribute are:

'completed': This includes any Job object whose state is 'completed', 'canceled', or 'aborted'.
'not-completed': This includes any Job object whose state is 'pending', 'processing', 'processing-stopped', or 'pending-held'.

A Printer object 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, the Printer object MUST copy the attribute and the unsupported value to the Unsupported Attributes response 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 object MUST respond as if the client had supplied the attribute with a value of 'not-completed'.

"my-jobs" (boolean):

The client OPTIONALLY supplies this attribute. The Printer object MUST support this attribute. It indicates whether~~all~~ jobs from all users or just the jobs submitted by the requesting user of this request MUST be returned by the Printer object. If the client does not supply this attribute, the Printer object MUST respond as if the client had supplied the attribute with a value of 'false', i.e., ~~all jobs.~~jobs from all users. The means for authenticating the requesting user and matching the jobs is described in section 8.

3.2.6.2 Get-Jobs Response

The Printer object returns all of the Job objects 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 Job objects are returned since there may literally be no Job objects at the Printer, or there may be no Job objects that match the criteria supplied by the client. If the client requests any Job attributes at all, there is a set of Job Object Attributes returned for each Job object.

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 1 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'. Issue 24

Group 1: Operation Attributes

Status Message:
In addition to the REQUIRED status code returned in every response, the response OPTIONALLY includes a "status-message" (text) operation attribute as described in sections 13 and 3.1.6.

Natural Language and Character Set:
The "attributes-charset" and "attributes-natural-language" attributes as described in section 3.1.4.2.

Group 2: Unsupported Attributes

See section 3.1.7 for details on returning Unsupported Attributes.

~~This is a set of Operation attributes supplied by the client (in the request) that are not supported by the Printer object or that conflict with one another (see sections 3.2.1.2 and the Implementer's Guide [IPP-IIG]).~~ The response NEED NOT contain the "requested-attributes" operation attribute with any supplied values (attribute keywords) that were requested by the client but are not supported by the IPP object. If the Printer ~~object is not returning any Unsupported Attributes in the response, the Printer object SHOULD omit Group 2 rather than sending an empty group. However, a client MUST be able to accept an empty group.~~ does include unsupported attributes referenced in "requested-attributes" and such attributes include group names, such as 'all', the unsupported attributes MUST NOT include attributes described in the standard but not supported by the implementation. Issue 23

Groups 3 to N: Job Object Attributes

The Printer object responds with one set of Job Object Attributes for each returned Job object. The Printer object ignores (does not respond with) any requested attribute or value which is not supported or which 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 8). However, the Printer object MUST respond with the 'unknown' value for any supported attribute (including all REQUIRED attributes) for which the Printer object 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 4.1.

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 object).

3.2.7 Pause-Printer Operation

This OPTIONAL operation allows a client to stop the Printer object 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 stopped as soon as the implementation permits or is completed, depending on implementation. The Printer object MUST still accept create operations 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(s) that were in the 'processing' or 'processing-stopped' states as soon as the implementation permits. If the implementation Issue 30 will take appreciable time to stop, the IPP Printer adds the 'moving-to-paused' value to the Printer object's "printer-state-reasons" attribute (see section 4.4.12). When the device(s) have all stopped, the IPP Printer transitions the Printer object to the 'stopped' state, removes the 'moving-to-paused' value, if present, and adds the 'paused' value to the Printer object's "printer-state-reasons" attribute.

When the current job(s) complete that were in the 'processing' state, the IPP Printer transitions them to the 'completed' state. When the current job(s) stop in mid processing that were in the 'processing' state, the IPP Printer transitions them to the 'processing-stopped' state and Issue 30 adds the 'printer-stopped' value to the job's "job-state-reasons" attribute.

Note: 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 NEED NOT update those jobs' "job-state-reasons" attributes and only need return the 'printer-stopped' value when those jobs are queried (so-called "lazy evaluation").

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

The IPP Printer MUST accept the request in any state and transition the Printer to the indicated new "printer-state" before returning as follows:

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'

Access Rights: The authenticated user (see section 8.3) performing this operation must be an operator or administrator of the Printer object (see Sections 1 and 8.5). Otherwise, the IPP Printer MUST reject the operation and return: 'client-error-forbidden', 'client-error-not-authenticated', or 'client-error-not-authorized' as appropriate.

3.2.7.1 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 3.1.4.1.

Target:
The "printer-uri" (uri) operation attribute which is the target for this operation as described in section 3.1.5.

Requesting User Name:
The "requesting-user-name" (name(MAX)) attribute SHOULD be supplied by the client as described in section 8.3.

3.2.7.2 Pause-Printer Response

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

Group 1: Operation Attributes

Status Message:

In addition to the REQUIRED status code returned in every response, the response OPTIONALLY includes a "status-message" (text) operation attribute as described in sections 13 and 3.1.6.

Natural Language and Character Set:

The "attributes-charset" and "attributes-natural-language" attributes as described in section 3.1.4.2.

Group 2: Unsupported Attributes

See section 3.1.7 for details on returning Unsupported Attributes.

3.2.8 Resume-Printer Operation

This operation allows a client to resume the Printer object scheduling jobs on all its devices. The Printer object Issue 30 MUST remove the 'paused' and 'moving-to-paused' values from the Printer object's "printer-state-reasons" attribute, if present. If there are no other reasons to keep a device paused (such as media-jam), the IPP Printer transitions itself to the 'processing' or 'idle' states, depending on whether there are jobs to be processed or not, respectively, and the device(s) resume 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, transition the Printer object to the indicated new state as follows:

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.

Access Rights: The authenticated user (see section 8.3) performing this operation must be an operator or administrator of the Printer object (see Sections 1 and 8.5). 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 3.2.7.1 and 3.2.7.2).

3.2.9 Purge-Jobs Operation

This OPTIONAL operation allows a client to remove all jobs from an IPP Printer object, regardless of their job states, including jobs in the Printer object's Job History (see Section 4.3.7.1). After a Purge-Jobs operation has been performed, a Printer object MUST return no jobs in subsequent Get-Job-Attributes and Get-Jobs responses (until new jobs are submitted).

Whether the Purge-Jobs (and Get-Jobs) operation affects jobs that were submitted to the device from other sources than the IPP Printer object in the same way that the Purge-Jobs operation affects jobs that were submitted to the IPP Printer object using IPP, depends on implementation, i.e., on whether the IPP protocol 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-Job operation.

The Printer object MUST accept this operation in any state and transition the Printer object to the 'idle' state.

Access Rights: The authenticated user (see section 8.3) performing this operation must be an operator or administrator of the Printer object (see Sections 1 and 8.5). Otherwise, the IPP object 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 3.2.7.1 and 3.2.7.2).

3.3 Job Operations

All Job operations are directed at Job objects. 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 MAY either be a single Job URI or a combination of a Printer URI with a Job ID. The IPP object implementation MUST support both forms of identification for every job.

3.3.1 Send-Document Operation

This OPTIONAL operation allows a client to create a multi-document Job object that is initially "empty" (contains no documents). In the Create-Job response, the Printer object returns the Job object's URI (the "job-uri" attribute) and the Job object'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'. 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 jobs with a 'false' value for "last-document" in the first Send-Document and a 'true' for "last-document" in the second Send-Document (with no data). Issue 34

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 an IPP Printer defined minimum time interval after the receipt of the previous request for the job. If a Printer object supports multiple document jobs, the Printer object MUST support the "multiple-operation-time-out" attribute (see section 4.4.31). This attribute indicates the minimum number of seconds the Printer object will wait for the next send operation before taking some recovery action.

An IPP object MUST recover from an errant client that does not supply a send operation, sometime after the minimum time interval specified by the Printer object's "multiple-operation-time-out" attribute. Such recovery MAY include any of the following 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 4.3.8), ~~if supported,~~Issue 30 and clean up all resources associated with the Job. In this case, if another send operation is finally received, the Printer responds with an "client-error-not-possible" or "client-error-not-found" depending on whether or not the Job object 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 object, 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, close the Job, but move it to the 'pending-held' and add the 'submission-interrupted' value to the job's "job-state-reasons" attribute (see section ~~4.3.8), if supported.~~ 4.3.8). Issue 30 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 3.3.6) or to cancel the job using the Cancel-Job operation (see section 3.3.3).

Each implementation is free to decide the "best" action to take depending on 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 object, it is possible that the Job object may already have been processed to the point that some media sheet pages have been printed.

Access Rights: The authenticated user (see section 8.3) performing this operation must either be the job owner (as determined in the Create-Job operation) or an operator or administrator of the Printer object (see Sections 1 and 8.5). Otherwise, the IPP object MUST reject the operation and return: 'client-error-forbidden', 'client-error-not-authenticated', or 'client-error-not-authorized' as appropriate. Issue 19

3.3.1.1 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 3.1.4.1.

Target:
Either (1) the "printer-uri" (uri) plus "job-id" (integer(1:MAX))or (2) the "job-uri" (uri) operation attribute(s) which define the target for this operation as described in section 3.1.5.

Requesting User Name:
The "requesting-user-name" (name(MAX)) attribute SHOULD be supplied by the client as described in section 8.3.

"document-name" (name(MAX)):
The client OPTIONALLY supplies this attribute. The Printer object MUST support this attribute. It contains the client supplied document name. The document name MAY be different than the Job name. It might be helpful, but NEED NOT 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 3.2.1.1) for more information about this attribute.

"compression" (type3 keyword)
See the description of "compression" for the Print-Job operation in Section 3.2.1.1.

"document-format" (mimeMediaType) :
~~The client OPTIONALLY supplies this attribute. The Printer object MUST support this attribute. The value of this attribute identifies the format of the supplied document data. If the client does not supply this attribute, the Printer object assumes that the document data is in the format defined by the Printer object's "document-format-default" attribute. If the client supplies this attribute, but the value is not supported by the Printer object, i.e., the value is not one of the values of the Printer object's "document-format-supported" attribute, the Printer object MUST reject the request and return the 'client-error-document-format-not-supported' status code.~~See the description of "document-format" for the Print-Job operation in Section 3.2.1.1. Issue 11

"document-natural-language" (naturalLanguage):
The client OPTIONALLY supplies this attribute. The Printer object OPTIONALLY supports this attribute. This attribute specifies the natural language of the document for those document-formats that require a specification of the natural language in order to image the document unambiguously. There are no particular values required for the Printer object to support.

~~"compression" (type3 keyword)~~

~~The client OPTIONALLY supplies this attribute. The Printer object OPTIONALLY supports this attribute and the "compression-supported" attribute (see section 4.4.29). The client supplied "compression" operation attribute identifies the compression algorithm used on the document data. If the client omits this attribute, the Printer object MUST assume that the data is not compressed. If the client supplies the attribute and the Printer object supports the attribute, the Printer object MUST use the corresponding decompression algorithm on the document data. If the client supplies this attribute, but the value is not supported by the Printer object, i.e., the value is not one of the values of the Printer object's "compression-supported" attribute, the Printer object MUST copy the attribute and its value to the Unsupported Attributes response group, reject the request, and return the 'client-error-attributes-or-values-not-supported' status code.~~

"last-document" (boolean):

The client MUST supply this attribute. The Printer object MUST support this attribute. It is a boolean flag that is set to 'true' if this is the last document for the Job, 'false' otherwise.

Group 2: Document Content

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 object's "number-of-documents" attribute, since no real document was added to the job.

3.3.1.2 Send-Document Response

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

Group 1: Operation Attributes

Status Message:

In addition to the REQUIRED status code returned in every response, the response OPTIONALLY includes a "status-message" (text) operation attribute as described in sections 13 and 3.1.6.

Natural Language and Character Set:

The "attributes-charset" and "attributes-natural-language" attributes as described in section 3.1.4.2.

Group 2: Unsupported Attributes

~~This is a set of Operation attributes supplied by the client (in the request) that are not supported by the Printer object or that conflict with one another (see sections 3.2.1.2 and the Implementer's Guide [IPP-IIG]). If the Printer object is not returning any Unsupported Attributes in the~~

~~response, the Printer object SHOULD omit Group 2 rather than sending an empty group. However, a client MUST be able to accept an empty group.~~ See section 3.1.7 for details on returning Unsupported Attributes.

Group 3: Job Object Attributes

This is the same set of attributes as described in the Print-Job response (see section 3.2.1.2).

3.3.2 Send-URI Operation

This OPTIONAL operation is identical to the Send-Document operation (see section 3.3.1) except that a client MUST supply a URI reference ("document-uri" operation attribute) rather than the document data itself. If a Printer object supports this operation, clients can use both Send-URI or Send-Document operations to add new documents to an existing multi-document Job object. 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 object supports this operation, it MUST also support the Print-URI operation (see section 3.2.2).

The Printer object MUST validate the syntax and URI scheme of the supplied URI before returning a response, just as in the Print-URI operation.

3.3.3 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 IPP object MUST accept or reject the request based on the job's current state and transition the job to the indicated new state as follows:

Current "job-state"	New "job-state"	IPP object's response status code and action:
'pending'	'canceled'	'successful-ok'
'pending-held'	'canceled'	'successful-ok'
'processing'	'canceled'	'successful-ok'
'processing'	'processing'	'successful-ok' See Note 1
'processing'	'processing'	'client-error-not-possible' See Note 2
'processing-stopped'	'canceled'	'successful-ok'
'processing-stopped'	'processing-stopped'	'successful-ok' See Note 1
'processing-stopped'	'processing-stopped'	'client-error-not-possible' See Note 2
'completed'	'completed'	'client-error-not-possible'
'canceled'	'canceled'	'client-error-not-possible'

Current "job-state"	New "job-state"	IPP object's response status code and action:
'aborted'	'aborted'	'client-error-not-possible'

Note 1: If the implementation requires some measurable time to cancel the job in the 'processing' or 'processing-stopped' job states, the IPP object 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 4.3.8).

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

Access Rights: The authenticated user (see section 8.3) performing this operation must either be the job owner or an operator or administrator of the Printer object (see Sections 1 and 8.5). Otherwise, the IPP object MUST reject the operation and return: 'client-error-forbidden', 'client-error-not-authenticated', or 'client-error-not-authorized' as appropriate.

3.3.3.1 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 3.1.4.1.

Target:
Either (1) the "printer-uri" (uri) plus "job-id" (integer(1:MAX))or (2) the "job-uri" (uri) operation attribute(s) which define the target for this operation as described in section 3.1.5.

Requesting User Name:
The "requesting-user-name" (name(MAX)) attribute SHOULD be supplied by the client as described in section 8.3.

"message" (text(127)):
The client OPTIONALLY supplies this attribute. The Printer object OPTIONALLY supports 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. It is an implementation decision of how or where to display this message to the operator (if at all).

3.3.3.2 Cancel-Job Response

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

Group 1: Operation Attributes

Status Message:

In addition to the REQUIRED status code returned in every response, the response OPTIONALLY includes a "status-message" (text) operation attribute as described in sections 13 and 3.1.6.

~~If the job is already in the 'completed', 'aborted', or 'canceled' state, or the 'process-to-stop-point' value is set in the Job's "job-state-reasons" attribute, the Printer object MUST reject the request and return the 'client-error-not-possible' error status code.~~

Natural Language and Character Set:

The "attributes-charset" and "attributes-natural-language" attributes as described in section 3.1.4.2.

Group 2: Unsupported Attributes

~~This is a set of Operation attributes supplied by the client (in the request) that are not supported by the Printer object or that conflict with one another (see section 3.2.1.2 and the Implementer's Guide [IPP-IIG]). If the Printer object is not returning any Unsupported Attributes in the response, the Printer object SHOULD omit Group 2 rather than sending an empty group. However, a client MUST be able to accept an empty group.~~See section 3.1.7 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 of the Cancel-Job operation is accepted and when the job enters the 'canceled' job-state (see section 4.3.7), 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.

3.3.4 Get-Job-Attributes Operation

This REQUIRED operation allows a client to request the values of attributes of a Job object and it is almost identical to the Get-Printer-Attributes operation (see section 3.2.5). The only differences are that the operation is directed at a Job object rather than a Printer object, there is no "document-format" operation attribute used when querying a Job object, and the returned attribute group is a set of Job object attributes rather than a set of Printer object attributes.

For Jobs, the possible names of attribute groups are:

- 'job-template': ~~all~~the subset of the Job Template attributes that apply to a Job object (the first column of the table in Section 4.2) that the implementation supports for Job objects.
- 'job-description': ~~all~~the subset of the Job Description attributes specified in Section 4.3 that the implementation supports for Job objects.

- 'all': the special group 'all' that includes all ~~supported attributes.~~attributes that the implementation supports for Job objects. Issue 23

Since a client MAY request specific attributes or named groups, there is a potential that there is 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 object NEED NOT return 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.

It is NOT REQUIRED that a Job object support all attributes belonging to a group (since some attributes are OPTIONAL). However it is REQUIRED that each Job object support all group names.

3.3.4.1 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 object:

Group 1: Operation Attributes

Natural Language and Character Set:
The "attributes-charset" and "attributes-natural-language" attributes as described in section 3.1.4.1.

Target:
Either (1) the "printer-uri" (uri) plus "job-id" (integer(1:MAX)) or (2) the "job-uri" (uri) operation attribute(s) which define the target for this operation as described in section 3.1.5.

Requesting User Name:
The "requesting-user-name" (name(MAX)) attribute SHOULD be supplied by the client as described in section 8.3.

"requested-attributes" (1setOf keyword) :
The client OPTIONALLY supplies this attribute. The IPP object 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 IPP object MUST respond as if this attribute had been supplied with a value of 'all'.

3.3.4.2 Get-Job-Attributes Response

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

Group 1: Operation Attributes

Status Message:

In addition to the REQUIRED status code returned in every response, the response OPTIONALLY includes a "status-message" (text) operation attribute as described in sections 13 and 3.1.6.

Natural Language and Character Set:

The "attributes-charset" and "attributes-natural-language" attributes as described in section 3.1.4.2. The "attributes-natural-language" MAY be the natural language of the Job object, rather than the one requested.

Group 2: Unsupported Attributes

See section 3.1.7 for details on returning Unsupported Attributes.

~~This is a set of Operation attributes supplied by the client (in the request) that are not supported by the Printer object or that conflict with one another (see sections 3.2.1.2 and the Implementer's Guide [IPP-IIG]).~~ The response NEED NOT contain the "requested-attributes" operation attribute with any supplied values (attribute keywords) that were requested by the client but are not supported by the IPP object. If the Printer ~~object is not returning any Unsupported Attributes in the response, the Printer object SHOULD omit Group 2 rather than sending an empty group. However, a client MUST be able to accept an empty group.~~ does include unsupported attributes referenced in "requested-attributes" and such attributes include group names, such as 'all', the unsupported attributes MUST NOT include attributes described in the standard but not supported by the implementation. Issue 23

Group 3: Job Object Attributes

This is the set of requested attributes and their current values. The IPP object ignores (does not respond with) any requested attribute or value which is not supported or which 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 8). However, the IPP object MUST respond with the 'unknown' value for any supported attribute (including all REQUIRED attributes) for which the IPP object 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 4.1.

3.3.5 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 IPP object MUST accept or reject the request based on the job's current state and transition the job to the indicated new state as follows:

Current "job-state"	New "job-state"	IPP object's response status code and action:

Current "job-state"	New "job-state"	IPP object’s response status code and action:
’pending’	’pending-held’	’successful-ok’ See Note 1
’pending’	’pending’	’successful-ok’ See Note 2
’pending-held’	’pending-held’	’successful-ok’ See Note 1
’pending-held’	’pending’	’successful-ok’ See 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’

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

Note 2: If the IPP object 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 IPP object MUST make the job be a candidate for processing immediately (see Section 4.2.2) by putting the job in the ’pending’ state.

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’ states. If an operation is needed to hold jobs while in 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 object’s "operations-supported" (see Section 4.4.15) and the Job object’s "job-state" (see Section 4.3.7) attributes which operations are possible.

Access Rights: The authenticated user (see section 8.3) performing this operation must either be the job owner or an operator or administrator of the Printer object (see Sections 1 and 8.5). Otherwise, the IPP object MUST reject the operation and return: ’client-error-forbidden’, ’client-error-not-authenticated’, or ’client-error-not-authorized’ as appropriate.

3.3.5.1 Hold-Job Request

The groups and operation attributes are the same as for a Cancel-Job request (see section 3.3.3.1), with the addition of the following Group 1 Operation attribute:

"job-hold-until" (type3 keyword | name(MAX)):

The client OPTIONALLY supplies this Operation attribute. The IPP object MUST support this operation attribute in a Hold-Job request, if it supports the "job-hold-until" Job template attribute in create operations. See section 4.2.2. The IPP object SHOULD support the "job-hold-until" Job Template attribute for use in job create operations 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 IPP object copies the supplied operation attribute to the Job object, 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 IPP object accepts the request, returns the unsupported attribute or value in the Unsupported Attributes Group according to section 3.1.7, returns the 'successful-ok-ignored-or-substituted-attributes, and holds the job indefinitely until a client performs a subsequent Release-Job operation.

If the client (1) supplies 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 IPP object supports the "job-hold-until" operation attribute and there are no other reasons to hold the job, the IPP object MUST accept the operation and make the job be a candidate for processing immediately (see Section 4.2.2).

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

3.3.5.2 Hold-Job Response

The groups and attributes are the same as for a Cancel-Job response (see section 3.3.3.2).

3.3.6 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 object that had been supplied in the create or most recent Hold-Job or Restart-Job operation and remove its effect on the job. Issue 30 The IPP object MUST remove the 'job-hold-until-specified' value from the job's "job-state-reasons" attribute, if present. See section 4.3.8.

The IPP object MUST accept or reject the request based on the job's current state and transition the job to the indicated new state as follows:

Current "job-state"	New "job-state"	IPP object's response status code and action:
'pending'	'pending'	'successful-ok' No effect on the job.
'pending-held'	'pending-held'	'successful-ok' See 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'

Current "job-state"	New "job-state"	IPP object's response status code and action:
'aborted'	'aborted'	'client-error-not-possible'

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' applied to them, but are for any reason to keep the job from being a candidate for scheduling and processing, such as 'resources-are-not-ready'. See the "job-hold-until" attribute (section 4.2.2).

Access Rights: The authenticated user (see section 8.3) performing this operation must either be the job owner or an operator or administrator of the Printer object (see Sections 1 and 8.5). Otherwise, the IPP object 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 section 3.3.3.1 and 3.3.3.2).

3.3.7 Restart-Job Operation

This OPTIONAL operation allows a client to restart a job that is retained in the queue after processing has completed (see section 4.3.7.1).

The job is moved to the 'pending' job state and restarts at the beginning on the same IPP Printer object with the same attribute values. The Job Description 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 object MUST continue to use the same "job-uri" and "job-id" attribute values.

Note: If in the future an operation is needed that does not reset the job progress attributes, then a new operation will be defined which 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 IPP object MUST accept or reject the request based on the job's current state, transition the job to the indicated new state as follows:

Current "job-state"	New "job-state"	IPP object'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'	'successful-ok' - job is started over.
'completed'	'completed'	'client-error-not-possible' - see Note 1
'canceled'	'pending'	'successful-ok' - job is started over.
'canceled'	'canceled'	'client-error-not-possible' - see Note 1
'aborted'	'pending'	'successful-ok' - job is started over.

Current "job-state"	New "job-state"	IPP object's response status code and action:
'aborted'	'aborted'	'client-error-not-possible' - see Note 1

Note 1: If the Job Retention Period has expired for the job in this state, then the IPP object rejects the operation. See section 4.3.7.1.

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' states. If in the future an operation is needed to hold or restart jobs while in 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 8.3) performing this operation must either be the job owner or an operator or administrator of the Printer object (see Sections 1 and 8.5). Otherwise, the IPP object MUST reject the operation and return: 'client-error-forbidden', 'client-error-not-authenticated', or 'client-error-not-authorized' as appropriate.

3.3.7.1 Restart-Job Request

The groups and attributes are the same as for a Cancel-Job request (see section 3.3.3.1), with the addition of the following Group 1 Operation attribute:

"job-hold-until" (type3 keyword | name(MAX)):

The client OPTIONALLY supplies this attribute. The IPP object MUST support this Operation attribute in a Restart-Job request, if it supports the "job-hold-until" Job Template attribute in create operations. See section 4.2.2. Otherwise, the IPP object NEED NOT support the "job-hold-until" Operation attribute in a Restart-Job request.

If supplied and supported as specified in the Printer's "job-hold-until-supported" attribute, the IPP object copies the supplied Operation attribute to the Job object, 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 4.2.2.

If supplied, but the value is not supported, the IPP object accepts the request, returns the unsupported attribute or value in the Unsupported Attributes Group according to section 3.1.7, 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 IPP object accepts the request, returns the unsupported attribute with the out-of-band 'unsupported' value in the Unsupported Attributes Group according to section 3.1.7, returns the 'successful-ok-ignored-or-substituted-attributes' status code, and restarts the job, i.e., ignores the "job-hold-until" attribute.

If the client (1) supplies 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 IPP object supports the "job-hold-until" operation attribute and there are no other reasons to hold the job, the IPP object makes the job a candidate for processing immediately (see Section 4.2.2).

If the client does not supply a "job-hold-until" operation attribute in the request, the IPP object 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 4.2.2).

3.3.7.2 Restart-Job Response

The groups and attributes are the same as for a Cancel-Job response (see section 3.3.3.2).

Note: In the future an OPTIONAL Modify-Job operation may be specified that allows the client to modify other attributes before releasing the restarted job.

4. 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 which are included within the scope of this protocol. Many of these attributes are derived from other relevant specifications:

- Document Printing Application (DPA) [ISO10175]
- RFC 1759 Printer MIB [RFC1759]

Each attribute is uniquely identified in this document using a "keyword" (see section 12.2.1) which is the name of the attribute. The keyword is included in the section header 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.

4.1 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 3 and 4 includes the name of attribute syntax(es) in the heading (in parentheses). A conforming implementation of an attribute MUST include the semantics of the attribute syntax(es) so identified. Section 6.3 describes how the protocol can be extended with new attribute syntaxes.

The attribute syntaxes are specified in the following sub-sections, where the sub-section heading 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 sub-section heading 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 whose special encoding rules are defined in the "Encoding and Transport" specification [IPP-PRO]. Standard "out-of-band" values are: Issue 12 and Issue 15

'unknown': The attribute is supported by the IPP object, but the value is unknown to the IPP object for some reason.
'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 object, but the~~system~~ administrator has not yet configured a value.

~~The Encoding and Transport specification [IPP-PRO] defines mechanisms for passing "out-of-band" values.~~ All attributes in a request MUST have one or more values as defined in Sections 4.2 to 4.4. Thus clients MUST NOT supply attributes with "out-of-band" values. All attributes in a response MUST have one or more values as defined in Sections 4.2 to 4.4 or 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 sub-section heading to indicate the choice. Since each value MUST be tagged as to its attribute syntax in the protocol, a single-valued attribute instance may have any one of its attribute syntaxes and a multi-valued attribute instance may have a mixture of its defined attribute syntaxes.

4.1.1 '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 specification, 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 specification 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, objects and clients accept and return only one of the two syntax 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 3.1.4 or "attributes-natural-language" job attribute defined in section 4.3.18, 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 object 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 object uses the attribute syntax 'textWithLanguage' for text attributes (this is the Natural Language Override mechanism described in section 3.1.4).

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

4.1.1.1 'textWithoutLanguage'

The 'textWithoutLanguage' syntax indicates a value that is sequence of zero or more characters. Text strings are encoded using the rules of some charset. The Printer object MUST support the UTF-8 charset ~~[RFC2044]~~[RFC2279] and MAY support additional charsets to represent 'text' values, provided that the charsets are registered with IANA [IANA-CS]. See Section 4.1.7 for the specification of the 'charset' attribute syntax, including restricted semantics and examples of charsets.

4.1.1.2 'textWithLanguage'

The 'textWithLanguage' attribute syntax is a compound attribute syntax consisting of two parts: a 'textWithoutLanguage' part plus an additional 'naturalLanguage' (see section 4.1.8) 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 give text attribute, the 'textWithoutLanguage' part is limited to the maximum length defined for that attribute, but the 'naturalLanguage' part is always limited to 63 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 create request, the Printer object MUST accept and store with the Job object any natural language in the "attributes-natural-language" operation attribute, whether the Printer object supports that natural language or not. Furthermore, the Printer object MUST accept and store any 'textWithLanguage' attribute value, whether the Printer object 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~~"Encoding and Transport" document [IPP-PRO] for a detailed example of the 'textWithLanguage' attribute syntax.

4.1.2 '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, objects and clients accept and return only one of the two syntax per attribute. The syntax 'name' never appears "on-the-wire".

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

Some attributes are defined as 'type3 keyword | name'. These attributes support values that are either type3 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.

4.1.2.1 'nameWithoutLanguage'

The nameWithoutLanguage' syntax indicates a value that is sequence of zero or more characters so that its encoded form does not exceed MAX octets.

4.1.2.2 'nameWithLanguage'

The 'nameWithLanguage' attribute syntax is a compound attribute syntax consisting of two parts: a 'nameWithoutLanguage' part plus an additional 'naturalLanguage' (see section 4.1.8) 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.

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.

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

4.1.2.3 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 object'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 3.1.4.1) match. The matching rules are:

   a. the name parts match if the two names are identical character by character, except it is RECOMMENDED that case be ignored. For example: 'Ajax-letter-head-white' MUST match 'Ajax-letter-head-white' and SHOULD match 'ajax-letter-head-white' and 'AJAX-LETTER-HEAD-WHITE'.

   b. the Associated Natural-Language parts match if the shorter of the two meets the syntactic requirements of RFC 1766 [RFC1766] and matches byte for byte with the longer. For example, 'en' matches 'en', 'en-us' and 'en-gb', but matches neither 'fr' nor 'e'.

4.1.3 'keyword'

The 'keyword' attribute syntax is a sequence of characters, length: 1 to 255, containing only the US-ASCII [ASCII] 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 U.S. 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 U.S. 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 which 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 U.S. English, they MAY be mapped to other U.S. English words for U.S. 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 defined keyword values for that attribute are listed.

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 6.1 describes how the protocol can be extended with new keyword values. Examples of attribute name keywords:

"job-name"
"attributes-charset"

Note: This document uses "type1", "type2", and "type3" prefixes to the "keyword" basic syntax to indicate different levels of review for extensions (see section 6.1).

4.1.4 '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 are 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 specification are also used for corresponding attributes in other standards [RFC1759]. This syntax type is not used for attributes to which the~~system~~ administrator may assign values. Section 6.1 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 which 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 U.S. English, they MAY be mapped to other U.S. English words for U.S. English users, since the user interface is outside the scope of this document.

Note: 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 4.1. Therefore, attributes of type 'enum' start at '3'.

Note: This document uses "type1", "type2", and "type3" prefixes to the "enum" basic syntax to indicate different levels of review for extensions (see section 6.1).

4.1.5 'uri'

The 'uri' attribute syntax is any valid Uniform Resource Identifier or URI [RFC2396]. Most often, URIs are simply Uniform Resource Locators or 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 lower-cased values, this attribute syntax type conforms to the case-sensitive and case-insensitive rules specified in [RFC2396]. See also [IPP-IIG] for a discussion of case in URIs.

4.1.6 'uriScheme'

The 'uriScheme' attribute syntax is a sequence of characters representing a URI scheme according to RFC 2396 [RFC2396]. Though RFC 2396 requires that the values be case-insensitive, IPP requires all lower case values in IPP attributes to simplify comparing by IPP clients and Printer objects. Standard values for this syntax type are the following keywords:

'http': for HTTP schemed URIs (e.g., ~~"http:…")~~"http:...")
'https': for use with HTTPS schemed URIs (e.g., "https:...") (not on IETF standards track)
'ftp': for FTP schemed URIs (e.g., "ftp:...")
'mailto': for SMTP schemed URIs (e.g., "mailto:...")
'file': for file schemed URIs (e.g., "file:...")

A Printer object 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.

4.1.7 '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 and 'text' and 'name' attribute values. The syntax and semantics of this attribute syntax are specified in RFC 2046 [RFC2046] and contained in the IANA character-set Registry [IANA-CS] according to the IANA procedures [RFC2278]. Though RFC 2046 requires that the values be case-insensitive US-ASCII, IPP requires all lower case values in IPP attributes to simplify comparing by IPP clients and Printer objects. 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) represented as the UTF-8 [RFC2279] transfer encoding scheme in which US-ASCII is a subset charset.
'us-ascii': 7-bit American Standard Code for Information Interchange (ASCII), ANSI X3.4-1986 [ASCII]. That standard defines US-ASCII, but RFC 2045 [RFC2045] eliminates most of the control characters from conformant usage in MIME and IPP.
'iso-8859-1': 8-bit One-Byte Coded Character Set, Latin Alphabet Nr 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.
'iso-10646-ucs-2': ISO 10646 Universal Multiple-Octet Coded Character Set (UCS) represented as two octets (UCS-2), with the high order octet of each pair coming first (so-called Big Endian integer).

Some attribute descriptions MAY place additional requirements on charset values that may 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.

4.1.8 'naturalLanguage'

The 'naturalLanguage' attribute syntax is a standard identifier for a natural language and optionally a country. The values for this syntax type are defined by RFC 1766 [RFC1766]. Though RFC 1766 requires that the values be case-insensitive US-ASCII, IPP requires all lower case to simplify comparing by IPP clients and Printer objects. 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.

4.1.9 '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 2048 [RFC2048] for identifying a document format. The value MAY include a charset parameter, depending on the specification of the Media Type in the IANA Registry [IANA-MT]. Although most other IPP syntax types allow for only lower-cased values, this syntax type allows for mixed-case values which 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 [52, 56].
'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 ~~[RFC2044]~~[RFC2279]
~~'text/plain, charset=iso-10646-ucs-2': A plain text document in ISO 10646 represented in two octets (UCS-2) [ISO10646-1]~~
'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 - see IANA MIME Media Type registry
'application/octet-stream': Auto-sense - see below

4.1.9.1 Application/octet-stream -- Auto-Sensing the document format

One special type is 'application/octet-stream'. If the Printer object supports this value, the Printer object MUST be capable of auto-sensing the format of the document data, either as part of the create operation and/or at document processing time. During auto-sensing, a Printer may determine that the document-data has a format ~~data.~~ 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 codes, the Printer adds the 'unsupported-document-format' value to the job's "job-state-reasons" attribute. Issue 9 and Issue 10

If the Printer object's default value attribute "document-format-default" is set to 'application/octet-stream', the Printer object 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 may 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 object MUST trust the client supplied value more than the outcome of applying an automatic format detection mechanism. For example, the client may be requesting the printing of a PostScript file as a 'text/plain' document. The Printer object 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 object MUST use its auto-sensing mechanism on the client supplied document data whether auto-sensing is the Printer object'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 object 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 be used.

The maximum length of a 'mimeMediaType' value to represent IPP attribute values is 255 octets.

4.1.10 'octetString'

The 'octetString' attribute syntax is a sequence of octets encoded in a maximum of 1023 octets which is indicated in sub-section headers using the notation: octetString(MAX). This syntax type is used for opaque data.

4.1.11 'boolean'

The 'boolean' attribute syntax has only two values: 'true' and 'false'.

4.1.12 '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 may specify the range constraint explicitly in sub-section headers 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. However, the enforcement of that additional constraint is up to the IPP objects, not the protocol.

4.1.13 '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 header description for an attribute in this document whose attribute syntax is 'rangeOfInteger' (i.e., 'X:Y' indicating X as a minimum value and Y as a maximum value), then the constraint applies to both integers.

4.1.14 'dateTime'

The 'dateTime' attribute syntax is a standard, fixed length, 11 octet representation of the "DateAndTime" syntax as defined in RFC 1903 [RFC1903]. RFC 1903 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 which are localized to the natural language and date format of the user.

4.1.15 'resolution'

The 'resolution' attribute syntax specifies a two-dimensional resolution in the indicated units. It consists of 3 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 Printer MIB [RFC1759] suggested values. That is, the cross feed direction component resolution component is the same as the prtMarkerAddressabilityXFeedDir object in the Printer MIB, the feed direction component 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. Example: '300', '600', '3' indicates a 300 dpi cross-feed direction resolution, a 600 dpi feed direction resolution, since a '3' indicates dots per inch (dpi).

4.1.16 '1setOf X'

The '1setOf X' attribute syntax is 1 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 may specify that the values MUST be in a certain order for that attribute.

4.2 Job Template Attributes

Job Template attributes describe job processing behavior. Support for Job Template attributes by a Printer object is OPTIONAL (see section 12.2.3 for a description of support for OPTIONAL attributes). Also, clients OPTIONALLY supply Job Template attributes in create requests.

Job Template attributes conform to the following rules. For each Job Template attribute called "xxx":

1. If the Printer object supports "xxx" then it MUST support both a "xxx-default" attribute (unless there is a "No" in the table below) and a "xxx-supported" attribute. If the Printer object 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" may be supported for some document formats and not supported for other document formats. For example, it is expected that a Printer object would only support "orientation-requested" for some document formats (such as 'text/plain' or 'text/html') but not others (such as 'application/postscript').

2. "xxx" is OPTIONALLY supplied by the client in a create 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 object 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.

   Note: Since an administrator MAY change the default value attribute after a Job object has been submitted but before it has been processed, the default value used by the Printer object at job processing time may be different that the default value in effect at job submission time.

3. The "xxx-supported" attribute is a Printer object attribute that describes which job processing behaviors are supported by that Printer object. A client can query the Printer object 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 "job-sheet-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 create 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 object for its supported value attributes. The application SHOULD also query the default value attributes. If the application then limits selectable values to only those value that are supported, the application can guarantee that the values supplied by the client in the create 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 'staple', 'punch', and/or 'cover'. A client can query the Printer object 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 create request and the document data does not contain any corresponding finishing instructions. If the client does supply the "finishings" attribute in the create request, the IPP object validates the value or values to make sure that they are a subset of the supported values identified in the Printer object's "finishings-supported" attribute. See section 3.1.7.

The table 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 object. These are the attributes that can optionally be supplied by the client in a create request. The last two columns (labeled "Printer: Default Value Attribute" and "Printer: Supported Values Attribute") shows the name and syntax for each Job Template attribute in the Printer object (the default value attribute and the supported values attribute). 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 (type3 keyword | name)	job-hold-until-default (type3 keyword | name)	job-hold-until-supported (1setOf (type3 keyword | name))
job-sheets (type3 keyword | name)	job-sheets-default (type3 keyword | name)	job-sheets-supported (1setOf (type3 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 (type3 keyword | name)	media-default (type3 keyword | name)	media-supported (1setOf (type3 keyword | name)) media-ready

		(1setOf (~~–~~type3 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)

4.2.1 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 object 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 object. If privileged jobs are implemented outside ~~IPP/1.0,~~IPP/1.1, 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 object, the Printer object MUST use the value of the Printer object's "job-priority-default" 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" is also integer(1:100). This single integer value indicates the number of priority levels supported. The Printer object MUST take the value supplied by the client and map it to the closest integer in a sequence of n integers 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.

If the value of the Printer object's "job-priority-supported" is 10 and the client supplies values in the range 1 to 10, the Printer object maps them to 5, in the range 11 to 20, the Printer object maps them to 15, etc.

4.2.2 job-hold-until (type3 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 not other reasons to hold the job

’indefinite’: - the job is held indefinitely, until a client performs a Release-Job (section 3.3.6)

’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 ~~IPP/1.0).~~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 MUST add the ’job-hold-until-specified’ value to the job’s "job-state-reasons" attribute, 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-reason" attribute and, 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 object, the Printer object MUST use the value of the Printer object’s "job-hold-until-default" at job submission time (unlike most Job Template attributes that are used if necessary at job processing time).

4.2.3 job-sheets (type3 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 sheet is printed

An administrator MAY define additional values using the 'name' or 'keyword' attribute syntax, depending on implementation.

Note: The effect of this attribute on jobs with multiple documents MAY be affected by the "multiple-document-handling" job attribute (section 4.2.4), depending on the job sheet semantics.

4.2.4 multiple-document-handling (type2 keyword)

This attribute is relevant only if a job consists of two or more documents. This attribute MUST be supported if the Printer supports multiple documents per job (see sections 3.2.4 and 3.3.1). Issue 34 The attribute controls finishing operations and the placement of one or more print-stream pages into impressions and onto media sheets. 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 explanations, 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(*)".

Standard keyword values are:

'single-document': If a Job object 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 operations; that is, finishing would be performed on the concatenation of the sequences a(*),b(*). The Printer object MUST NOT force the data in each document instance to be formatted onto a new print-stream page, 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 object MUST force each copy (a(*),b(*)) to start on a new media sheet.

'separate-documents-uncollated-copies': If a Job object 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 operations; that is, the sets a(*) and b(*) would each be finished separately. The Printer object 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 object 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 operations; that is, the sets a(*) and b(*) would each be finished separately. The Printer object 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 object 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 sheet.

The 'single-document' value is the same as 'separate-documents-collated-copies' with respect to ordering of print-stream pages, but not media sheet generation, since 'single-document' will put the first page of the next document on the back side of a 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 sheet. In addition, if the "finishings" attribute specifies 'staple', then with 'single-document', documents a and b are stapled together as a single document with no regard to new sheets, with 'single-document-new-sheet', documents a and b are stapled together as a single document, but document b starts on a new sheet, but with 'separate-documents-uncollated-copies' and 'separate-documents-collated-copies', documents a and b are stapled separately.

Note: None of these values provide means to produce uncollated sheets within a document, i.e., where multiple copies of sheet n are produced before sheet n+1 of the same document.

The relationship of this attribute and the other attributes that control document processing is described in section 15.3.

4.2.5 copies (integer(1:MAX))

This 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 may be different from the number of uncollated copies which can be supported.

Note: The effect of this attribute on jobs with multiple documents is controlled by the "multiple-document-handling" job attribute (section 4.2.4) and the relationship of this attribute and the other attributes that control document processing is described in section 15.3.

4.2.6 finishings (1setOf type2 enum)

This attribute identifies the finishing operations 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 are:

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 is site-defined.
'5'	'punch': This value indicates that holes are required in the finished document. The exact number and placement of the holes is site-defined The punch specification MAY be satisfied (in a site- 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 is 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 is implementation 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 is implementation and/or site-defined.

'10'-'19' reserved for future generic finishing enum values.

The following values are more specific; they indicate a corner or an edge as if the document were a portrait document (see below):

'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 is implementation 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 is implementation 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 is implementation 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 is implementation 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).

The 'staple-xxx' values are specified with respect to the document as if the document were a portrait document. If the document is actually a landscape or a reverse-landscape document, 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 from portrait, i.e., anti-clockwise). 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 from portrait, i.e., clockwise).

The angle (vertical, horizontal, angled) of each staple with respect to the document depends on the implementation which may 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 4.2.4) and the relationship of this attribute and the other attributes that control document processing is described in section 15.3.

If the client supplies a value of 'none' along with any other combination of values, it is the same as if only that other combination of values had been supplied (that is the 'none' value has no effect).

4.2.7 page-ranges (1setOf rangeOfInteger (1:MAX))

This attribute identifies the range(s) of print-stream pages that the Printer object uses for each copy of each document which are to be printed. Nothing is printed for any pages identified that do not exist in the document(s). Ranges MUST be in ascending order, for example: 1-3, 5-7, 15-19 and MUST NOT overlap, so that a non-spooling Printer object can process the job in a single pass. If the ranges are not ascending or are overlapping, the IPP object MUST reject the request and return the 'client-error-bad-request' status code. The attribute is associated with print-stream pages not application-numbered pages (for example, 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 "copy" for purposes of the specified page range(s). When "multiple-document-handling" is 'single-document', the Printer object MUST apply each supplied page range once to the concatenation of the print-stream 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 object 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 documents.

In most cases, the exact pages to be printed will be generated by a device driver and this attribute would not be required. However, when printing an archived document which has already been formatted, the end user may elect to print just a subset of the pages contained in the document. In this case, if page-range = 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.

"page-ranges-supported" is a boolean value indicating whether or not the printer is capable of supporting the printing of page ranges. This capability may 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 will be printed.

Note: The effect of this attribute on jobs with multiple documents is controlled by the "multiple-document-handling" job attribute (section 4.2.4) and the relationship of this attribute and the other attributes that control document processing is described in section 15.3.

4.2.8 sides (type2 keyword)

This attribute specifies how print-stream pages are to be imposed upon the sides of an instance of a selected medium, i.e., an impression.

The standard keyword values are:

’one-sided’: imposes each consecutive print-stream page upon the same side of consecutive media sheets.

’two-sided-long-edge’: imposes each consecutive pair of print-stream pages upon front and back sides of consecutive media sheets, such that the orientation of each pair of print-stream pages 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 print-stream pages upon front and back sides of consecutive media sheets, such that the orientation of each pair of print-stream pages 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’.

’two-sided-long-edge’, ’two-sided-short-edge’, ’tumble’, and ’duplex’ all work the same for portrait or landscape. However ’head-to-toe’ is ’tumble’ in portrait but ’duplex’ in landscape. ’head-to-head’ also switches between ’duplex’ and ’tumble’ when using portrait and landscape modes.

Note: The effect of this attribute on jobs with multiple documents is controlled by the "multiple-document-handling" job attribute (section 4.2.4) and the relationship of this attribute and the other attributes that control document processing is described in section 15.3.

4.2.9 number-up (integer(1:MAX))

This attribute specifies the number of print-stream pages to impose upon a single side of an instance of a selected medium. For example, if the value is:

Value	Description
’1’	the Printer MUST place one print-stream page on a single side of an instance of the selected medium (MAY add some sort of translation, scaling, or rotation).
’2’	the Printer MUST place two print-stream pages on a single side of an instance of the selected medium (MAY add some sort of translation, scaling, or rotation).
’4’	the Printer MUST place four print-stream pages on a single side of an instance of the selected medium (MAY add some sort of translation, scaling, or rotation).

This attribute primarily controls the translation, scaling and rotation of print-stream pages.

Note: The effect of this attribute on jobs with multiple documents is controlled by the "multiple-document-handling" job attribute (section 4.2.4) and the relationship of this attribute and the other attributes that control document processing is described in section 15.3.

4.2.10 orientation-requested (type2 enum)

This attribute indicates the desired orientation for printed print-stream pages; it does not describe the orientation of the client-supplied print-stream pages.

For some document formats (such as 'application/postscript'), the desired orientation of the print-stream pages is specified within the document data. This information is generated by a device 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 object to bind the desired orientation to the document data after it has been submitted. It is expected that a Printer object would only support "orientations-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 4.2, item 1, points out that a Printer object may 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 object will support "orientation-requested" for only a subset of the supported document formats.

Standard enum values are:

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 print-stream page to be imaged by +90 degrees with respect to the medium (i.e. anti-clockwise) 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 print-stream 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 print-stream 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.

Note: The effect of this attribute on jobs with multiple documents is controlled by the "multiple-document-handling" job attribute (section 4.2.4) and the relationship of this attribute and the other attributes that control document processing is described in section 15.3.

4.2.11 media (type3 keyword | name(MAX))

This attribute identifies the medium that the Printer uses for all impressions of the Job.

The values for "media" include medium-names, medium-sizes, input-trays and electronic forms so that one attribute specifies the media. If a Printer object 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 object 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 object 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 object 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 its prints each page.

Standard keyword values are (taken from ISO DPA and the Printer MIB) and are listed in section 14. 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. If an IPP object supports "media-supported", it NEED NOT support "media-ready".

The relationship of this attribute and the other attributes that control document processing is described in section 15.3.

4.2.12 printer-resolution (resolution)

This attribute identifies the resolution that Printer uses for the Job.

4.2.13 print-quality (type2 enum)

This attribute specifies the print quality that the Printer uses for the Job.

The standard enum values are:

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

4.3 Job Description Attributes

The attributes in this section form the attribute group called "job-description". The following table summarizes these attributes. The third column indicates whether the attribute is a REQUIRED attribute that MUST be supported by Printer objects. If it is not indicated as REQUIRED, then it is OPTIONAL. The maximum size in octets for 'text' and 'name' attributes is indicated in parenthesizes.

Attribute	Syntax	REQUIRED?
job-uri	uri	REQUIRED
job-id	integer(1:MAX)	REQUIRED
job-printer-uri	uri	REQUIRED
job-more-info	uri	
job-name	name (MAX)	REQUIRED
job-originating-user-name	name (MAX)	REQUIRED
job-state	type1 enum	REQUIRED
job-state-reasons	1setOf type2 keyword	REQUIRED
job-state-message	text (MAX)	
number-of-documents	integer (0:MAX)	
output-device-assigned	name (127)	
time-at-creation	integer (0:MAX)	REQUIRED
time-at-processing	integer (0:MAX)	REQUIRED
time-at-completed	integer (0:MAX)	REQUIRED
job-printer-up-time	integer (1:MAX)	REQUIRED
number-of-intervening-jobs	integer (0:MAX)	
job-message-from-operator	text (127)	
job-k-octets	integer (0:MAX)	
job-impressions	integer (0:MAX)	
job-media-sheets	integer (0:MAX)	
job-k-octets-processed	integer (0:MAX)	
job-impressions-completed	integer (0:MAX)	
job-media-sheets-completed	integer (0:MAX)	
attributes-charset	charset	REQUIRED

attributes-natural-language	naturalLanguage	REQUIRED

4.3.1 job-uri (uri)

This REQUIRED attribute contains the URI for the job. The Printer object, on receipt of a new job, generates a URI which identifies the new Job. The Printer object returns the value of the "job-uri" attribute as part of the response to a create request. The precise format of a Job URI is implementation dependent. If the Printer object supports more than one URI and there is some relationship between the newly formed Job URI and the Printer object's URI, the Printer object uses the Printer URI supplied by the client in the create request. For example, if the create request comes in over a secure channel, the new Job URI MUST use the same secure channel. This can be guaranteed because the Printer object is responsible for generating the Job URI and the Printer object is aware of its security configuration and policy as well as the Printer URI used in the create request.

For a description of this attribute and its relationship to "job-id" and "job-printer-uri" attribute, see the discussion in section 2.4 on "Object Identity".

4.3.2 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 which identifies the new Job on that Printer. The Printer returns the value of the "job-id" attribute as part of the response to a create request. The 0 value is not included to allow for compatibility with SNMP index values which also cannot be 0.

For a description of this attribute and its relationship to "job-uri" and "job-printer-uri" attribute, see the discussion in section 2.4 on "Object Identity".

4.3.3 job-printer-uri (uri)

This REQUIRED attribute identifies the Printer object that created this Job object. When a Printer object creates a Job object, it populates this attribute with the Printer object URI that was used in the create request. This attribute permits a client to identify the Printer object that created this Job object when only the Job object's URI is available to the client. The client queries the creating Printer object to determine which languages, charsets, operations, are supported for this Job.

For a description of this attribute and its relationship to "job-uri" and "job-id" attribute, see the discussion in section 2.4 on "Object Identity".

4.3.4 job-more-info (uri)

Similar to "printer-more-info", this attribute contains the URI referencing some resource with more information about this Job object, perhaps an HTML page containing information about the Job.

4.3.5 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" attribute value. 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 create request (see Section 3.2.1.1). If, however, the "job-name" operation attribute is not supplied by the client in the create request, the Printer object, 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 Content information.

4.3.6 job-originating-user-name (name(MAX))

This REQUIRED attribute contains the name of the end user that submitted the print job. The Printer object 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 is not available, does the Printer object use the value supplied by the client in the "requesting-user-name" operation attribute of the create operation (see Section 8).

Note: The Printer object needs to keep an internal originating user id of some form, typically as a credential of a principal, with the Job object. Since such an internal attribute is implementation-dependent and not of interest to clients, it is not specified as a Job Description attribute. This originating user id is used for authorization checks (if any) on all subsequent operation.

4.3.7 job-state (type1 enum)

This REQUIRED attribute identifies the current state of the job. Even though the IPP protocol defines ~~eight~~seven values for job states (plus the out-of-band ’unknown’ value - see Section 4.1), implementations only need to support those states which 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 object implementation.

Standard enum values are:

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-reason" attribute MUST indicate why the job is no longer a candidate for processing.

’5’ ’processing’: One or more of:

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 object 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, awaiting 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 object's "printer-state", "printer-state-reasons", and "printer-state-message" attributes.

Implementations MAY, though they NEED NOT, 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 IPP object is in the process of canceling or aborting the job. Most implementations won't bother with this nuance.

'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-reason" 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 object'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 object has completed canceling the job and all job status attributes have reached their final values for the job. While the Printer object 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' value. When the job moves to the 'canceled' state, the 'processing-to-stop-point' value, if present, MUST be removed, but the '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 and all job status attributes have reached their final values for the job. While the Printer object 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 and 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: 'completed-successfully', 'completed-with-warnings', or 'completed-with-errors' values.

The final value for this attribute MUST be one of: '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 4.3.7.1.

The following figure shows the normal job state transitions.

```
                                                   +----> canceled
                                                  /
    +----> pending --------> processing ---------+------> completed
    |         ^                   ^               \
--->+         |                   |                +----> aborted
    |         v                   v               /
    +----> pending-held    processing-stopped ---+
```

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.

Jobs reach one of the three terminal states: 'completed', 'canceled', or 'aborted', after the jobs have completed all activity, including stacking output media, after the jobs have completed all activity, and all job status attributes have reached their final values for the job.

Note: As with all other IPP attributes, if the implementation can~~-~~not determine the correct value for this attribute, it SHOULD respond with the out-of-band value 'unknown' (see section 4.1) rather than try to guess at some possibly incorrect value and give the end user the wrong impression about the state of the Job object. For example, if the implementation is just a gateway into some printing system ~~that does not provide~~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 object's state

~~might literally be 'unknown'.~~to 'completed', provided that it also sets the 'queued-in-device' value in the job's "job-state-reasons" attribute (see section 4.3.8). Issue 14

4.3.7.1 Partitioning of Job States

This section partitions the 7 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 Description attribute for use with the Restart-Job operation.

Job Not Completed: When a job is in the 'pending', 'pending-held', 'processing', or 'processing-stopped' states, 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 object 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 may restart the job using the Restart-Job operation. If the IPP object supports Issue 30 the Restart-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 4.3.8) during the Job Retention phase.

Job History: After the Job Retention phase expires for a job, the Printer object deletes the document data for the job and the job becomes part of the Job History. The Printer object MAY also delete any number of the job attributes. Since the job is no longer restartable, the Printer object MUST remove the 'job-restartable' value from the job's "job-state-reasons" attribute, if present.

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.

4.3.8 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.

~~Implementation~~While implementation of this attribute is REQUIRED, implementation of these values is ~~OPTIONAL, i.e., a Printer NEED NOT implement them, even if (1) the output device supports the functionality represented by the reason and (2) is available to the Printer object implementation.~~OPTIONAL. Issue 30 These values MAY be used with any job state or states for which the reason makes sense. 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-~~ "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), starting with the 'job-incoming' value:

'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': The Create-Job operation has been accepted by the Printer, but the Printer is expecting additional Send-Document and/or Send-URI operations and/or is accessing/accepting document data.

'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' remains. If a Printer starts processing after it has received all data, the Printer removes the 'job-data-insufficient' reason and the 'job-incoming' at the same time. Issue 13

'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. Whether the Printer aborts the job and moves the job to the 'aborted' job state or prints all documents that are accessible and moves the job to the 'completed' job state and adds the 'completed-with-errors' value in the job's "job-state-reasons" attribute depends on implementation and/or site policy. Issue 35

'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, (3) the client crashed or failed to close the job before the time-out period. See section 4.4.31.

'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.

'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 printer's 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 may 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’: Job is in the ’processing’ state, but more specifically, the Printer is interpreting the document data.

’job-queued’: Job is in the ’processing’ state, but more specifically, the Printer has queued the document data.

’job-transforming’: Job is in the ’processing’ state, but more specifically, the Printer is interpreting document data and producing another electronic representation.

’job-queued-for-marker’: 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 (or back into) the ’processing’ state when marking does begin. Issue 31

’job-printing’: The output device is marking media. This value is useful for Printers which spend a great deal of time processing (1) when no marking is happening and then 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 object, or by some other authorized end-user, such as a member of the job owner’s security group.

’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 may 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.

’job-canceled-at-device’: The job was canceled by an unidentified local user, i.e., a user at a console at the device.

’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.

’unsupported-compression’: The job was aborted by the system because the Printer determined while attempting to decompress the document-data’s that the compression is actually not among those supported by the Printer. Issue 6

’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-reason. Issue 6

’unsupported-document-format’: The job was aborted by the system because the document-data’s document-format is not among those supported by the Printer. If the client specifies the document-format as ’application/octet-stream’, the printer MAY abort the job and post this reason even though the format is a member of the "document-format-supported" printer attribute, but not among the auto-sensed document-formats. Issue 3

’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-reason. Issue 3

’processing-to-stop-point’: The requester has issued a ~~Cancel-job~~Cancel-Job operation or the Printer object has aborted the job, but is still performing some actions on the job until a specified stop point occurs or job termination/cleanup is completed.

This reason is recommended to be used in conjunction with the ’processing’ job state to indicate that the Printer object is still performing some actions on the job while the job remains in the ’processing’ state. After all the job’s job description attributes have stopped incrementing, the Printer object moves the job from the ’processing’ state to the ’canceled’ or ’aborted’ job states.

’service-off-line’: The Printer is off-line 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.

’job-completed-with-warnings’: The job completed with warnings.

’job-completed-with-errors’: The job completed with errors (and possibly warnings too).

’job-restartable’ - This job is retained (see section 4.3.7.1) and is currently able to be restarted using the Restart-Job operation (see section 3.3.7). If ’job-restartable’ is a value of the job’s ’job-state-reasons’ attribute, then the IPP object MUST accept a Restart-Job operation for that job.

’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 note in section 4.3.7. Issue 14

4.3.9 job-state-message (text(MAX))

This attribute specifies information about the "job-state" and "job-state-reasons" attributes in human readable text. If the Printer object supports this attribute, the Printer object 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 3.1.4.1).

Note: the value SHOULD NOT contain additional information not contained in the values of the "job-state" and "job-states-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, a new attribute~~consumption or specific document access errors, new attributes with keyword values, needs to be developed and registered.

4.3.10 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 object or not.

Implementations supporting the OPTIONAL Create-Job/Send-Document/Send-URI operations SHOULD support this attribute so that clients can query the number of documents in each job.

4.3.11 output-device-assigned (name(127))

This attribute identifies the output device to which the Printer object has assigned this job. If an output device implements an embedded Printer object, the Printer object NEED NOT set this attribute. If a print server implements a Printer object, the value MAY be empty (zero-length string) or not returned until the Printer object assigns an output device to the job. This attribute is particularly useful when a single Printer object support multiple devices (so called "fan-out").

4.3.12 Event Time Job Description Attributes Issue 17

This section defines the Job Description attributes that indicate the time at which certain events occur for a job. The attribute syntax MUST be either 'integer' or 'dateTime' for any response in which the "version-number" parameter is supplied as '1.1', but MUST be an 'integer' for any response in which the "version-number" parameter is supplied as '1.0', for compatibility with IPP/1.0 [RFC2566]. See section 3.1.8.

In order to populate these Event Time Job Description Attributes, the Printer object copies either:

1. the value in its "printer-current-time" attribute for the 'dateTime' value at the time the event occurred if the printer supports the attribute "printer-current-time" and its value is not the out-of-band 'no-value' value,

2. the value in its "printer-up-time" attribute for the 'integer' value at the time the event occurred otherwise

Note: because the time MAY become known to the Printer some time after power-up, a client could receive jobs that contain some Event Time Job Description Attributes that use the 'integer' time tick representation while the later events use the 'dateTime' date/time representation.

If the Printer implementation keeps jobs persistently across power cycles, then an implementation MUST reset its "printer-up-time" attribute to 1 on each power-up. In addition, an implementation that uses the 'integer' form MUST change all of its Event Time Job Description attributes for those persistent jobs either:

1. to 0 to indicate that the event happened before the most recent power up

2. to the negative of the number of seconds before the most recent power-up that the event took place, though the negative number NEED NOT reflect the exact number of seconds

An implementation that uses the 'dateTime' form does not change the values of any of its Event Time Job Description Attributes for persistent jobs on power-up.

~~4.3.12~~4.3.12.1 time-at-creation ~~(integer(0:MAX))~~(integer(MIN:MAX) | dateTime)

This REQUIRED attribute indicates the~~point in~~ time at which the Job object was created. ~~In order to populate this attribute, the Printer object uses the value in its "printer-up-time" attribute at the time the Job object is created.~~

~~4.3.13~~4.3.12.2 time-at-processing ~~(integer(0:MAX))~~(integer(MIN:MAX) | dateTime)

This REQUIRED attribute indicates the~~point in~~ time at which the Job object began processing. ~~In order to populate this attribute, the Printer object uses the value in its "printer-up-time" attribute at the time the Job object is moved into~~The out-of-band 'no-value' value is returned if the job has not yet been in the 'processing' state ~~for the first time.~~ (see the beginning of Section 4.1)..

~~4.3.14~~4.3.12.3 time-at-completed ~~(integer(0:MAX))~~(integer(MIN:MAX) | dateTime)

This REQUIRED attribute indicates the~~point in~~ time at which the Job object completed (or was cancelled 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 4.1).

4.3.12.4 job-printer-up-time (integer(1:MAX)) Issue 17

~~In order to populate this attribute, the Printer object uses the value in its~~This REQUIRED Job Description 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 Description attribute (see Section 4.4.29).

Note: 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 Description Attributes in order to display time attributes to a user when the IPP Printer returns them using the 'integer' attribute syntax. The difference between this attribute ~~at the time the Job object is moved into the 'completed' or 'canceled' or 'aborted' state.~~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.

4.3.13 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.

4.3.14 job-message-from-operator (text(127))

This attribute provides a message from an operator, system administrator or "intelligent" process to indicate to the end user the reasons for modification or other management action taken on a job.

4.3.15 Job Size Attributes

This sub-section 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 object may try to compute the value if it is not supplied in the create request. Even if the client does supply a value for these three attributes in the create request, the Printer object MAY choose to change the value if the Printer object is able to compute a value which is more accurate than the client supplied value. The Printer object may be able to determine the correct value for these attributes either right at job submission time or at any later point in time.

4.3.15.1 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 MUST also 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.

~~Note: This attribute and the following two attributes ("job-impressions" and "job-media-sheets") are not intended to be counters; they are intended to be useful routing and scheduling information if known. For these three attributes, the Printer object may try to compute the value if it is not supplied in the create request. Even if the client does supply a value for these three attributes in the create request, the Printer object MAY choose to change the value if the Printer object is able to compute a value which is more accurate than the client supplied value. The Printer object may be able to determine the correct value for these three attributes either right at job submission time or at any later point in time.~~

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

This attribute specifies the total size in number of impressions of the document(s) being submitted (see the definition of impression in section 12.2.5).

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 MUST also 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.

~~See the Note in the "job-k-octets" attribute that also applies to this attribute.~~

4.3.15.3 job-media-sheets (integer(0:MAX))

This 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 system 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".

~~See the Note in the "job-k-octets" attribute that also applies to this attribute.~~

4.3.16 Job Progress Attributes

This sub-section defines job attributes that describe the progress of the job. These attributes are intended to be counters. That is, the value 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.

4.3.16.1 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.

~~Note: This attribute and the following two attributes ("job-impressions-completed" and "job-sheets-completed") are intended to be counters. That is, the value 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.~~

4.3.16.2 job-impressions-completed (integer(0:MAX))

This job 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.

~~See the note in "job-k-octets-processed" which also applies to this attribute.~~

4.3.16.3 job-media-sheets-completed (integer(0:MAX))

This job attribute specifies the media-sheets completed marking and stacking for the entire job so far whether those sheets have been processed on one side or on both.

~~See the note in "job-k-octets-processed" which also applies to this attribute.~~

4.3.17 attributes-charset (charset)

This REQUIRED attribute is populated using the value in the client supplied "attributes-charset" attribute in the create request. It identifies the charset (coded character set and encoding method) used by any Job attributes with attribute syntax 'text' and 'name' that were supplied by the client in the create request. See Section 3.1.4 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 object. The internal charset is implementation-defined. The IPP object MUST convert from whatever the internal charset is to that being requested in an operation as specified in Section 3.1.4.

4.3.18 attributes-natural-language (naturalLanguage)

This REQUIRED attribute is populated using the value in the client supplied "attributes-natural-language" attribute in the create request. It identifies the natural language used for any Job attributes with attribute syntax 'text' and 'name' that were supplied by the client in the create request. See Section 3.1.4 for a complete description of the "attributes-natural-language" operation attribute. See Sections 4.1.1.2 and 4.1.2.2 for how a Natural Language Override may be supplied explicitly for each 'text' and 'name' attribute value that differs from the value identified by the "attributes-natural-language" attribute.

4.4 Printer Description Attributes

These attributes form the attribute group called "printer-description". The following table summarizes these attributes, their syntax, and whether or not they are REQUIRED for a Printer object 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 parenthesizes.

Note: How these attributes are set by an Administrator is outside the scope of this ~~specification.~~IPP/1.1 document.

Attribute	Syntax	REQUIRED?
printer-uri-supported	1setOf uri	REQUIRED
uri-security-supported	1setOf type2 keyword	REQUIRED
uri-authentication-supported	1setOf type2 keyword	REQUIRED
printer-name	name (127)	REQUIRED
printer-location	text (127)	
printer-info	text (127)	
printer-more-info	uri	
printer-driver-installer	uri	
printer-make-and-model	text (127)	
printer-more-info-manufacturer	uri	
printer-state	type1 enum	REQUIRED
printer-state-reasons	1setOf type2 keyword	REQUIRED
printer-state-message	text (MAX)	
ipp-versions-supported	1setOf type2 keyword	REQUIRED
operations-supported	1setOf type2 enum	REQUIRED
ipp-multiple-document-jobs-supported	boolean	
charset-configured	charset	REQUIRED
charset-supported	1setOf charset	REQUIRED
natural-language-configured	naturalLanguage	REQUIRED
generated-natural-language-supported	1setOf naturalLanguage	REQUIRED
document-format-default	mimeMediaType	REQUIRED
document-format-supported	1setOf mimeMediaType	REQUIRED

printer-is-accepting-jobs	boolean	REQUIRED
queued-job-count	integer (0:MAX)	~~RECOMMENDED~~ REQUIRED
printer-message-from-operator	text (127)	
color-supported	boolean	
reference-uri-schemes-supported	1setOf uriScheme	
pdl-override-supported	type2 keyword	REQUIRED
printer-up-time	integer (1:MAX)	REQUIRED
printer-current-time	dateTime	
multiple-operation-time-out	integer (1:MAX)	
compression-supported	1setOf type3 keyword	REQUIRED
job-k-octets-supported	rangeOfInteger (0:MAX)	
job-impressions-supported	rangeOfInteger (0:MAX)	
job-media-sheets-supported	rangeOfInteger (0:MAX)	
pages-per-minute	integer(0:MAX)	
pages-per-minute-color	integer(0:MAX)	

4.4.1 printer-uri-supported (1setOf uri)

This REQUIRED Printer attribute contains at least one URI for the Printer object. It OPTIONALLY contains more than one URI for the Printer object. An administrator determines a Printer object's URI(s) and configures this attribute to contain those URIs by some means outside the scope of ~~IPP/1.0.~~this IPP/1.1 document. The precise format of this URI is implementation dependent and depends on the protocol. See the next ~~section~~two sections for a description of the "uri-security-supported" and "uri-authentication-supported" attributes, both of which ~~is~~are the REQUIRED companion attributes to this "printer-uri-supported" attribute. See section 2.4 on Printer object identity and section 8.2 on security and URIs for more information.

4.4.2 uri-authentication-supported (1setOf type2 keyword) Issue 2

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 object uses the specified mechanism to identify the authenticated user (see section 8.3) . 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 [IPP-PRO] 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 object assumes that the authenticated user is "anonymous".
’requesting-user-name’: When a client performs an operation whose target is the associated URI, the Printer object assumes that the authenticated user is specified by the "requesting-user-name" Operation attribute (see section 8.3). If the "requesting-user-name" attribute is absent in a request, the Printer object assumes that the authenticated user is "anonymous".
’basic’: When a client performs an operation whose target is the associated URI, the Printer object challenges the client with HTTP basic authentication. The Printer object 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 object challenges the client with HTTP digest authentication. The Printer object 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 object expects the client to provide a certificate. The Printer object assumes that the authenticated user is the textual name contained within the certificate.

4.4.3 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 object via that URI. See [IPP-PRO] 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.
’ssl3’: SSL3 [SSL] is the secure communications channel protocol in use for the given URI.
’tls’: TLS [RFC2246] is the secure communications channel protocol in use for the given URI.

This attribute is orthogonal to the specification of a Client Authentication mechanism. Specifically, ’none’ does not exclude Client Authentication. See section 4.4.2. Issue 21

Consider the following example. For a single Printer object, an administrator configures the "printer-uri-supported", "uri-authentication-supported" and "uri-security-supported" attributes as follows:

"printer-uri-supported": ’~~http~~xxx://acme.com/open-use-printer’, ’~~http~~xxx://acme.com/restricted-use-printer’, ’~~http~~xxx://acme.com/private-printer’
"uri-authentication-supported": ’none’, ’digest’, ’basic’
"uri-security-supported": ’none’, ’none’, ~~’ssl3’~~’tls’

Note: ’xxx’ is not a valid scheme. See the IPP/1.1 "Transport and Encoding" specification [IPP-PRO] for the actual URI schemes to be used in object target attributes.

In this case, one Printer object has three URIs.

- For the first URI, ’~~http~~xxx://acme.com/open-use-printer’, the value ’none’ in "uri-security-supported" indicates that there is no secure channel protocol configured to run under HTTP. The ~~name implies that there is no Basic or Digest authentication being used, but it is up to the client to determine that while using HTTP~~value of ’none’ in "uri-authentication-supported" indicates that all users are ’anonymous’. There will be no challenge and the Printer will ignore ~~underneath the IPP application protocol.~~"requesting-user-name".
- For the second URI, ’~~http~~xxx://acme.com/restricted-use-printer’, the value ’none’ in "uri-security-supported" indicates that there is no secure channel protocol configured to run under HTTP. ~~In this case, although the name does imply that there is some sort of Basic or Digest authentication being used within HTTP, it is up to the~~The value of ’digest’ in "uri-authentication-supported" indicates that the Printer will issue a challenge and that the Printer will use the name ~~client to determine that while using HTTP and by processing any ’401 Unauthorized’ HTTP error messages.~~supplied by the digest mechanism to determine the authenticated user (see section 8.3).
- For the third URI, ’~~http~~xxx://acme.com/private-printer’, the value ’ssl3’ the value ’tls’ in "uri-security-supported" indicates that ~~SSL3~~TLS is being used to secure the channel. The client SHOULD be prepared to use ~~SSL3~~TLS framing to negotiate an acceptable ciphersuite to use while communicating with the Printer object. In this case, the name implies the use of a secure communications channel, but the fact is made explicit by the presence of the ~~’ssl3’~~’tls’ value in "uri-security-supported". The client does not need to resort to understanding which security it must use by following naming conventions or by parsing the URI to determine which security mechanisms are ~~implied.~~
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 digest mechanism to determine the authenticated user (see section 8.3) . Because this challenge occurs in a tls session, the channel is secure.

It is expected that many IPP Printer objects will be configured to support only one channel (either configured to use ~~SSL3 access or not), and will therefore only ever~~TLS access or not) and only one authentication mechanism. Such Printer objects only have one URI listed in the "printer-uri-supported" attribute. No matter the configuration of the Printer object (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.

4.4.4 printer-name (name(127))

This REQUIRED Printer attribute contains the name of the Printer object. 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 may be the last part of the printer's URI or it may be unrelated. In non-US-English locales, a name may contain characters that are not allowed in a URI.

4.4.5 printer-location (text(127))

This Printer attribute identifies the location of the device. This could include things like: "in Room 123A, second floor of building XYZ".

4.4.6 printer-info (text(127))

This Printer attribute identifies the descriptive information about this Printer object. 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, 1997, please find a new printer".

4.4.7 printer-more-info (uri)

This Printer attribute contains a URI used to obtain more information about this specific Printer object. For example, this could be an HTTP type 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, end user assistance). The device manufacturer may initially populate this attribute.

4.4.8 printer-driver-installer (uri)

This Printer attribute contains a URI to use to locate the driver installer for this Printer object. This attribute is intended for consumption by automata. The mechanics of print driver installation is outside the scope of ~~IPP.~~this IPP/1.1 document. The device manufacturer may initially populate this attribute.

4.4.9 printer-make-and-model (text(127))

This Printer attribute identifies the make and model of the device. The device manufacturer may initially populate this attribute.

4.4.10 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, print 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 may initially populate this attribute.

4.4.11 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 object need only update this attribute before responding to an operation which requests the attribute; the Printer object NEED NOT update this attribute continually, since asynchronous event notification is not part of ~~IPP/1.0.~~IPP/1.1. A Printer NEED NOT implement all values if they are not applicable to a given implementation.

The following standard enum values are defined:

Value Symbolic Name and Description

'3' 'idle': If a Printer receives a job (whose required resources are ready) while in this state, such a job MUST transit into the 'processing' state immediately. If the "printer-state-reasons" attribute contains any reasons, they MUST be reasons that would not prevent a job from transiting into the 'processing' state immediately, e.g., 'toner-low'.

~~Note:~~If a Printer can interpret one or more jobs while marking a job, then it is idle if it is available to interpret jobs even while marking a job. Issue 31

If a Printer controls more than one output device, the above definition implies that a Printer is 'idle' if at least one output device is idle, i.e., the IPP Printer is available to immediately start processing a job if a client submitted it.

'4' 'processing': If a Printer receives a job (whose required resources are ready) while in this state, such a job MUST transit into the 'pending' state immediately. Such a job MUST transit into the 'processing' state only after jobs ahead of it complete. If the "printer-state-reasons" attribute contains any reasons, they MUST be reasons that do not prevent the current job from printing, e.g. 'toner-low'.

~~Note:~~If a Printer can interpret one or more jobs while marking a job and receives a job (whose required resources are ready) while in this state, such a received job MAY transit into the 'processing' state along with the job that is being marked, if any. Issue 31

If a Printer controls more than one output device, the above definition implies that a Printer is 'processing' if at least one output device is processing, and none is idle.

'5' 'stopped': If a Printer receives a job (whose required resources are ready) while in this state, such a job MUST transit into the 'pending' state immediately. Such a job MUST transit into the 'processing' state only after some human fixes the problem that stopped the printer and after jobs ahead of it complete processing. ~~If supported, the~~Issue 30 The "printer-state-reasons" attribute MUST contain at least one reason, e.g. 'media-jam', which prevents it from either processing the current job or transitioning a 'pending' job to the 'processing' state.

If a Printer can interpret one or more jobs while marking a job and receives a job (whose required resources are ready) while in this state, such a submitted job MAY transit into the 'processing' state in order to be interpreted even while the Printer is in the 'stopped' state. However, before such a job can be completed, a human needs to fix the problem. Issue 31

~~Note: if~~If a Printer controls more than one output device, the above definition implies that a Printer is 'stopped' only if all output devices are stopped.

~~Also,~~Note: it is tempting to define 'stopped' as when a sufficient number of output devices are stopped and leave it to an implementation to define the sufficient number. But such a rule complicates the definition of 'stopped' and 'processing'. For example, with this alternate definition of 'stopped', a job can move from 'pending' to 'processing' without human intervention, even though the Printer is stopped.

4.4.12 printer-state-reasons (1setOf type2 keyword)

This REQUIRED Printer attribute supplies additional detail about the device's state. Issue 30

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 may 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.
- '-warning': This suffix indicates that the reason is a "warning". An implementation may 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 may be of lower quality.
- '-error': This suffix indicates that the reason is an "error". An implementation MUST include all errors. If this attribute contains one or more errors, printer MUST be in the stopped state.

If the implementation does not add any one of the three suffixes, all parties MUST assume that the reason is an "error".

If a Printer object 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 object as a whole MAY appear as a warning in the Printer's "printer-state-reasons attribute". If the "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:

'other': The device has detected an error other than one listed in this document.

'none': There are not 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.

'media-needed': A tray has run out of media.

'media-jam': The device has a media jam.

'moving-to-paused': Someone has paused the Printer object using the Pause-Printer operation (see section 3.2.7) or other means, but the device(s) are taking an appreciable time to stop. 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.

'paused': Someone has paused the Printer object using the Pause-Printer operation (see section 3.2.7) or other means and the Printer object'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.

'shutdown': Someone has removed a Printer object from service, and the device may be powered down or physically removed. In this state, a Printer object MUST NOT produce printed output, and unless the Printer object is realized by a print server that is still active, the Printer object MUST perform no other operations requested by a client, including returning this value. If a Printer object had been printing a job when it was shutdown, the Printer NEED NOT resume printing that job when the Printer is no longer shutdown. If the Printer resumes printing such a job, it may leave evidence in the printed output of such a shutdown, e.g. the part printed before the shutdown may be printed a second time after the shutdown.

'connecting-to-device': The Printer object 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).

'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.

'stopping': The Printer object is in the process of stopping the device and will be stopped in a while. When the device is stopped, the Printer object will change the Printer object'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.

'stopped-partly': When a Printer object 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.

'toner-low': The device is low on toner.
~~'marker-supply-low':~~'toner-empty': The device is ~~low on marker supply (ink, paint, etc.).~~out of toner.
'spool-area-full': The limit of persistent storage allocated for spooling has been reached.
'cover-open': One or more covers on the device are open.
'interlock-open': One or more interlock devices on the printer are unlocked.
'door-open': One or more doors on the device are open.
'input-tray-missing': One or more input trays are not in the device.
'media-low': At least one input tray is low on media.
'media-empty': At least one input tray is empty.
'output-tray-missing': One or more output trays are not in the device
'output-area-almost-full': One or more output area is almost full (e.g. tray, stacker, collator).
'output-area-full': One or more output area is full. (e.g. tray, stacker, collator)
'marker-supply-low': The device is low on at least one marker supply. (e.g. toner, ink, ribbon)
'marker-supply-empty: The device is out of 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.
'fuser-over-temp': The fuser temperature is above normal.
'fuser-under-temp': The fuser temperature is below normal.
'opc-near-eol': The optical photo conductor is near end of life.
'opc-life-over': The optical photo conductor is no longer functioning.
'developer-low': The device is low on developer.
'developer-empty: The device is out of developer.
'interpreter-resource-unavailable': An interpreter resource is unavailable (i.e. font, form)

4.4.13 printer-state-message (text(MAX))

This Printer attribute specifies the additional information about the printer state and printer state reasons in human readable text. If the Printer object supports this attribute, the Printer object 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 3.1.4.1).

4.4.14 ipp-versions-supported (1setOf type2 keyword) Issue 36

This REQUIRED attribute identifies the IPP protocol versions that this Printer supports, including minor versions, i.e., the values of the "version-number" parameter that it will accept in requests and return in responses. If an IPP Printer receives a request with the "version-number" parameter set to a (two-octet binary) value that does not correspond to one of the values of this (US-ASCII) keyword, it MUST reject the request and return the 'server-error-version-not-supported' status code. See Section 3.1.8.

The following standard keyword values are defined:

'1.0': Version 1.0 as specified in RFC 2566 [RFC2566] and RFC 2565 [RFC2565] including any extensions registered according to Section 1.1 and any extension defined in this version or any future version of this document following the rules when the "version-number" parameter is '1.0', if any. For an example of such a '1.0' rule, see section 4.3.12.

'1.1': Version 1.1 as specified in this document and [IPP-PRO] including any extensions registered according to Section 1.1 or defined in any future version of this document following the rules when the "version-number" parameter is '1.1', if any.

4.4.15 operations-supported (1setOf type2 enum)

This REQUIRED Printer attribute specifies the set of supported operations for this Printer object and contained Job objects.

~~All~~Note: This attribute is encoded as any other enum attribute syntax according to [IPP-PRO] as 32-bits. However, all 32-bit enum values for this attribute MUST NOT exceed 0x00008FFF, since these same values are also passed in two octets in the "operation-id" parameter (see section 3.1.1) in each Protocol request with the two high order octets omitted in order to indicate the operation being performed [IPP-PRO].

The following standard enum and "operation-id" (see section 3.1.2) values are defined:

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
~~0x000C-0x3FFF~~0x00013-0x3FFF	reserved for future operations
0x4000-0x8FFF	reserved for private extensions

~~This~~The reserved block for private extensions allows for~~certain~~ vendors to implement private extensions that are guaranteed to not conflict with future registered extensions. However, there is no guarantee that two or more private extensions will not conflict.

4.4.16 multiple-document-jobs-supported (boolean) Issue 34

This Printer attribute indicates whether or not the Printer supports more than one document per job, i.e., more than one Send-Document or Send-Data operation with document data. If the Printer supports the Create-Job and Send-Document operations (see section 3.2.4 and 3.3.1), it MUST support this attribute.

4.4.17 charset-configured (charset)

This REQUIRED Printer attribute identifies the charset that the Printer object has been configured to represent 'text' and 'name' Printer attributes that are set by the operator, system 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 object's "charset-configured" attribute MUST also be among the values of the Printer object's "charset-supported" attribute.

4.4.18 charset-supported (1setOf charset)

This REQUIRED Printer attribute identifies the set of charsets that the Printer and contained Job objects support in attributes with attribute syntax 'text' and 'name'. At least the value 'utf-8' MUST be present, since IPP objects MUST support the UTF-8 ~~[RFC2044]~~[RFC2279] charset. If a Printer object supports a charset, it means that for all attributes of syntax 'text' and 'name' the IPP object MUST (1) accept the charset in requests and return the charset in responses as needed.

If more charsets than UTF-8 are supported, the IPP object MUST perform charset conversion between the charsets as described in Section 3.1.4.2.

4.4.19 natural-language-configured (naturalLanguage)

This REQUIRED Printer attribute identifies the natural language that the Printer object has been configured to represent 'text' and 'name' Printer attributes that are set by the operator, system 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 object 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 3.1.4.1 for the specification of the OPTIONAL multiple natural language support. Therefore, the value of the Printer object's "natural-language-configured" attribute MUST also be among the values of the Printer object's "generated-natural-language-supported" attribute.

4.4.20 generated-natural-language-supported (1setOf naturalLanguage)

This REQUIRED Printer attribute identifies the natural language(s) that the Printer object and contained Job objects support in attributes with attribute syntax 'text' and 'name'. The natural language(s) supported depends on implementation and/or configuration. Unlike charsets, IPP objects MUST accept requests with any natural language or any Natural Language Override whether the natural language is supported or not.

If a Printer object supports a natural language, it means that for any of the attributes for which the Printer or Job object generates messages, i.e., for the "job-state-message" and "printer-state-message" attributes and Operation Messages (see Section 3.1.5) in operation responses, the Printer and Job objects MUST be able to generate messages in any of the Printer's supported natural languages. See section 3.1.4 for the specification of 'text' and 'name' attributes in operation requests and responses.

Note: A Printer object that supports multiple natural languages, often has separate catalogs of messages, one for each natural language supported.

4.4.21 document-format-default (mimeMediaType)

This REQUIRED Printer attribute identifies the document format that the Printer object 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 types). For further details see the description of the 'mimeMediaType' attribute syntax in Section 4.1.9.

4.4.22 document-format-supported (1setOf mimeMediaType)

This REQUIRED Printer attribute identifies the set of document formats that the Printer object and contained Job objects can support. For further details see the description of the 'mimeMediaType' attribute syntax in Section 4.1.9.

4.4.23 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 object is currently rejecting any jobs submitted to it. In this case, the Printer object returns the 'server-error-not-accepting-jobs' status code.

Note: 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 may cause the Printer to reject jobs when the "printer-state" is 'idle' or it may cause the Printer object to accepts jobs when the "printer-state" is 'stopped'.

4.4.24 queued-job-count (integer(0:MAX))

This ~~RECOMMENDED~~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 object. Issue 29

4.4.25 printer-message-from-operator (text(127))

This Printer attribute provides a message from an operator, system 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.

4.4.26 color-supported (boolean)

This 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 (none are external IPP attributes in ~~IPP/1.0).~~IPP/1.1).

Note: end-users are able to determine the nature and details of the color support by querying the "printer-more-info-manufacturer" Printer attribute.

4.4.27 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 operation. If a Printer object supports these optional operations, it MUST support the "reference-uri-schemes-supported" Printer attribute with at least the following schemed URI value:

'ftp': The Printer object will use an FTP 'get' operation as defined in RFC 2228 [RFC2228] using FTP URLs as defined by [RFC2396] and[RFC2316].

The Printer object MAY OPTIONALLY support other URI schemes (see section 4.1.6).

4.4.28 pdl-override-supported (type2 keyword)

This REQUIRED Printer attribute expresses the ability for a particular Printer implementation to either attempt to override document data instructions with IPP attributes or not.

This attribute takes on the following values:

- 'attempted': This value indicates that the Printer object 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 object makes no attempt to make the IPP attribute values take precedence over embedded instructions in the document data.

Section 15 contains a full description of how this attribute interacts with and affects other IPP attributes, especially the "ipp-attribute-fidelity" attribute.

4.4.29 printer-up-time (integer(1:MAX))

This REQUIRED Printer attribute indicates the amount of time (in seconds) that this Printer instance~~of this Printer implementation~~ has been up and running. ~~This value is used to populate the Job attributes~~

~~"time-at-creation", "time-at-processing", and "time-at-completed". These time values are all measured in seconds and all have meaning only relative to this attribute, "printer-up-time".~~ The value is a monotonically increasing value starting from 1 when the Printer object is started-up (initialized, booted, etc.).

This value or the value of "printer-current-time" is used to populate the Event Time Job Description attributes "time-at-creation", "time-at-processing", and "time-at-completed", depending on implementation (see Section 4.3.12).

~~If the Printer object 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 the first case, the Printer SHOULD not tweak any existing related Job attributes ("time-at-creation", "time-at-processing", and "time-at-completed"). In the second case, the Printer object SHOULD reset those attributes to 0. If a client queries a time-related Job attribute and finds the value to be 0, the client MUST assume that the Job was submitted in some life other t~~han the Printer's current life.

If the Printer object software ceases running, and restarts without knowing the last value for "printer-up-time", the implementation MUST reset this value to 1. However, if the device or devices that the Printer object is representing are restarted or power cycled, the Printer object MAY continue counting this value or MAY reset this value to 1 depending on implementation. If this value is reset and the implementation has persistent jobs and the Event Time Job Description Attributes are represented using the 'integer' form (instead of the 'dateTime' form), they MUST be reset according to Section 4.3.12. Issue 17

4.4.30 printer-current-time (dateTime)

This Printer attribute indicates the current~~absolute~~ wall-clock time. ~~If an implementation supports thisattribute, then a client could calculate the absolute wall-clock time each Job's~~This value or the value of "printer-uptime-time" is used to populate the Job attributes "time-at-creation", "time-at-processing", and ~~"time-at-completed" attributes by using both "printer-up-time" and this attribute, "printer-current-time".~~ "time-at-completed", depending on implementation (see Section 4.3.12).

~~If an implementation does not support this attribute, a client can only calculate the relative time of certain events based on the REQUIRED "printer-up-time" attribute.~~

The date and time is obtained on a "best efforts basis" and does not have to be that precise in order to work in practice. 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 [IPP-IIG] for examples. If an implementation supports this attribute and the implementation knows that it has not yet been set to a correct value, then the implementation MUST return the value of this attribute using the out-of-band 'no-value' meaning not configured. See the beginning of section 4.1. Issue 17

The time zone of this attribute NEED NOT be the time zone used by people located near the Printer object or device. The client MUST NOT expect that 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. Issue 17

The client SHOULD display any dateTime attributes to the user in client 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. Issue 17

4.4.31 multiple-operation-time-out (integer(1:MAX))

This Printer attributes identifies the minimum time (in seconds) that the Printer object waits for additional Send-Document or Send-URI operations to follow a still-open multi-document Job object before taking any recovery actions, such as the ones indicated in section 3.3.1~~.~~

. If the Printer object supports the Create-Job and Send-Document operations (see section 3.2.4 and 3.3.1), it MUST support this attribute.

It is RECOMMENDED that vendors supply a value for this attribute that is between 60 and 240 seconds. An implementation MAY allow a system administrator to set this attribute (by means outside this IPP/1.1 document). If so, the system administrator MAY be able to set values outside this range.

4.4.32 compression-supported (1setOf type3 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, Send-Document, and Send-URI requests. Issue 28

Standard values are :

'none': no compression is used.
'deflate': ZIP public domain inflate/deflate) compression technology
'gzip' GNU zip compression technology described in RFC 1952 [RFC1952].
'compress': UNIX compression technology

4.4.33 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 attributes in create requests. The corresponding job description attribute "job-k-octets" is defined in section 4.3.15.1.

4.4.34 job-impressions-supported (rangeOfInteger(0:MAX))

This 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 attributes in create requests. The corresponding job description attribute "job-impressions" is defined in section 4.3.15.2.

4.4.35 job-media-sheets-supported (rangeOfInteger(0: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 attributes in create requests. The corresponding Job attribute "job-media-sheets" is defined in section 4.3.15.3.

4.4.36 pages-per-minute (integer(0:MAX))

This Printer attributes specifies the nominal number of pages per minute to the nearest whole number which may 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 device.

A value of 0 indicates a device that takes more than two minutes to process a page.

4.4.37 pages-per-minute-color (integer(0:MAX))

This Printer attributes specifies the nominal number of pages per minute to the nearest whole number which may be generated by this printer when printing color (e.g., simplex, color). For purposes of this attribute, "color" means the same as 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.

Note: 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.

Black and white only printers 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.

Note: The values of these two 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 3.2.5.1 Get-Printer-Attributes.

5. Conformance

This section describes conformance issues and requirements. This document introduces model entities such as objects, operations, attributes, attribute syntaxes, and attribute values. These conformance sections describe the conformance requirements which apply to these model entities.

5.1 Client Conformance Requirements

This section describes the conformance requirements for a client (see section 2.1), whether it be:

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

2. a component of a print server that communicates (using IPP operations) with either an output device or another "downstream" print server. Issue 4 and Issue 5

A conforming client MUST support 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 of the Model document as specified in Sections 3 and 4. A conforming client MAY supply any registered extensions and/or private extensions in an operation request, as long as they meet the requirements in Section 1.1.

Otherwise, there are no conformance requirements placed on the user interfaces provided by IPP clients or their applications. For example, one application might not allow an end user to submit multiple documents per job, while another does. One application might first query a Printer object in order to supply a graphical user interface (GUI) dialogue box with supported and default values whereas a different implementation might not.

When sending a request, an IPP client NEED NOT supply any attributes that are indicated as OPTIONALLY supplied by the client.

A client MUST be able to accept any of the attribute syntaxes defined in Section 4.1, including their full range, that may be returned to it in a response from a Printer object. 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 ~~query~~ response ~~may~~MAY contain attribute groups, attributes, and values that the client does not expect. Therefore, a client implementation MUST gracefully handle such responses and not refuse to inter-operate with a conforming Printer that is returning~~extended~~ registered or private ~~attributes and/or~~extensions, including attribute groups, attributes, and attribute values that conform to Section 1.1. Clients may choose to ignore any parameters, attributes, or values that they do not understand. Issue 25 and Issue 26

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 [IPP-PRO] for more details. Issue 4 and Issue 5

A client MUST/SHOULD [which is to be determined in consultation with the Area Director] support Client Authentication as defined in the IPP/1.1 Encoding and Transport document [IPP-PRO]. A client SHOULD support Operation Privacy and Server Authentication as defined in the IPP/1.1 Encoding and Transport document [IPP-PRO]. See also [IPP-MOD] section 8. Issue 32

5.2 IPP Object Conformance Requirements

This section specifies the conformance requirements for conforming implementations with respect to objects, operations, and attributes whether they be (1) IPP objects that accept IPP requests and control one or more devices or are embedded in a single device or (2) servers that accept IPP requests and forward them to networked ~~attributes.~~devices (using IPP or other protocol).

5.2.1 Objects

Conforming implementations MUST implement all of the model objects as defined in this specification in the indicated sections:

Section 2.1 - Printer Object
Section 2.2 - Job Object

whether they are (embedded) software that controls a device or are part of a print server that accepts IPP operation requests and, in turn, sends operation requests using (the IPP or other) protocol to one or more networked device(s). See sections 2.1 and 2.2.

5.2.2 Operations

Conforming IPP object implementations MUST implement all of the REQUIRED model operations, including REQUIRED responses, as defined in this specification in the indicated sections:

For a Printer object:

Print-Job (section 3.2.1)	REQUIRED
Print-URI (section 3.2.2)	OPTIONAL
Validate-Job (section 3.2.3)	REQUIRED
Create-Job (section 3.2.4)	OPTIONAL
Get-Printer-Attributes (section 3.2.5)	REQUIRED
Get-Jobs (section 3.2.6)	REQUIRED
Pause-Printer (section 3.2.7)	OPTIONAL
Resume-Printer (section 3.2.8)	OPTIONAL
Purge-Jobs (section 3.2.9)	OPTIONAL

For a Job object:

Send-Document (section 3.3.1)	OPTIONAL
Send-URI (section 3.3.2)	OPTIONAL
Cancel-Job (section 3.3.3)	REQUIRED
Get-Job-Attributes (section 3.3.4)	REQUIRED
Hold-Job (section 3.3.5)	OPTIONAL
Release-Job (section 3.3.6)	OPTIONAL
Restart-Job (section 3.3.7)	OPTIONAL

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 attribute 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 attributes groups, attributes name and attribute values that are extensions to this standard. The additional attribute groups MAY occur in any order. Issue 26

The following section on object attributes specifies the support required for object attributes.

5.2.3 IPP Object Attributes

Conforming IPP objects MUST support all of the REQUIRED object attributes, as defined in this specification 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 5.2.4. 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.

5.2.4 Versions

Clients MUST support version 1.1 and SHOULD also support version 1.0. IPP objects MUST support version 1.1 and SHOULD also support version 1.0. See section 3.1.8. ISSUE 36

5.2.5 Extensions

A conforming IPP object MAY support registered extensions and private extensions, as long as they meet the requirements specified in Section 1.1.

For each attribute included in an operation response, a conforming IPP object MUST return a value whose type and value syntax conforms to the requirement of the Model document as specified in Sections 3 and 4.

5.2.6 Attribute Syntaxes

An IPP object MUST be able to accept any of the attribute syntaxes defined in Section 4.1, including their full range, in any operation in which a client may supply attributes or the system administrator may configure attributes (by means outside the scope of ~~IPP/1.0).~~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 syntax specified in Section 4.1, including their full range if supplied previously by a client.

5.2.7 Security Issue 32

An IPP Printer implementation MUST/SHOULD [which is to be determined in consultation with the Area Director] contain support for Client Authentication as defined in the IPP/1.1 Encoding and Transport document [IPP-PRO]. A Printer implementation MAY allow an administrator to configure the Printer so that all, some, or none of the users are authenticated. See also [IPP-MOD] section 8.

An IPP Printer implementation SHOULD contain support for Operation Privacy and Server Authentication as defined in the IPP/1.1 Encoding and Transport document [IPP-PRO]. A Printer implementation MAY allow an administrator to configure the degree of support for Operation Privacy and Server Authentication. See also [IPP-MOD] section 8.

5.3 Charset and Natural Language Requirements

All clients and IPP objects MUST support the 'utf-8' charset as defined in section 4.1.7.

IPP objects MUST be able to accept any client request which 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 3.1.4). That is, the IPP object that supports a natural language NEED NOT be a general purpose translator of any arbitrary 'text' or 'name' value supplied by the client into that natural language. However, the object MUST be able to translate (automatically generate) any of its own attribute values and messages into that natural language.

~~5.4Security Conformance Requirements~~

~~Conforming IPP Printer objects MAY support Secure Socket Layer Version 3 (SSL3) [SSL] access, support access without SSL3 or support both means of access.~~

~~Conforming IPP clients SHOULD support SSL3 access and non-SSL3 access. Note: This client requirement to support both means that conforming IPP clients will be able to inter-operate with any IPP Printer object.~~

~~For a detailed discussion of security considerations and the IPP application security profile required for SSL3 support, see section 8.~~

6. IANA Considerations (registered and private extensions)

This section describes how IPP can be extended to allow the following registered and private extensions to IPP:

1. keyword attribute values
2. enum attribute values
3. attributes
4. attribute syntaxes
5. operations
6. attribute groups
7. status codes

Extensions registered for use with ~~IPP/1.0~~IPP/1.1 are OPTIONAL for client and IPP object conformance to the ~~IPP/1.0~~IPP/1.1 Model specification.

These extension procedures are aligned with the guidelines as set forth by the IESG [IANA-CON]. Section 11 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 Section 11. IPP/1.1 may also be extended by an appropriate RFC that specifies any of the above extensions.

6.1 Typed 'keyword' and 'enum' Extensions

IPP allows for 'keyword' and 'enum' extensions (see sections 4.1.2.3 and 4.1.4). This document uses prefixes to the 'keyword' and 'enum' basic attribute syntax type in order to communicate extra information to the reader through its name. This extra information is not represented in the protocol because it is unimportant to a client or Printer object. The list below describes the prefixes and their meaning.

"type1": The IPP specification must be revised to add a new keyword or a new enum. No private keywords or enums are allowed.

"type2": Implementers can, at any time, add new keyword or enum values by proposing the complete specification to IANA:

iana@iana.org

IANA will forward the registration proposal to the IPP Designated Expert who will review the proposal with a mailing list that the Designated Expert keeps for this purpose. Initially, that list will be the mailing list used by the IPP WG:

ipp@pwg.org

even after the IPP WG is disbanded as permitted by [IANA-CON]. The IPP Designated Expert is appointed by the IESG Area Director responsible for IPP, according to [IANA-CON].

When a type2 keyword or enum is approved, the IPP Designated Expert becomes the point of contact for any future maintenance that might be required for that registration.

"type3": Implementers can, at any time, add new keyword and enum values by submitting the complete specification to IANA as for type2 who will forward the proposal to the IPP Designated Expert. While no additional technical review is required, the IPP Designated Expert may, at his/her discretion, forward the proposal to the same mailing list as for type2 registrations for advice and comment.

When a type3 keyword or enum is approved by the IPP Designated Expert, the original proposer becomes the point of contact for any future maintenance that might be required for that registration.

For type2 and type3 keywords, the proposer includes the name of the keyword in the registration proposal and the name is part of the technical review.

After type2 and type3 enums specifications are approved, the IPP Designated Expert in consultation with IANA assigns the next available enum number for each enum value.

IANA will publish approved type2 and type3 keyword and enum attributes value registration specifications in:

ftp.isi.edu/iana/assignments/ipp/attribute-values/xxx/yyy.txt

where xxx is the attribute name that specifies the initial values and yyy.txt is a descriptive file name that contains one or more enums or keywords approved at the same time. For example, if several additional enums for stapling are approved for use with the "finishings" attribute (and "finishings-default" and "finishings-supported" attributes), IANA will publish the additional values in the file:

ftp.isi.edu/iana/assignments/ipp/attribute-values/finishings/stapling.txt~~.~~

Note: Some attributes are defined to be: 'type3 keywords' | 'name' which allows for attribute values to be extended by a site administrator with administrator defined names. Such names are not registered with IANA.

By definition, each of the three types above assert some sort of registry or review process in order for extensions to be considered valid. Each higher numbered level (1, 2, 3) tends to be decreasingly less stringent than the previous level. Therefore, any typeN value MAY be registered using a process for some typeM where M is less than N, however such registration is NOT REQUIRED. For example, a type3 value MAY be registered in a type 1 manner (by being included in a future version of an IPP specification), however, it is NOT REQUIRED.

This specification defines keyword and enum values for all of the above types, including type3 keywords.

For private (unregistered) keyword extensions, implementers SHOULD use keywords with a suitable distinguishing prefix, such as "xxx-" where xxx is the (lowercase) fully qualified company name registered with IANA for use in domain names [RFC1035]. For example, if the company XYZ Corp. had obtained the domain name "XYZ.com", then a private keyword 'abc' would be: 'xyz.com-abc'.

Note: RFC 1035 [RFC1035] indicates that while upper and lower case letters are allowed in domain names, no significance is attached to the case. That is, two names with the same spelling but different case are to be treated as if identical. Also, the labels in a domain name must follow the rules for ARPANET host names: They must start with a letter, end with a letter or digit, and have as interior characters only letters, digits, and hyphen. Labels must be 63 characters or less. Labels are separated by the "." character.

For private (unregistered) enum extension, implementers MUST use values in the reserved integer range which is 2**30 to 2**31-1.

6.2 Attribute Extensibility

Attribute names are type2 keywords. Therefore, new attributes may be registered and have the same status as attributes in this document by following the type2 extension rules. For private (unregistered) attribute extensions, implementers SHOULD use keywords with a suitable distinguishing prefix as described in Section 6.1.

IANA will publish approved attribute registration specifications as separate files:

ftp.isi.edu/iana/assignments/ipp/attributes/xxx-yyy.txt

where "xxx-yyy" is the new attribute name.

If a new Printer object 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 3.2.5.1)."

If the specification does not, then its value in the Get-Printer-Attributes response MUST NOT depend on the "document-format" 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.

6.3 Attribute Syntax Extensibility

Attribute syntaxes are like type2 enums. Therefore, new attribute syntaxes may be registered and have the same status as attribute syntaxes in this document by following the type2 extension rules described in Section 6.1. The value codes that identify each of the attribute syntaxes are assigned in the ~~Encoding and Transport~~"Encoding and Transport" specification [IPP-PRO], including a designated range for private, experimental use.

For attribute syntaxes, the IPP Designated Expert in consultation with IANA assigns the next attribute syntax code in the appropriate range as specified in [IPP-PRO]. IANA will publish approved attribute syntax registration specifications as separate files:

ftp.isi.edu/iana/assignments/ipp/attribute-syntaxes/xxx-yyy.txt

where 'xxx-yyy' is the new attribute syntax name.

6.4 Operation Extensibility

Operations may also be registered following the type2 procedures described in Section 6.1, though major new operations will usually be done by a new standards track RFC that augments this document. For private (unregistered) operation extensions, implementers MUST use the range for the "operation-id" in requests specified in Section 4.4.15 "operations-supported" Printer attribute.

For operations, the IPP Designated Expert in consultation with IANA assigns the next operation-id code as specified in Section 4.4.15. IANA will publish approved operation registration specifications as separate files:

ftp.isi.edu/iana/assignments/ipp/operations/Xxx-Yyy.txt

where "Xxx-Yyy" is the new operation name.

6.5 Attribute Groups

Attribute groups passed in requests and responses may be registered following the type2 procedures described in Section 6.1. The tags that identify each of the attribute groups are assigned in [IPP-PRO].

For attribute groups, the IPP Designated Expert in consultation with IANA assigns the next attribute group tag code in the appropriate range as specified in [IPP-PRO]. IANA will publish approved attribute group registration specifications as separate files:

ftp.isi.edu/iana/assignments/ipp/attribute-group-tags/xxx-yyy-tag.txt where ’xxx-yyy-tag’ is the new attribute group tag name.

6.6 Status Code Extensibility

Operation status codes may also be registered following the type2 procedures described in Section 6.1. The values for status codes are allocated in ranges as specified in Section 14 for each status code class:

"informational" - Request received, continuing process
"successful" - The action was successfully received, understood, and accepted
"redirection" - Further action must be 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 private (unregistered) operation status code extensions, implementers MUST use the top of each range as specified in Section 13.

For operation status codes, the IPP Designated Expert in consultation with IANA assigns the next status code in the appropriate class range as specified in Section 13. IANA will publish approved status code registration specifications as separate files:

ftp.isi.edu/iana/assignments/ipp/status-codes/xxx-yyy.txt

where "xxx-yyy" is the new operation status code keyword.

6.7 Registration of MIME types/sub-types for document-formats

The "document-format" attribute’s syntax is ’mimeMediaType’. This means that valid values are Internet Media Types (see Section 4.1.9). RFC 2045 [RFC2045] defines the syntax for valid Internet media types. IANA is the registry for all Internet media types.

6.8 Registration of charsets for use in ’charset’ attribute values

The "attributes-charset" attribute’s syntax is ’charset’. This means that valid values are charsets names. When a charset in the IANA registry has more than one name (alias), the name labeled as "(preferred MIME name)", if present, MUST be used (see Section 4.1.7). IANA is the registry for charsets following the procedures of [RFC2278].

7. Internationalization Considerations

Some of the attributes have values that are text strings and names which are intended for human understanding rather than machine understanding (see the ’text’ and ’name’ attribute syntaxes in Sections 4.1.1 and 4.1.2).

In each operation request, the client

- identifies the charset and natural language of the request which 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 3.1.4.1).

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' technique described section 4.1.1.2 and 4.1.2.2 respectively.

All IPP objects MUST support the UTF-8 ~~[RFC2044]~~[RFC2279] 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 3.1.4.2. 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 3.1.4.1).

For Printers that support multiple charsets and/or multiple natural languages in 'text' and 'name' attributes, different jobs may 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' mechanism to identify the differing natural languages with each job attribute returned.

The Printer object also has configured charset and natural language attributes. The client can query the Printer object to determine the list of charsets and natural languages supported by the Printer object and what the Printer object'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" attributed 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 which 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. Just because a Printer object is currently configured to support 'en-us' natural language does not mean that the Printer object should reject a job if the client supplies a job name that is in 'fr-ca'.

The "natural-language-configured" attribute identifies the one supported natural language for generated messages which is the native natural language given the current configuration of the IPP object (administrator defined).

Attributes of type 'text' and 'name' are populated from different sources. These attributes can be categorized into 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 object'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 system administrator (e.g., the Printer object's "printer-name" and "printer-location" attributes). These too can 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 object's "printer-make-and-model" attribute). These too can 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 object's "job-message-from-operator" attribute). These too can 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 object's "job-state-message" attribute, the Printer object'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 1.1) are:

Attributes		Source
Operation Attributes:		
	job-name (name)	client
	document-name (name)	client
	requesting-user-name (name)	client
	status-message	Job or Printer object
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 object
	job-originating-user-name (name)	Printer object
	job-state-message (text)	Job or Printer object
	output-device-assigned (name(127))	administrator
	job-message-from-operator (text(127))	operator
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 object
	printer-message-from-operator (text(127))	operator

8. Security Considerations

~~Some IPP objects MAY be deployed over protocol stacks that support Secure Socket Layer Version 3 (SSL3) [SSL]. Note: SSL3 is not an IETF standards track specification. Other IPP objects MAY be deployed over protocol stacks that do not support SSL3. Some IPP objects MAY be deployed over both types of protocol stacks. Those IPP objects that support SSL3, are capable of supporting mutual authentication as well as privacy of messages via multiple encryption schemes. An important point about security related information for SSL3 access to an IPP object, is that the security-related parameters (authentication, encryption keys, etc.) are "out-of-band" to the actual IPP protocol.~~

~~An IPP object that does not support SSL3 MAY elect to support a transport layer that provides other security mechanisms. For example, in a mapping of IPP over HTTP/1.1 [IPP-PRO], if the IPP object does not support SSL3, HTTP still allows for client authentication using Digest Access Authentication (DAA) [RFC2069].~~

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 corporation over a private network, the risks of exposing document data may be low enough that the corporation 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 may wish to protect the content of the information during transmission through the network with encryption.

Furthermore, the value of the information being printed may 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 possibly of denial-of-service attacks, but denial-of-service attacks against printing resources are not well understood and there is 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 ~~IPP/1.0,~~IPP/1.1, and must be established via some other type of administrative or access control framework. However, there are operation status codes that allow an IPP server to return information back to a client about any potential access control violations for an IPP object.

During a create operation, 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 8.3 below for more details.

Since the security levels or the specific threats that any given IPP system administrator may be concerned with cannot be anticipated, IPP 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, to none at all, and corresponding security mechanisms will be required. ~~SSL3 supports the type of negotiated levels of security required by most, if not all, potential IPP environments. IPP environments that require no security can elect to deploy IPP objects that do not utilize the optional SSL3 security mechanisms.~~

8.1 Security Scenarios

The following sections describe specific security attacks for IPP environments. Where examples are provided they should be considered illustrative of the environment and not an exhaustive set. Not all of these environments will necessarily be addressed in initial implementations of IPP.

8.1.1 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 work-group printers, or where batch applications print their output on large production printers. Although the identity of the user may be trusted in this environment, a user might want to protect the content of a document against such attacks as eavesdropping, replaying or tampering.

8.1.2 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 are also required. It will also be important in this environment to protect Printers against "spamming" and malicious document content.

8.1.3 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 3.2.2 and 3.3.2). Standard methods currently do not exist for remote entities to "assume" the credentials of a client for forwarding requests to a 3rd party. It is anticipated that Print-By-Reference will be used to access "public" documents and that sophisticated methods for authenticating "proxies" ~~will not be specified for version 1 of IPP.~~is not specified in this document.

~~1.2~~8.2 URIs ~~for SSL3 and non-SSL3 Access~~in Operation, Job, and Printer attributes

~~As described earlier, an IPP object can support SSL3 access, non-SSL3 access, or both.~~ The "printer-uri-supported" attribute contains the Printer object'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 object using one if its URIs. This duality is not needed for Job objects, since the Printer objects is the factory for Job objects, and the Printer object will generate the correct URI for new Job objects depending on the Printer object's security configuration.

8.3 URIs for each authentication mechanisms~~The "requesting-user-name" (name(MAX)) Operation Attribute~~

Each URI has an authentication mechanism associated with it. If the URI is the ith element of "printer-uri-supported", then authentication mechanism is the "i th" element of "uri-authentication-supported". For a list of possible authentication mechanisms, see section 4.4.2.

The Printer object 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 operations, the Printer initializes the value of the "job-originating-user-name" attribute to be the authenticated user. The authenticated user is this case is called the "job-owner".

If an implementation can be configured to support more than one authentication mechanism, then it MUST implement rules for determining equality of authenticated user names which have been authenticated via different authentication mechanisms. One possible policy is that identical names that are authenticated via different mechanism 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 mechanism 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 cancelled via 'digest' authentication. With the first policy, the job could not be cancelled in this way.

A client is able to determine the authentication mechanism used to create a job. It is the ith value of the Printer's "uri-authentication-supported" attribute, where i is the index of the element of the Printer's "uri-printer-supported" attribute equal to the job's "job-printer-uri" attribute.

~~Each operation MUST specify the user who is performing the operation in both of the following two ways:~~

~~1.31) via the REQUIRED "requesting-user-name" operation attribute that a client SHOULD supply in all operations. The client MUST obtain the value for this attribute from an environmental or network login name for the user, rather than allowing the user to supply any value. If the client does not supply a value for "requesting-user-name", the printer MUST assume that the client is supplying some anonymous name, such as "anonymous".~~

~~2) via an authentication mechanism of the underlying transport which may be configured to give no authentication information.~~

~~There are six cases to consider:~~

~~a) the authentication mechanism gives no information, and the client doesn't specify "requesting-user-name".~~
~~b) the authentication mechanism gives no information, but the client specifies "requesting-user-name".~~
~~c) the authentication mechanism specifies a user which has no human readable representation, and the client doesn't specify "requesting-user-name".~~
~~d) the authentication mechanism specifies a user which has no human readable representation, but the client specifies "requesting-user-name".~~
~~e) the authentication mechanism specifies a user which has a human readable representation. The Printer object ignores the "requesting-user-name".~~
~~f) the authentication mechanism specifies a user who is trusted and whose name means that the value of the "requesting-user-name", which MUST be present, is treated as the authenticated name.~~

~~Note: Case "f" is intended for a tightly coupled gateway and server to work together so that the "user" name is able to be that of the gateway client and not that of the gateway. Because most, if not all, system vendors will initially implement IPP via a gateway into their existing print system, this mechanism is necessary unless the authentication mechanism allows a gateway (client) to act on behalf of some other client.~~

~~The user-name has two forms: one that is human readable: it is held in the REQUIRED "job-originating-user-name" Job Description attribute which is set during the job creation operations. It is used for presentation only, such as returning in queries or printing on start sheets~~

~~- one for authorization: it is held in an undefined (by IPP) Job object attribute which is set by the job creation operation. It is used to authorize other operations, such as Send-Document, Send-URI, Cancel-Job, to determine the user when the "my-jobs" attribute is specified with Get-Jobs, and to limit what attributes and values to return with Get-Job-Attributes and Get-Jobs.~~

~~The human readable user name:~~

~~- is the value of the "requesting-user-name" for cases b, d and f.~~
~~- comes from the authentication mechanism for case e~~
~~- is some anonymous name, such as "anonymous" for cases a and c.~~

~~The user name used for authorization:~~

~~- is the value of the "requesting-user-name" for cases b and f.~~
~~- comes from the authentication mechanism for cases c, d and e~~
~~- is some anonymous name, such as "anonymous" for case a.~~

~~The essence of these rules for resolving conflicting sources of user-names is that a printer implementation is free to pick either source as long as it achieves consistent results. That is, if a user uses the same path for a series of requests, the requests MUST appear to come from the same user from the standpoint of both the human-readable user name and the user name for authorization. This rule MUST continue to apply even if a request could be authenticated by two or more mechanisms. It doesn't matter which of several authentication mechanisms a Printer uses as long as it achieves consistent results. If a client uses more than one authentication mechanism, it is recommended that an administrator make all credentials resolve to the same user and user-name as much as possible.~~

8.4 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 may 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 on the object.

8.5 Operations performed by operators and system administrators

For the three printer operations Pause-Printer, Resume-Printer, and Purge-Jobs (see sections 3.2.7, 3.2.8 and 3.2.9), the requesting user is intended to be an operator or administrator of the Printer object (see section 1). For operations on jobs, the requesting user is intended to be the job owner or may be an operator or administrator of the Printer object. The means for authorizing an operator or administrator of the Printer object are not specified in this document.

8.6 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, it is RECOMMENDED that such an implementation at least allow such "foreign" jobs to be queried using Get-Jobs returning "job-id" and "job-uri" as 'unknown'. Such an implementation NEED NOT 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.

It is further RECOMMENDED, that the IPP Printer generate "job-id" and "job-uri" values for such "foreign jobs", if possible, so that they may 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'. Only if the IPP client has been authenticated as an operator or administrator of the IPP Printer object, could the foreign jobs be queried by 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.

~~8.6IPP Security Application Profile for SSL3~~

~~The IPP application profile for SSL3 follows the "Secure Socket Layer" requirement as documented in the SSL3 specification [SSL]. For interoperability, the SSL3 cipher suites are:~~

~~SSL_RSA_WITH_RC4_128_MD5~~
~~SSL_RSA_WITH_3DES_EDE_CBC_SHA~~
~~SSL_RSA_WITH_DES_CBC_SHA~~
~~SSL_RSA_EXPORT_WITH_RC4_40_MD5~~
~~SSL_RSA_EXPORT_WITH_RC2_CBC_40_MD5~~

~~SSL_RSA_WITH_NULL_MD5~~

~~Client implementations MUST NOT assume any other cipher suites are supported by an IPP Printer object.~~

~~If a conforming IPP object supports SSL3, it MUST implement and support the cipher suites listed above and MAY support additional cipher suites.~~

~~A conforming IPP client SHOULD support SSL3 including the cipher suites listed above. A conforming IPP client MAY support additional cipher suites.~~

~~It is possible that due to certain government export restrictions some non-compliant versions of this extension could be deployed. Implementations wishing to inter-operate with such non-compliant versions MAY offer the SSL_RSA_EXPORT_WITH_RC4_40_MD5 and SSL_RSA_EXPORT_WITH_RC2_CBC_40_MD5 mechanisms. However, since 40 bit ciphers are~~

~~known to be vulnerable to attack by current technology, any client which actives a 40 bit cipher MUST NOT indicate to the user that the connection is completely secure from eavesdropping.~~

9. References

[ASCII]
Coded Character Set - 7-bit American Standard Code for Information Interchange (ASCII), ANSI X3.4-1986. This standard is the specification of the US-ASCII charset.

[BCP-11]
Bradner S., Hovey R., "The Organizations Involved in the IETF Standards Process", 1996/10/29 (RFC 2028)

[HTPP]
J. Barnett, K. Carter, R. DeBry, "Initial Draft - Hypertext Printing Protocol - HTPP/1.0", October 1996, ftp://ftp.pwg.org/pub/pwg/ipp/historic/htpp/overview.ps.gz

[IANA-CON]
Narte, T. and Alvestrand, H.T.: Guidelines for Writing an IANA Considerations Section in RFCs, Work in Progress, draft-iesg-iana-considerations-04.txt, May 21, 1998.

[IANA-CS]
IANA Registry of Coded Character Sets: ftp://ftp.isi.edu/in-notes/iana/assignments/character-sets

[IANA-MT]
IANA Registry of Media Types: ftp://ftp.isi.edu/in-notes/iana/assignments/media-types/

[IPP-IIG]
Hastings, T., Manros, C., "Internet Printing Protocol/1.1: draft-ietf-ipp-implementers-guide-v11-??.txt, ?? 1999, work in progress.

[IPP-IIG1.0]
Hastings, T., Manros, C., "Internet Printing Protocol/1.0: Implementer's Guide", draft-ietf-ipp-implementers-guide-01.txt, February 1999, work in progress.

~~draft-ietf-ipp-implementors-guide-00.txt, November 1998, work in progress.~~

[RFC2569]
~~[IPP LPD]~~
Herriot, R., Hastings, T., Jacobs, N., Martin, J., "Mapping between LPD and IPP Protocols", draft-ietf-ipp-lpd-ipp-map-05.txt, November 1998.

[RFC2566]
R. deBry, T. Hastings, R. Herriot, S. Isaacson, P. Powell, "Internet Printing Protocol/1.0: Model and Semantics", RFC 2566, April 1999.

[IPP-PRO]
Herriot, R., Butler, S., Moore, P., Tuner, R., "Internet Printing Protocol/1.1: Encoding and Transport", draft-ietf-ipp-protocol-v11-01.txt, May, 1999.

[RFC2565]
Herriot, R., Butler, S., Moore, P., Tuner, R., "Internet Printing Protocol/1.0: Encoding and Transport", ~~draft-ietf-ipp-pro-07.txt, November, 1998.~~RFC 2565, April 1999.

~~[IPP-RAT]~~[RFC2568]
Zilles, S., "Rationale for the Structure and Model and Protocol for the Internet Printing Protocol", draft-ietf-ipp-rat-04.txt, November, 1998.

~~[IPP-REQ]~~[RFC2567]
Wright, D., "Design Goals for an Internet Printing Protocol", draft-ietf-ipp-req-03.txt, November, 1998.

[ISO10646-1]
ISO/IEC 10646-1:1993, "Information technology -- Universal Multiple-Octet Coded Character Set (UCS) - Part 1: Architecture and Basic Multilingual Plane, JTC1/SC2."

[ISO8859-1]
ISO/IEC 8859-1:1987, "Information technology -- 8-bit One-Byte Coded Character Set - Part 1: Latin Alphabet Nr 1", 1987, JTC1/SC2.

[ISO10175]
ISO/IEC 10175 Document Printing Application (DPA), June 1996.

[LDPA]
T. Hastings, S. Isaacson, M. MacKay, C. Manros, D. Taylor, P. Zehler, "LDPA - Lightweight Document Printing Application", October 1996, ftp://ftp.pwg.org/pub/pwg/ipp/historic/ldpa/ldpa8.pdf.gz

[P1387.4]
Kirk, M. (editor), POSIX System Administration - Part 4: Printing Interfaces, POSIX 1387.4 D8, 1994.

[PSIS] Herriot, R. (editor), X/Open A Printing System Interoperability Specification (PSIS), August 1995.

[PWG]
Printer Working Group, http://www.pwg.org.

[RFC1035]
P. Mockapetris, "DOMAIN NAMES - IMPLEMENTATION AND SPECIFICATION", RFC 1035, November 1987.

[RFC1179]
McLaughlin, L. III, (editor), "Line Printer Daemon Protocol" RFC 1179, August 1990.

[RFC1759]
Smith, R., Wright, F., Hastings, T., Zilles, S., and Gyllenskog, J., "Printer MIB", RFC 1759, March 1995.

[RFC1766]
H. Alvestrand, "Tags for the Identification of Languages", RFC 1766, March 1995.

[RFC1903]
J. Case, et.al., "Textual Conventions for Version 2 of the Simple Network Management Protocol (SNMP v2)" RFC 1903, January 1996.

[RFC1952]
P. Deutsch, "GZIP file format specification version 4.3", RFC 1952, May 1996.

~~[RFC2044]~~
~~F. Yergeau, "UTF-8, a transformation format of Unicode and ISO 10646", RFC 2044, October 1996.~~

[RFC2026]
S. Bradner, "The Internet Standards Process -- Revision 3", RFC 2026, October 1996.

[RFC2045]
N. Fried, N. Borenstein, ", Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies " RFC 2045, November 1996.

[RFC2046]
Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types. N. Freed & N. Borenstein. November 1996. (Obsoletes RFC1521, RFC1522, RFC1590), RFC 2046.

[RFC2048]
N. Freed, J. Klensin & J. Postel, "Multipurpose Internet Mail Extension (MIME) Part Four: Registration Procedures". RFC 2048, November 1996.

[RFC2068]
R. Fielding, J. Gettys, J. Mogul, H. Frystyk, T. Berners-Lee, "Hypertext Transfer Protocol - HTTP/1.1", RFC 2068, January 1997

[RFC2069]
J. Franks, P. Hallam-Baker, J. Hostetler, P. Leach, A. Luotonen, E. Sink, L. Stewart, "An Extension to HTTP: Digest Access Authentication", RFC-2069, Jan 1997.

[RFC2119]
S. Bradner, "Key words for use in RFCs to Indicate Requirement Levels", RFC 2119 , March 1997

[RFC2228]
M. Horowitz, S. Lunt, "FTP Security Extensions", RFC 2228, October 1997.

[RFC2246]
T. Dierks, C. Allen, "The TLS Protocol Version 1.0", RFC 2246, January 1999.

[RFC2277]
H. Alvestrand, "IETF Policy on Character Sets and Languages" RFC 2277, January 1998.

[RFC2278]
N. Freed, J. Postel: "IANA CharSet Registration Procedures", RFC 2278, January 1998.

[RFC2279]
F. Yergeau , "UTF-8, a transformation format of ISO 10646", RFC 2279. January 1998.

[RFC2316]
S. Bellovin , "Report of the IAB Security Architecture Workshop", RFC 2316, April 1998.

[RFC2396]
Berners-Lee, T., Fielding, R., Masinter, L., "Uniform Resource Identifiers (URI): Generic Syntax", RFC 2396, August 1998.

[SSL]
Netscape, The SSL Protocol, Version 3, (Text version 3.02), November 1996.

[SWP]
P. Moore, B. Jahromi, S. Butler, "Simple Web Printing SWP/1.0", May 7, 1997, ftp://ftp.pwg.org/pub/pwg/ipp/new_PRO/swp9705.pdf

~~10.Copyright Notice~~

~~This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this paragraph are included on all such copies and derivative works. However, this document itself may not be modified in any way, such as by removing the copyright notice or references to the Internet Society or other Internet organizations, except as needed for the purpose of developing Internet standards in which case the procedures for copyrights defined in the Internet Standards process must be followed, or as required to translate it into languages other than English.~~

~~The limited permissions granted above are perpetual and will not be revoked by the Internet Society or its successors or assigns.~~

~~This document and the information contained herein is provided on an "AS IS" basis and THE INTERNET SOCIETY AND THE INTERNET ENGINEERING TASK FORCE DISCLAIMS ALL~~

~~WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.~~

10. Author's Address

Scott A. Isaacson (Editor)
Novell, Inc.
122 E 1700 S
Provo, UT 84606

Phone: 801-861-7366
Fax: 801-861-2517
e-mail: sisaacson@novell.com

Tom Hastings
Xerox Corporation
~~701 S. Aviation Blvd.~~737 Hawaii St. ESAE 231
El Segundo, CA 90245

Phone: 310-333-6413
Fax: 310-333-5514
e-mail: hastings@cp10.es.xerox.com

Robert Herriot
~~Sun Microsystems Inc.~~
~~901 San Antonio.Road, MPK-17~~
~~Palo Alto, CA 94303~~

~~Phone: 650-786-8995~~
~~Fax: 650-786-7077~~
~~e-mail: robert.herriot@eng.sun.com~~Xerox Corp.
3400 Hill View Ave, Building 1
Palo Alto, CA 94304

Phone: 650-813-7696
Fax: 650-813-6860
e-mail: robert.herriot@pahv.xerox.com

Roger deBry
~~HUC/003G~~
~~IBM Corporation~~
~~P.O. Box 1900~~

~~Boulder, CO 80301-9191~~

~~Phone: (303) 924-4080~~
~~Fax: (303) 924-9889~~
~~e-mail: debry@vnet.ibm.com~~Utah Valley State College
Orem, UT 84058

Phone: (801) 222-8000
EMail: debryro@uvsc.edu

Patrick Powell
Astart Technologies
9475 Chesapeake Dr., Suite D
San Diego, CA 95123

Phone: (619) 874-6543
Fax: (619) 279-8424
e-mail: papowell@astart.com

IPP Mailing List: ipp@pwg.org
IPP Mailing List Subscription: ipp-request@pwg.org
IPP Web Page: http://www.pwg.org/ipp/

Implementers of this specification are encouraged to join IPP Mailing List in order to participate in any discussions of clarification issues and review of registration proposals for additional attributes and values.

Other Participants:

Chuck Adams - Tektronix
Jeff Barnett - IBM
Ron Bergman - Dataproducts Corp.
Sylvan Butler - HP
Keith Carter - IBM Corporation
Jeff Copeland - QMS
Andy Davidson - Tektronix
Mabry Dozier - QMS
Lee Farrell - Canon Information Systems
Steve Gebert - IBM
Babek Jahromi - Microsoft
David Kellerman - Northlake Software
Rick Landau - Digital
Greg LeClair - Epson
Harry Lewis - IBM
Pete Loya - HP
Ray Lutz - Cognisys

Mike MacKay - Novell, Inc.
Daniel Manchala - Xerox
Carl-Uno Manros - Xerox
Jay Martin - Underscore
Larry Masinter - Xerox
Stan McConnell - Xerox
Ira McDonald - High North Inc.
Paul Moore - Microsoft
Tetsuya Morita - Ricoh
Yuichi Niwa - Ricoh
Pat Nogay - IBM
Ron Norton - Printronics
Bob Pentecost - HP
Rob Rhoads - Intel
Xavier Riley - Xerox
David Roach - Unisys
Stuart Rowley - Kyocera
Hiroyuki Sato - Canon
Bob Setterbo - Adobe
Devon Taylor - Novell, Inc.
Mike Timperman - Lexmark
Randy Turner - Sharp
Atsushi Yuki - Kyocera
Rick Yardumian - Xerox
Lloyd Young - Lexmark
Bill Wagner - DPI
Jim Walker - DAZEL
Chris Wellens - Interworking Labs
Rob Whittle - Novell, Inc.
Don Wright - Lexmark
Peter Zehler - Xerox
Steve Zilles - Adobe

11. 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 6 for:

1. type2 'keyword' attribute values
2. type3 'keyword' attribute values
3. type2 'enum' attribute values
4. type3 'enum' attribute values
5. attributes
6. attribute syntaxes
7. operations
8. status codes

11.1 Type2 keyword attribute values 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 IPP Model Section 4.1.2.3):
Name of proposer:
Address of proposer:
Email address of proposer:

Note: For type2 keywords, the Designated Expert will be the point of contact for the approved registration specification, if any maintenance of the registration specification is needed.

11.2 Type3 keyword attribute values registration

Type of registration: type3 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 IPP Model Section 4.1.2.3):
Name of proposer:
Address of proposer:
Email address of proposer:

Note: For type3 keywords, the proposer will be the point of contact for the approved registration specification, if any maintenance of the registration specification is needed.

11.3 Type2 enum attribute values 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 IPP Model Section 4.1.4):
Name of proposer:
Address of proposer:
Email address of proposer:

Note: For type2 enums, the Designated Expert will be the point of contact for the approved registration specification, if any maintenance of the registration specification is needed.

11.4 Type3 enum attribute values registration

Type of registration: type3 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 IPP Model Section 4.1.4):
Name of proposer:
Address of proposer:
Email address of proposer:

Note: For type3 enums, the proposer will be the point of contact for the approved registration specification, if any maintenance of the registration specification is needed.

11.5 Attribute registration

Type of registration: attribute
Proposed keyword name of this attribute:
Types of attribute (Operation, Job Template, Job Description, Printer Description):
Operations to be used with if the attribute is an operation attribute:
Object (Job, Printer, etc. if bound to an object):
Attribute syntax(es) (include 1setOf and range as in Section 4.2):
If attribute syntax is 'keyword' or 'enum', is it type2 or type3:
If this is a Printer attribute, MAY the value returned depend on "document-format" (See Section 6.2):
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 IPP Model Section 4.2):
Name of proposer:
Address of proposer:
Email address of proposer:

Note: For attributes, the IPP Designated Expert will be the point of contact for the approved registration specification, if any maintenance of the registration specification is needed.

11.6 Attribute Syntax registration

Type of registration: attribute syntax
Proposed name of this attribute syntax:
Type of attribute syntax (integer, octetString, character-string, see [IPP-PRO]):
Numeric value (to be assigned by the IPP Designated Expert in consultation with IANA):
Specification of this attribute (follow the style of IPP Model Section 4.1):
Name of proposer:
Address of proposer:
Email address of proposer:

Note: For attribute syntaxes, the IPP Designated Expert will be the point of contact for the approved registration specification, if any maintenance of the registration specification is needed.

11.7 Operation registration

Type of registration: operation
Proposed name of this operation:
Numeric operation-id value (to be assigned by the IPP Designated Expert in consultation with IANA):
Object Target (Job, Printer, etc. that operation is upon):
Specification of this attribute (follow the style of IPP Model Section 3):
Name of proposer:
Address of proposer:
Email address of proposer:

Note: For operations, the IPP Designated Expert will be the point of contact for the approved registration specification, if any maintenance of the registration specification is needed.

11.8 Attribute Group registration

Type of registration: attribute group
Proposed name of this attribute group:
Numeric tag according to [IPP-PRO] (to be assigned by the IPP Designated Expert in consultation with IANA):
Operation requests and group number for each operation in which the attribute group occurs:
Operation responses and group number for each operation in which the attribute group occurs:
Specification of this attribute group (follow the style of IPP Model Section 3):
Name of proposer:
Address of proposer:
Email address of proposer:

Note: For attribute groups, the IPP Designated Expert will be the point of contact for the approved registration specification, if any maintenance of the registration specification is needed.

11.9 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 may be used with:
Specification of this status code (follow the style of IPP Model Section 13 APPENDIX B: Status Codes and Suggested Status Code Messages):
Name of proposer:
Address of proposer:
Email address of proposer:

Note: For status codes, the Designated Expert will be the point of contact for the approved registration specification, if any maintenance of the registration specification is needed.

12. APPENDIX A: Terminology

This specification uses the terminology defined in this section.

12.1 Conformance Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

12.1.1 NEED NOT

This term is not included in RFC 2119. The verb "NEED NOT" indicates an action that the subject of the sentence does not have to implement in order to claim conformance to the standard. The verb "NEED NOT" is used instead of "MAY NOT" since "MAY NOT" sounds like a prohibition.

12.2 Model Terminology

12.2.1 Keyword

Keywords are used within this document as identifiers of semantic entities within the abstract model (see section 4.1.2.3). Attribute names, some attribute values, attribute syntaxes, and attribute group names are represented as keywords.

12.2.2 Attributes

An attribute is an item of information that is associated with an instance of an IPP object. 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 4 and all operation attributes are defined in section 3.

Job Template Attributes are described in section 4.2. The client optionally supplies Job Template attributes in a create request (operation requests that create Job objects). The Printer object has associated attributes which define supported and default values for the Printer.

12.2.2.1 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 header describing 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.

12.2.2.2 Attribute Group Name

Related attributes are grouped into named groups. The name of the group is a keyword. The group name may be used in place of naming all the attributes in the group explicitly. Attribute groups are defined in section 3.

12.2.2.3 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.

12.2.2.4 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~~"Encoding and Transport" document [IPP-PRO] indicates the actual "on-the-wire" encoding rules for each syntax type. Attribute syntax types are defined in section 4.1.

12.2.3 Supports

By definition, a Printer object supports an attribute only if that Printer object responds with the corresponding attribute populated with some value(s) in a response to a query for that attribute. A Printer object supports an attribute value if the value is one of the Printer object's "supported values" attributes. The device behind a Printer object may exhibit a behavior that corresponds to some IPP attribute, but if the Printer object, 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 object'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 object does not support that particular value.

A conforming implementation MUST support 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, possibly only one, 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 which 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 "finishings-supported" attribute.

1) If a Printer object is not physically capable of stapling, the "finishings-supported" attribute MUST NOT be populated with the value of 'staple'.
2) A Printer object 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 object 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 object 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 object attribute. Doing so, would enable 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 object is OPTIONAL, it is RECOMMENDED that if the device behind a Printer object is capable of realizing any feature or function that corresponds to an IPP attribute and some associated value, then that implementation 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 ~~IPP.~~this IPP/1.1 document. For administrative policy and control reasons, an administrator may 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 may be capable of a certain feature, however an administrator is specifying that access to that feature not be exposed to the end user through the IPP protocol. Also, since a Printer object may 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 object supports a value, some manual human action may 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 object.

For another example of how supported attributes function, consider a system 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 system 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 create 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 create request is rejected or the "job-sheets" value is ignored (again depending on the value of "ipp-attribute-fidelity").

12.2.4 print-stream page

A "print-stream page" is a page according to the definition of pages in the language used to express the document data.

12.2.5 impression

An "impression" is the image (possibly many print-stream pages in different configurations) imposed onto a single media page.

13. APPENDIX B: Status Codes 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. Since the status message is an OPTIONAL component of the operation response, an IPP application (i.e., a browser, GUI, print driver or gateway) is NOT REQUIRED to examine or display the status message, since it MAY not be returned to the application.

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 must be 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 codes are extensible. IPP clients are NOT REQUIRED to understand the meaning of all registered status codes, though such understanding is obviously desirable. However, 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. In such cases, IPP applications SHOULD present the OPTIONAL message (if present) to the end user since the message is likely to contain human readable information which will help to explain the unusual status. The name of the enum is the suggested status message for US English.

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" - 0x0200 to 0x02FF
"client-error" - 0x0400 to 0x04FF
"server-error" - 0x0500 to 0x05FF

The top half (128 values) of each range (0x0n40 to 0x0nFF, for n = 0 to 5) is reserved for private use within each status code class. Values 0x0600 to 0x7FFF are reserved for future assignment and MUST NOT be used.

13.1 Status Codes

Each status code is described below. Section 13.1.5.9 contains a table that indicates which status codes apply to which operations. The Implementer's Guide [IPP-IIG] describe the suggested steps for processing IPP attributes for all operations, including returning status codes.

13.1.1 Informational

This class of status code indicates a provisional response and is to be used for informational purposes only.

There are no status codes defined in ~~IPP/1.0~~IPP/1.1 for this class of status code.

13.1.2 Successful Status Codes

This class of status code indicates that the client's request was successfully received, understood, and accepted.

13.1.2.1 successful-ok (0x0000)

The request has succeeded and no request attributes were substituted or ignored. In the case of a response to a create 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.

13.1.2.2 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 MAY, but is NOT REQUIRED to, return the "requested-attributes" attribute in the Unsupported Attribute response group (with the unsupported values only). See sections 3.1.7 and 3.2.1.2.

13.1.2.3 successful-ok-conflicting-attributes (0x0002)

The request has succeeded, but some supplied attribute values conflicted with the values of other supplied attributes. These conflicting values were either (1) substituted with (supported) values or (2) the attributes were removed in order to process the job without rejecting it. Attributes or values which 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 3.1.7 and 3.2.1.2.

13.1.3 Redirection Status Codes

This class of status code indicates that further action needs to be taken to fulfill the request.

There are no status codes defined in ~~IPP/1.0~~IPP/1.1 for this class of status code.

13.1.4 Client Error Status Codes

This class of status code 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.

13.1.4.1 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

Implementer's Guide [IPP-IIG]). The IPP application SHOULD NOT repeat the request without modifications.

13.1.4.2 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.

13.1.4.3 client-error-not-authenticated (0x0402)

The request requires user authentication. The IPP client may 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 may contain relevant diagnostic information. This status codes reveals more information than "client-error-forbidden".

13.1.4.4 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 codes reveals more information than "client-error-forbidden" and "client-error-not-authenticated".

13.1.4.5 client-error-not-possible (0x0404)

This status code is used when the request is for something that can not 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.

13.1.4.6 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 and this error status code was returned in response to the Send-Document request (see section 3.3.1). 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.

13.1.4.7 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 mean time 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 can not 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 can not 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.

13.1.4.8 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.

13.1.4.9 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.

13.1.4.10 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 has 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 may pass the value onto some other system component which is not able to accept the large value. For more details, see the Implementer's Guide [IPP-IIG] .

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.

13.1.4.11 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 object. This error is returned independent of the client-supplied "ipp-attribute-fidelity". The Printer object 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 3.1.7 and 3.2.1.1. Issue 11

13.1.4.12 client-error-attributes-or-values-not-supported (0x040B)

In a create request, if the Printer object does not support one or more attributes, attribute syntaxes, or attribute values supplied in the request and the client supplied the "ipp-attributes-fidelity" operation attribute with the ’true’ value, the Printer object MUST return this status code. The Printer object MUST also return in the Unsupported Attributes Group all the attributes and/or values supplied by the client that are not supported. See section 3.1.7. Issue 11 For example, if the request indicates ’iso-a4’ media, but that media type is not supported by the Printer object. Or, if the client supplies ~~an optional~~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 MAY return the unsupported attributes as values of the "requested-attributes" in the Unsupported Attributes Group (see section 13.1.2.2).

13.1.4.13 client-error-uri-scheme-not-supported (0x040C)

The ~~type of the client supplied~~scheme of the client-supplied URI in a Print-URI or a Send-URI operation is not supported. See section 3.1.7. Issue 11

13.1.4.14 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 and any ’text’ or ’name’ attributes using the ’utf-8’ charset (see Section 3.1.4.1). See section 3.1.7. Issue 11

13.1.4.15 client-error-conflicting-attributes (0x040E)

The request is rejected because some attribute values conflicted with the values of other attributes which this specification does not permit to be substituted or ignored. The Printer object MUST also return in the Unsupported Attributes Group the conflicting attributes supplied by the client. See sections 3.1.7 and 3.2.1.2. Issue 27

13.1.4.16 client-error-compression-not-supported (0x040F) Issue 6

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 object. This error is returned independent of the client-supplied "ipp-attribute-fidelity". The Printer object 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. Issue 6 See sections 3.1.7 and 3.2.1.1. Issue 11

13.1.4.17 client-error-compression-error (0x0410) Issue 6

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". The Printer object 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 3.1.7 and 3.2.1.1.

13.1.4.18 client-error-document-format-error (0x0411) Issue 28

The IPP object is refusing to service the request because Printer encountered an error in the document data while interpreting it. This error is returned independent of the client-supplied "ipp-attribute-fidelity". The Printer object 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 3.1.7 and 3.2.1.1.

13.1.4.19 client-error-document-access-error (0x0412) Issue 35

The IPP object is refusing to service the Print-URI or Send-URI request because Printer encountered an access error while attempting to validate the accessibility or access the document data specified in the "document-uri" operation attribute. This error is returned independent of the client-supplied "ipp-attribute-fidelity". The Printer object 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 section 3.1.7.

13.1.5 Server Error Status Codes

This class of status codes 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.

13.1.5.1 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 probably some knowledgeable human intervention is required.

13.1.5.2 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 section 3.1.7. Issue 18

13.1.5.3 server-error-service-unavailable (0x0502)

The IPP object is currently unable to handle the request due to a temporary overloading or maintenance of the IPP object. The implication is that this is a temporary condition which will be alleviated after some delay. If known, the length of the delay may 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 error status codes "client-error-gone" or "client-error-not-found" could be used.

13.1.5.4 server-error-version-not-supported (0x0503)

The IPP object does not support, or refuses to support, the IPP protocol version that was ~~used in the request message.~~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 response ~~should~~SHOULD contain a ~~Message~~"status-message" attribute describing why that version is not supported and what other versions are supported by that IPP object. See section 3.1.6.

~~A conforming IPP/1.0 client MUST specify the valid version ('1.0') on each request. A conforming IPP/1.0 object MUST NOT return this status code to a conforming IPP/1.0 client. An IPP object MUST return this status code to a non-conforming IPP client. The~~The error response MUST identify in the "version-number" operation ~~attribute~~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 MUST respond with version '1.0'. If the IPP/1.1 object does not support version '1.0', then it MUST respond with this error code. Issue 36

13.1.5.5 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~~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 create 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 create request is that the printer currently has a paper jam. In many cases however, where the Printer object 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.

13.1.5.6 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 may have been cleared. Alternatively, as an implementation option, a Printer object MAY delay the response until the temporary condition is cleared so that no error is returned.

13.1.5.7 server-error-not-accepting-jobs (0x0506)

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-not-accepting-jobs" attribute to ’false’ (by means outside ~~of IPP/1.0).~~the scope of this IPP/1.1 document).

13.1.5.8 server-error-busy (0x0507)

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.

13.1.5.9 server-error-job-canceled (0x0508)

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 and job-uri 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 are returned in the response.

13.1.5.10 server-error-multiple-document-jobs-not-supported (0x0509) Issue 34

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.

13.2 Status Codes 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					
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						
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						

14. APPENDIX C: "media" keyword values

Standard keyword values are taken from several sources.

Standard values are defined (taken from DPA[ISO10175] and the Printer MIB[RFC1759]):

’default’: The default medium for the output device

’iso-a4-white’: Specifies the ISO A4 white medium

’iso-a4-colored’: Specifies the ISO A4 colored medium

’iso-a4-transparent’ Specifies the ISO A4 transparent medium

’iso-a3-white’: Specifies the ISO A3 white medium

’iso-a3-colored’: Specifies the ISO A3 colored medium

’iso-a5-white’: Specifies the ISO A5 white medium

’iso-a5-colored’: Specifies the ISO A5 colored medium

’iso-b4-white’: Specifies the ISO B4 white medium

’iso-b4-colored’: Specifies the ISO B4 colored medium

’iso-b5-white’: Specifies the ISO B5 white medium

’iso-b5-colored’: Specifies the ISO B5 colored medium

’jis-b4-white’: Specifies the JIS B4 white medium

’jis-b4-colored’: Specifies the JIS B4 colored medium

’jis-b5-white’: Specifies the JIS B5 white medium

’jis-b5-colored’: Specifies the JIS B5 colored medium

The following standard values are defined for North American media:

’na-letter-white’: Specifies the North American letter white medium

’na-letter-colored’: Specifies the North American letter colored medium

’na-letter-transparent’: Specifies the North American letter transparent medium

’na-legal-white’: Specifies the North American legal white medium

’na-legal-colored’: Specifies the North American legal colored medium

The following standard values are defined for envelopes:

’iso-b4-envelope’: Specifies the ISO B4 envelope medium

’iso-b5-envelope’: Specifies the ISO B5 envelope medium

’iso-c3-envelope’: Specifies the ISO C3 envelope medium

’iso-c4-envelope’: Specifies the ISO C4 envelope medium

’iso-c5-envelope’: Specifies the ISO C5 envelope medium

’iso-c6-envelope’: Specifies the ISO C6 envelope medium

’iso-designated-long-envelope’: Specifies the ISO Designated Long envelope medium

’na-10x13-envelope’: Specifies the North American 10x13 envelope medium

’na-9x12-envelope’: Specifies the North American 9x12 envelope medium

’monarch-envelope’: Specifies the Monarch envelope
’na-number-10-envelope’: Specifies the North American number 10 business envelope medium
’na-7x9-envelope’: Specifies the North American 7x9 inch envelope
’na-9x11-envelope’: Specifies the North American 9x11 inch envelope
’na-10x14-envelope’: Specifies the North American 10x14 inch envelope
’na-number-9-envelope’: Specifies the North American number 9 business envelope
’na-6x9-envelope’: Specifies the North American 6x9 inch envelope
’na-10x15-envelope’: Specifies the North American 10x15 inch envelope

The following standard values are defined for the less commonly used media (white-only):

’executive-white’: Specifies the white executive medium
’folio-white’: Specifies the folio white medium
’invoice-white’: Specifies the white invoice medium
’ledger-white’: Specifies the white ledger medium
’quarto-white’: Specified the white quarto medium
’iso-a0-white’: Specifies the ISO A0 white medium
’iso-a1-white’: Specifies the ISO A1 white medium
’iso-a2-white’: Specifies the ISO A2 white medium
’iso-a6-white’: Specifies the ISO A6 white medium
’iso-a7-white’: Specifies the ISO A7 white medium
’iso-a8-white’: Specifies the ISO A8 white medium
’iso-a9-white’: Specifies the ISO A9 white medium
’iso-10-white’: Specifies the ISO A10 white medium
’iso-b0-white’: Specifies the ISO B0 white medium
’iso-b1-white’: Specifies the ISO B1 white medium
’iso-b2-white’: Specifies the ISO B2 white medium
’iso-b3-white’: Specifies the ISO B3 white medium
’iso-b6-white’: Specifies the ISO B6 white medium
’iso-b7-white’: Specifies the ISO B7 white medium
’iso-b8-white’: Specifies the ISO B8 white medium
’iso-b9-white’: Specifies the ISO B9 white medium
’iso-b10-white’: Specifies the ISO B10 white medium
’jis-b0-white’: Specifies the JIS B0 white medium
’jis-b1-white’: Specifies the JIS B1 white medium
’jis-b2-white’: Specifies the JIS B2 white medium
’jis-b3-white’: Specifies the JIS B3 white medium
’jis-b6-white’: Specifies the JIS B6 white medium
’jis-b7-white’: Specifies the JIS B7 white medium
’jis-b8-white’: Specifies the JIS B8 white medium
’jis-b9-white’: Specifies the JIS B9 white medium
’jis-b10-white’: Specifies the JIS B10 white medium

The following standard values are defined for engineering media:

’a’: Specifies the engineering A size medium
’b’: Specifies the engineering B size medium
’c’: Specifies the engineering C size medium
’d’: Specifies the engineering D size medium
’e’: Specifies the engineering E size medium

The following standard values are defined for input-trays (from ISO DPA and the Printer MIB):

’top’: The top input tray in the printer.
’middle’: The middle input tray in the printer.
’bottom’: The bottom input tray in the printer.
’envelope’: The envelope input tray in the printer.
’manual’: The manual feed input tray in the printer.
’large-capacity’: The large capacity input tray in the printer.
’main’: The main input tray
’side’: The side input tray

The following standard values are defined for media sizes (from ISO DPA):

’iso-a0’: Specifies the ISO A0 size: 841 mm by 1189 mm as defined in ISO 216
’iso-a1’: Specifies the ISO A1 size: 594 mm by 841 mm as defined in ISO 216
’iso-a2’: Specifies the ISO A2 size: 420 mm by 594 mm as defined in ISO 216
’iso-a3’: Specifies the ISO A3 size: 297 mm by 420 mm as defined in ISO 216
’iso-a4’: Specifies the ISO A4 size: 210 mm by 297 mm as defined in ISO 216
’iso-a5’: Specifies the ISO A5 size: 148 mm by 210 mm as defined in ISO 216
’iso-a6’: Specifies the ISO A6 size: 105 mm by 148 mm as defined in ISO 216
’iso-a7’: Specifies the ISO A7 size: 74 mm by 105 mm as defined in ISO 216
’iso-a8’: Specifies the ISO A8 size: 52 mm by 74 mm as defined in ISO 216
’iso-a9’: Specifies the ISO A9 size: 37 mm by 52 mm as defined in ISO 216
’iso-a10’: Specifies the ISO A10 size: 26 mm by 37 mm as defined in ISO 216
’iso-b0’: Specifies the ISO B0 size: 1000 mm by 1414 mm as defined in ISO 216
’iso-b1’: Specifies the ISO B1 size: 707 mm by 1000 mm as defined in ISO 216
’iso-b2’: Specifies the ISO B2 size: 500 mm by 707 mm as defined in ISO 216
’iso-b3’: Specifies the ISO B3 size: 353 mm by 500 mm as defined in ISO 216
’iso-b4’: Specifies the ISO B4 size: 250 mm by 353 mm as defined in ISO 216
’iso-b5’: Specifies the ISO B5 size: 176 mm by 250 mm as defined in ISO 216
’iso-b6’: Specifies the ISO B6 size: 125 mm by 176 mm as defined in ISO 216
’iso-b7’: Specifies the ISO B7 size: 88 mm by 125 mm as defined in ISO 216
’iso-b8’: Specifies the ISO B8 size: 62 mm by 88 mm as defined in ISO 216
’iso-b9’: Specifies the ISO B9 size: 44 mm by 62 mm as defined in ISO 216
’iso-b10’: Specifies the ISO B10 size: 31 mm by 44 mm as defined in ISO 216
’na-letter’: Specifies the North American letter size: 8.5 inches by 11 inches
’na-legal’: Specifies the North American legal size: 8.5 inches by 14 inches
’executive’: Specifies the executive size (7.25 X 10.5 in)
’folio’: Specifies the folio size (8.5 X 13 in)

’invoice’: Specifies the invoice size (5.5 X 8.5 in)

’ledger’: Specifies the ledger size (11 X 17 in)

’quarto’: Specifies the quarto size (8.5 X 10.83 in)

’iso-c3’: Specifies the ISO C3 size: 324 mm by 458 mm as defined in ISO 269

’iso-c4’: Specifies the ISO C4 size: 229 mm by 324 mm as defined in ISO 269

’iso-c5’: Specifies the ISO C5 size: 162 mm by 229 mm as defined in ISO 269

’iso-c6’: Specifies the ISO C6 size: 114 mm by 162 mm as defined in ISO 269

’iso-designated-long’: Specifies the ISO Designated Long size: 110 mm by 220 mm as defined in ISO 269

’na-10x13-envelope’: Specifies the North American 10x13 size: 10 inches by 13 inches

’na-9x12-envelope’: Specifies the North American 9x12 size: 9 inches by 12 inches

’na-number-10-envelope’: Specifies the North American number 10 business envelope size: 4.125 inches by 9.5 inches

’na-7x9-envelope’: Specifies the North American 7x9 inch envelope size

’na-9x11-envelope’: Specifies the North American 9x11 inch envelope size

’na-10x14-envelope’: Specifies the North American 10x14 inch envelope size

’na-number-9-envelope’: Specifies the North American number 9 business envelope size

’na-6x9-envelope’: Specifies the North American 6x9 envelope size

’na-10x15-envelope’: Specifies the North American 10x15 envelope size

’monarch-envelope’: Specifies the Monarch envelope size (3.87 x 7.5 in)

’jis-b0’: Specifies the JIS B0 size: 1030mm x 1456mm

’jis-b1’: Specifies the JIS B1 size: 728mm x 1030mm

’jis-b2’: Specifies the JIS B2 size: 515mm x 728mm

’jis-b3’: Specifies the JIS B3 size: 364mm x 515mm

’jis-b4’: Specifies the JIS B4 size: 257mm x 364mm

’jis-b5’: Specifies the JIS B5 size: 182mm x 257mm

’jis-b6’: Specifies the JIS B6 size: 128mm x 182mm

’jis-b7’: Specifies the JIS B7 size: 91mm x 128mm

’jis-b8’: Specifies the JIS B8 size: 64mm x 91mm

’jis-b9’: Specifies the JIS B9 size: 45mm x 64mm

’jis-b10’: Specifies the JIS B10 size: 32mm x 45mm

15. APPENDIX D: Processing IPP Attributes

When submitting a print job to a Printer object, the IPP model allows a client to supply operation and Job Template attributes along with the document data. These Job Template attributes in the create request affect the rendering, production and finishing of the documents in the job. Similar types of instructions may also be contained in the document to be printed, that is, embedded within the print data itself. In addition, the Printer has a set of attributes that describe what rendering and finishing options which are supported by that Printer. This model, which allows for flexibility and power, also introduces the potential that at job submission time, these client-supplied attributes may conflict with either:

- what the implementation is capable of realizing (i.e., what the Printer supports), as well as
- the instructions embedded within the print data itself.

The following sections describe how these two types of conflicts are handled in the IPP model.

15.1 Fidelity

If there is a conflict between what the client requests and what a Printer object supports, the client may request one of two possible conflict handling mechanisms:

1) either reject the job since the job can not 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 to the Printer object: "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 to the Printer object: "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 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 create request, "ipp-attribute-fidelity" is a boolean operation attribute that is OPTIONALLY 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 them or substitute any supported value for unsupported values, respectively. The Printer may 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', the Printer may choose to substitute 'iso-a4' rather than a default value of 'envelope'. 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 object 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 object 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 sub-optimal result to nothing at all.
3) The End User just wants something reasonable in lieu of nothing at all.

15.2 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 were 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'. The job submission attribute should take precedence over the embedded PDL instruction. 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 objects capabilities to override instructions embedded in the PDL data stream. The value of the "pdl-override-supported" attribute is configured by means outside ~~IPP/1.0.~~the scope of this IPP/1.1 document.

This REQUIRED Printer attribute takes on the following values:

- 'attempted': This value indicates that the Printer object 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 object 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 object 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 pre-pend 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 object pre-pended. In other words, this implementation is using the same implementation semantics for the client-supplied IPP attributes as for the Printer object 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 can not 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 may fail to affect the Job processing and output when the corresponding instruction is embedded in the document data.

15.3 Using Job Template Attributes During Document Processing.

The Printer object uses some of the Job object'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 and any alternative steps that accomplishes the same effect can be used to implement this specification.

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 or not 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 specification. The formatting of the document data uses the "orientation-requested" attribute to determine how the formatted print data should be placed on a print-stream page, see section 4.2.10 for the details.

2. The document data is in the form of a print-stream in a known media type. The "page-ranges" attribute is used to select, as specified in section 4.2.7, a sub-sequence of the pages in the print-stream that are to be processed and images.

3. The input to this step is a sequence of print-stream pages. This step is controlled by the "number-up" attribute. If the value of "number-up" is N, then during the processing of the print-stream pages, each N print-stream pages are positioned, as specified in section 4.2.9, to create a single impression. If a given document does not have N more print-stream pages, then the completion of the impression is controlled by the "multiple-document-handling" attribute as described in section 4.2.4; when the value of this attribute is 'single-document' or 'single-document-new-sheet', the print-stream pages of document data from subsequent documents is used to complete the impression.

   The size(scaling), position(translation) and rotation of the print-stream pages on the impression is implementation defined. Note that during this process the print-stream pages may 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 4.2.12 and 4.2.13. In the case N=1, the impression is nearly the same as the print-stream page; the differences would only be in the size, position and rotation of the print-stream page and/or any decoration, such as a frame to the page, that is added by the implementation.

4. 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 print-stream page, as described in section 4.2.8. The orientation of the print-stream pages affects the orientation of the impression; for example, if "number-up" equals 2, then, typically, two portrait print-stream 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 4.2.4.

5. The "copies" and "multiple-document-handling" attributes are used to determine how many copies of each media instance are created and in what order. See sections 4.2.5 and 4.2.4 for the details.

6. When the correct number of copies are created, the media instances are finished according to the values of the "finishings" attribute as described in 4.2.6. Note that sometimes finishing operations may require manual intervention to perform the finishing operations on the copies, especially uncollated copies. This specification allows any or all of the processing steps to be performed automatically or manually at the discretion of the Printer object.

16. APPENDIX E: Generic Directory Schema

This section defines a generic schema for an entry in a directory service. 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. IPP 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 object 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 object.

The generic schema is a subset of IPP Printer Job Template and Printer Description attributes (sections 4.2 and 4.4). 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 Printers 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 object.

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 object, one of the RECOMMENDED directory entry attributes is the Printer object's "printer-uri-supported" attribute. The IPP client queries the "printer-uri-supported" attribute in the directory entry and then addresses the IPP Printer object using one of its URIs. The "uri-security-supported" attribute identifies the protocol (if any) used to secure a channel.

The following attributes define the generic schema for directory entries of type PRINTER:

printer-uri-supported	RECOMMENDED	Section 4.4.1
uri-authentication-supported	RECOMMENDED	Section 4.4.2
uri-security-supported	RECOMMENDED	Section 4.4.3
printer-name	RECOMMENDED	Section 4.4.4
printer-location	RECOMMENDED	Section 4.4.5
printer-info	OPTIONAL	Section 4.4.6
printer-more-info	OPTIONAL	Section 4.4.7
printer-make-and-model	RECOMMENDED	Section 4.4.9
charset-supported	OPTIONAL	Section 4.4.18
generated-natural-language-supported	OPTIONAL	Section 4.4.20
document-format-supported	RECOMMENDED	Section 4.4.22
compression-supported	RECOMMENDED	Section 4.4.32
color-supported	RECOMMENDED	Section 4.4.26
finishings-supported	OPTIONAL	Section 4.2.6
number-up-supported	OPTIONAL	Section 4.2.7
sides-supported	RECOMMENDED	Section 4.2.8
media-supported	RECOMMENDED	Section 4.2.11
printer-resolution-supported	OPTIONAL	Section 4.2.12
print-quality-supported	OPTIONAL	Section 4.2.13

ipp-versions-supported	RECOMMENDED	Section 4.4.14
multiple-document-jobs-supported	OPTIONAL	Section 4.4.16
pages-per-minute	OPTIONAL	Section 4.4.36
pages-per-minute-color	OPTIONAL	Section 4.4.37

~~APPENDIX F: Change History for the IPP Model and Semantics document~~

~~The following substantive changes and major clarifications have been made to this document from the June 30, 1998 version based on the interoperability testing that took place September 23-25 1998 and subsequent mailing list and meeting discussions. They are listed in the order of occurrence in the document. These changes are the ones that might affect implementations. Clarifications that are unlikely to affect implementations are not listed. The issue numbers refer to the IPP Issues List which is available in the following directory:~~

~~ftp://ftp.pwg.org/pub/pwg/ipp/approved-clarifications/~~

~~Section~~	~~Description~~
~~global~~	~~Replaced TLS references with SSL3 references as agreed with our Area Director on 11/12/1998.~~
~~global~~	~~Removed the indications that some of these IPP documents are informational, since the intent is now to publish all IPP/1.0 documents as informational as agreed with our Area Director on 11/12/1998.~~
~~3.1.2, 16.3.3 [now IPP-IIG]~~	~~Clarify that the IPP object SHOULD NOT validate the range of the request-id being 1 to 2**31-1, but accepts and returns any value. Clients MUST still keep in the range 1 to 2**31 though. 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 (Issue 1.36).~~
~~3.1.4.1, 14.1.4.14~~	~~Clarified that when a client submits a request in a charset that is not supported, the IPP object SHOULD return any 'text' or 'name' attributes in the 'utf-8' charset, if it returns any, since clients and IPP objects MUST support 'utf-8'. (Issue 1.19)~~
~~3.1.4.1~~	~~Clarified Section 3.1.4.1 Request Operation Attributes that a client MAY use the attribute level natural language override (text/nameWithLanguage) redundantly in a request. (Issue 1.46)~~
~~3.1.4.2~~	~~Clarified Section 3.1.4.2 Response Operation Attributes that an IPP object MAY use the attribute level natural language override (text/nameWithLanguage) redundantly in a response. (Issue 1.46)~~
~~3.1.6~~	~~Clarified section 3.1.6: If the Printer object supports the "status-message" operation attribute, it NEED NOT return a status message for the following error status codes: 'client-error-bad-request', 'client-error-charset-not-supported', 'server-error-internal-error', 'server-error-operation-not-~~

	~~supported’, and ’server-error-version-not-supported’.~~
~~3.2.1.1~~	~~Clarified that if a client is not supplying any Job Template attributes in a request, the client SHOULD omit Group 2 rather than sending an empty group. However, a Printer object MUST be able to accept an empty group. This makes [IPP-MOD] agree with [IPP-PRO]. (Issue 1.16)~~
~~3.2.1.2,~~ ~~3.2.5.2,~~ ~~3.2.6.2,~~ ~~3.3.1.2,~~ ~~3.3.3.2,~~ ~~3.3.4.2,~~	~~Clarified that if an IPP object is not returning any Unsupported Attributes in a response, the IPP object SHOULD omit Group 2 rather than sending an empty group. However, a client MUST be able to accept an empty group. This makes [IPP-MOD] agree with [IPP-PRO]. (Issue 1.17)~~
~~3.2.1.2,~~ ~~14.1.2.2,~~ ~~14.1.4.12~~	~~Clarified that an IPP object MUST treat an unsupported attribute syntax supplied in a request in the same way as an unsupported value. The IPP object MUST return the attribute, the attribute syntax, and the value in the Unsupported Attributes group. (Issue 1.26)~~
~~3.2.5.2,~~ ~~3.2.6.2,~~ ~~3.3.4.2,~~ ~~14.1.2.1,~~ ~~14.1.2.2,~~ ~~14.1.4.12~~	~~Clarified for Get-Printer-Attributes, Get-Jobs, and Get-Job-Attributes that an IPP object MUST return ’successful-ok-ignored-or-substituted-attributes’ (0x1), rather than ’successful-ok’ (0x0), when a client supplies unsupported attributes as values of the ’requested-attributes’ operation attribute. (Issue 1.24)~~ ~~Also clarified that the response NEED NOT contain the "requested-attributes" operation attribute with any supplied values (attribute keywords) that were requested by the client but are not supported by the IPP object. (Issue 1.18)~~
~~3.2.6.2~~ ~~4.1.1.2~~ ~~4.3.24~~	~~Deleted the job-level natural language override (NLO) from Section 3.2.6.2 Get-Jobs Response so that all operation responses are the same with respect to NLO. (Issue 1.47)~~
~~3.3.1~~	~~Clarified that an IPP Printer that supports the Create-Job operation MUST handle the situation when a client does not supply Send-Document or Send-URI operations within a one- to four-minute time period. Also clarified that a client MUST send documents in a multi-document job without undue or unbounded delay. (Issue 1.28)~~
~~3.3.3~~	~~Clarified that the IPP object MUST reject a Cancel-Job request if the job is in ’completed’, ’canceled’, or ’aborted’ job states. (Issue 1.12)~~
~~4.1.2.3~~	~~Added this new sub-section: it specifies that nameWithoutLanguage plus the implicit natural language matches nameWithLanguage, if the values and natural languages are the same. Also added that keyword never matches nameWithLanguage or nameWithoutLanguage. Clarified that if both have countries, that the countries SHOULD match as well. If either do not, then the country field SHOULD be ignored. (Issues 1.33 and 1.34)~~
~~4.1.5~~	~~Clarified regarding the case-insensitivity of URLs to refer only to the RFCs that define them. (Issue 1.10)~~
~~4.1.11~~	~~Clarified that ’boolean’ is not a full-sized integer. (Issue 1.38)~~
~~4.1.15~~	~~Clarified that ’resolution’ is not three full-sized integers. (Issue 1.20)~~
~~4.2.*~~	~~Clarified that standard values are keywords or enums, not names. (Issue~~

	~~1.49).~~
~~4.2.4~~	~~Added the 'single-document-new-sheet' value to Section 4.2.4 multiple-document-handling. (Issue 1.54)~~
~~4.4.18,~~ ~~4.4.19~~	~~Clarified that the "document-format-default" and "document-format-supported" Printer Description attributes are REQUIRED to agree with the table. (Issue 1.4)~~
~~4.4.21~~	~~Changed "queued-job-count" from OPTIONAL to RECOMMENDED. (Issue 1.14)~~
~~4.4.28~~	~~Clarified that the implementation supplied value for the "multiple-operation-time-out" attribute SHOULD be between 30 and 240 seconds, though the implementation MAY allow the administrator to set values, and MAY allow values outside this range. (Issue 1.28)~~
~~5.1,~~ ~~5.2.5~~	~~Clarified Client Conformance that if a client supports an attribute of 'text' attribute syntax, that it MUST support both the textWithoutLanguage and the textWithLanguage forms. Same for 'name' attribute syntax. Same for an IPP object (Issue 1.48)~~
~~6.5,~~ ~~12.8~~	~~Added new section to allow Attribute Groups to be registered as extensions for being passed in operation requests and responses. (Issue 1.25)~~
~~7.~~	~~Updated the table of text and name attributes to agree with Section 4.2.~~
~~8.5~~	~~Added a new section RECOMMENDING that the Get-Jobs SHOULD return non-IPP jobs whether or not assigning them a job-id and job-uri. Also RECOMMENDED generating, if possible, job-id and job-uri and supporting other IPP operations on foreign jobs as an implementer option. (Issue 1.32)~~
~~9.~~	~~Updated document references.~~
~~14.1.4.14~~	~~Clarified 'client-error-charset-not-supported' that 'utf-8' must be used for any 'text' or 'name' attributes returned in the error response (Issue 1.19).~~
~~14.1.5.9~~	~~Added a new error code 'server-error-job-canceled' (0x0508) to be returned if a job is canceled by another client or aborted by the IPP object while the first client is still sending the document data. (Issue 1.29)~~
~~16.3,~~ ~~16.4~~	~~Moved these sections recommending operation processing steps to the new Implementer's Guide (informational). There indicated that all of the error checks are not required, so an IPP object MAY be forgiving and accept non-conforming requests. However, a conforming client MUST supply requests that would pass all of the error checks indicated. (Issue 1.21)~~
~~17~~	~~Changed directory schema attributes from REQUIRED to RECOMMENDED. Changed some of the OPTIONAL to RECOMMENDED to agree with the SLP template. Changed the "charset-supported" and "natural-language-supported" from REQUIRED to OPTIONAL. Recommended that the names be the same in a directory entry as the IPP attribute names. (Issue 1.53)~~

17. APPENDIX F: Differences between the IPP/1.0 and IPP/1.1 "Model and Semantics" Specifications

This Appendix is divided into two lists that summarize the differences between IPP/1.1 (this document) and IPP/1.0 [RFC2566]. The section numbers refer to the numbers in this document which in some cases have changed from RFC 2566. When a change affects multiple sections, the item is listed once in the order of the first section affected and the remaining affected section numbers are indicated.

The first list contains extensions and clarifications and the second list contains changes in semantics or conformance. However, note that client and IPP object implementations of IPP/1.0 MAY implement any of the extensions and clarifications in this document.

The following extensions and clarifications have been incorporated into this document:

1. Section 2.1 - clarified that the term "client" can be either contained in software controlled by an end user or a part of a print server that controls devices. Issue 4 and Issue 5
2. Section 2.4 - added the description of the new "uri-authentication-supported" Printer Description attribute. Issue 2
3. Section 3.1.3, 3.1.6, 3.2.5.2, and - clarified the error handling for operation attributes that have their own status code. Issues 18, 23, and 27
4. Sections 3.1.7 - Added this new section to clarify returning Unsupported Attributes for all operations, including only returning attributes that were in the request. Issues 18, 23, and 27
5. Sections 3.1.7 and 4.1 - clarified the encoding of the "out-of-band" 'unsupported' and 'unknown' values. Issues 12 and 15
6. Section 3.1.8 - clarified that only the version number parameter will be carried forward into future major or minor versions of the protocol.
7. Section 3.1.8 - indicated that IPP/1.1 Printers SHOULD support version '1.0' and that not all previous minor versions need be supported. Issue 33
8. Section 3.1.8 - relaxed the requirements to increment the major version number. Issue 33
9. Section 3.1.9, and 3.2.5 - added the 'processing' state to the list of job states that a job can be in after a Create-Job operation. Issue 13
10. Section 3.1.9 - clarified that a non-spooling Printer MAY accept zero or more subsequent jobs while processing a job and flow control them down. Subsequent create requests are rejected with the 'server-error-busy' error status. Issue 20
11. Section 3.2.1.1 - clarified the validation of the "compression" operation attribute and its relationship to the validation of the "document-format" attribute and returning Unsupported Attributes. Issues 6, 11, and 28
12. Sections 3.2.1.1, 4.3.8, 13.1.4.16, and 13.1.4.17 - added the 'client-error-compression-not-supported', 'client-error-compression-error' status codes and the 'unsupported-compression' and 'compression-error' job-state-reasons. Issue 28
13. Sections 3.2.1.1 and 4.3.8 - added 'unsupported-document-format' and 'document-format-error' job-state-reasons. Issue 3
14. Sections 3.2.2, 4.3.8 and 13.1.4.19 - added 'client-error-document-access-error' status code and 'document-access-error' job state reason. Issue 35

15. Section 3.2.6 - clarified that "limit" takes precedence over "which-jobs" and "my-jobs’. Issue 8
16. Section 3.2.6.2 - clarified that Get-Jobs returns ’successful-ok’ when no jobs to return. Issue 24
17. Sections 3.2.7, 3.2.8, and 3.2.9 - added the OPTIONAL Pause-Printer, Resume-Printer, and Purge-Jobs operations
18. Sections 3.3.5, 3.3.6, and 3.3.7 - added the OPTIONAL Hold-Job, Release-Job, and Restart-Job operations.
19. Section 4.1.9.1 - clarified that ’application/octet-stream’ auto-sensing can happen at create request time and/or job/document processing time. Issue 9 and Issue 10
20. Section 4.2 - clarified that xxx-supported have multiple keywords and/or names by adding parentheses to the table to give: (1setOf (type3 keyword | name))
21. Section 4.2.2 - added the ’indefinite’ keyword value to the "job-hold-until" attribute for use with the create operations and Hold-Job and Restart-Job operations.
22. Section 4.2.4 - clarified that "multiple-document-handling" MUST be supported if the Printer supports multiple documents per job Issue 34
23. Section 4.2.6 - added more enum values to the "finishings" Job Template attribute.
24. Section 4.3.7 - added that a forwarding server that cannot get any job state MAY return the job’s state as ’completed’, provided that it also return the new ’queued-in-device’ job state reason. Issue 14
25. Section 4.3.7.1 - added the Partitioning of Job States section.
26. Section 4.3.8 - added ’job-data-insufficient’ job state reason to indicate whether sufficient data has arrived for the document to start to be processed. Issue 13
27. Section 4.3.8 - added ’document-access-error’ job state reason to indicate an access error of any kind. Issue 35
28. Section 4.3.8 - added ’job-queued-for-marker’ job state reason to indicate whether the job has completed some processing and is waiting for the marker. Issue 31
29. Section 4.3.8 - added ’unsupported-compression’ and ’compression-error’ job state reasons to indicate compression not supported or compression processing error after the create has been accepted. Issue 6
30. Section 4.3.8 - added ’unsupported-document-format’ and ’document-format-error’ job state reasons to indicate document not supported or document format processing error after the create has been accepted. Issue 3
31. Section 4.3.8 - added ’queued-in-device’ job state reason to indicate that a job as been forwarded to a print system or device that does not provide any job status. Issue 14

32. Section 4.4.3 - clarified "uri-security-supported" is orthogonal to Client Authentication so that 'none' does not exclude Client Authentication. Issue 2
33. Section 4.4.12 - added the 'moving-to-paused' keyword value to the "printer-state-reasons" attribute for use with the Pause-Job operation.
34. Section 4.4.12 - replaced the duplicate 'marker-supply-low' keyword with the missing 'toner-empty' keyword for the "printer-state-reasons" attribute.
35. Section 4.4.15 - added the enum values to the "operations-supported" attribute for the new operations. Clarified that the values of this attribute are encoded as any enum, namely 32-bit values.
36. Sections 4.4.36 and 4.4.37 - added the OPTIONAL "pages-per-minute" and "pages-per-minute-color" Printer Description attributes.
37. Section 4.4.30 - clarified that the dateTime value of "printer-current-time" is on a "best efforts basis". If a proper date-time cannot be obtained, the implementation returns the 'no-value' out-of-band value. Also clarified that the time zone NEED NOT be the time zone that the people near the device use and that the client SHOULD display the dateTime attributes in the user's local time. Issue 17
38. Section 5.1 - clarified that any response MAY contain additional attribute groups, attributes, or attribute values. Issues 25 and 26
39. Section 5.1 - clarified that a client SHOULD do its best to prevent a channel from being closed by a lower layer when the channel is flow controlled off by the IPP Printer. Issues 4 and 5
40. Section 8.3 - clarified the use of URIs for each Client Authentication mechanism.
41. Section 8.5 - added the security discussion around the new operator operations.
42. Section 13.1.4.16 - added client-error-compression-not-supported (0x040F) Issue 6
43. Section 13.1.4.17 - added client-error-compression-error (0x0410) Issue 6
44. Section 13.1.4.18 - added client-error-document-format-error (0x0411) Issue 28
45. Section 13.1.4.19 - added client-error-document-access-error (0x0412) Issue 35
46. Section 13.1.5.10 - added server-error-multiple-document-jobs-not-supported (0x0509) Issue 34
47. Section 16 - added the OPTIONAL "pages-per-minute" and "pages-per-minute-color" Printer attributes to the Directory schema.
48. Section 16 - added OPTIONAL "multiple-document-jobs-supported" to the Directory schema. Issue 34
49. Section 16 - added RECOMMENDED "uri-security-supported", "compression-supported", and "ipp-versions-supported" to the Directory schema. Issues 2,

The following changes in semantics and/or conformance have been incorporated into this document:

1. Section 3.1.8, 5.2.4, and 13.1.5.4 - Clients and IPP objects MUST support version 1.1 and SHOULD support version 1.0. Issue 33 and Issue 36
2. Section 3.2.1.1 and section 4.4.32 - changed the "compression" and "compression-supported" attributes from OPTIONAL to REQUIRED. Issue 28
3. Sections 3.2.1.2 and 4.3.8 - changed "job-state-reasons" from RECOMMENDED to REQUIRED, so that "job-state-reasons" MUST be returned in create operation responses. Issue 30
4. Sections 3.2.4, 3.3.1, 4.4.16, and 16 - changed Create-Job/Send-Document so that they MAY be implemented while only supporting one document jobs. Added the "multiple-document-jobs-

supported" boolean Printer Description attribute to indicate whether Create-Job/Send-Document support multiple document jobs or not. Added to the Directory schema. Issue 34

5. Section 4.1.9 - deleted 'text/plain; charset=iso-10646-ucs-2', since binary is not legal with the 'text' type.
6. Section 4.3.8 - changed "job-state-reasons" from RECOMMENDED to REQUIRED. Issue 30
7. Section 4.3.12 - added OPTIONAL 'dateTime' attribute syntax to "time-at-creation", "time-at-processing", and "time-at-completed" Job Description attributes for use in version '1.1' responses. Issue 17
8. Section 4.3.12 - changed the "time-at-creation", "time-at-processing", and "time-at-completed" Event Time Job Description attributes from OPTIONAL to REQUIRED. Issue 17
9. Section 4.3.12.4 - added the REQUIRED "job-printer-up-time (integer(1:MAX))" Job Description attribute as an alias for "printer-up-time" to reduce number of operations to get job times. Issue 17
10. Section 4.4.2 - added the REQUIRED "uri-authentication-supported (1setOf type2 keyword)" Printer Description attribute to describe the Client Authentication used by each Printer URI. Issue 2
11. Section 4.4.11 - clarified the "printer-state" to allow a Printer that can interpret one or more jobs (rip) while marking one job to have those jobs all in the 'processing' state. Issue 31
12. Section 4.4.12 - changed "printer-state-reasons" Printer Description attribute from OPTIONAL to REQUIRED. Issue 30
13. Section 4.4.14 - added the REQUIRED "ipp-versions-supported (1setOf keyword)" Printer Description attribute, since IPP/1.1 Printers do not have to support version '1.0'.
14. Section 4.4.16 - added the REQUIRED "multiple-document-jobs-supported (boolean)" Printer Description attribute so that a client can tell whether a Printer that supports Create-Job/Send-Document supports multiple document jobs or not. Issue 34
15. Section 4.4.24 - changed the "queued-job-count" Printer Description attribute from RECOMMENDED to REQUIRED. Issue 29
16. Section 4.4.32 - changed "compression-supported (1setOf type3 keyword)" Printer Description attribute from OPTIONAL to REQUIRED. Issue 28
17. Section 5.1 - changed the client security requirements from RECOMMENDED non-standards track SSL3 to MUST/SHOULD [which is to be determined in consultation with the Area Director] support Client Authentication as defined in the IPP/1.1 Encoding and Transport document [IPP-PRO]. A client SHOULD support Operation Privacy and Server Authentication as defined in the IPP/1.1 Encoding and Transport document [IPP-PRO].
18. Section 5.2.7 - changed the IPP object security requirements from OPTIONAL non-standards track SSL3 to MUST/SHOULD [which is to be determined in consultation with the Area Director] contain support for Client Authentication as defined in the IPP/1.1 Encoding and Transport document [IPP-PRO]. A Printer implementation MAY allow an administrator to configure the Printer so that all, some, or none of the users are authenticated. An IPP Printer implementation SHOULD contain support for Operation Privacy and Server Authentication as defined in the IPP/1.1 Encoding and Transport document [IPP-PRO]. A Printer implementation MAY allow an administrator to configure the degree of support for Operation Privacy and Server Authentication.

See also the "IPP/1.1 Encoding and Transport" [IPP-PRO] document for differences between IPP/1.0 [RFC2565] and IPP/1.1 [IPP-PRO].

18. Full Copyright Statement