R. deBry
IBM Corporation
T. Hastings
Xerox Corporation
R. Herriot
Sun Microsystems
S. Isaacson
Novell, Inc.
P. Powell
San Diego State University
June 3, 1997

Internet Printing Protocol/1.0: Model and Semantics
draft-ietf-ipp-model-01.txt

Status of this Memo

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 technology. The protocol is heavily influenced by the printing model introduced in the Document Printing Application (ISO/IEC 10175 DPA) standard. Although DPA specifies both end user and administrative features, IPP version 1.0 is focused only on end user functionality.

The full set of IPP documents includes:

Internet Printing Protocol: Requirements
Internet Printing Protocol/1.0: Model and Semantics
Internet Printing Protocol/1.0: Security
Internet Printing Protocol/1.0: Protocol Specification
Internet Printing Protocol/1.0: Directory Schema

The requirements 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. The requirements document calls out a subset of end user requirements that must be satisfied in the first version of IPP. Operator and administrator requirements are out of scope for v1.0. The model and semantics document describes a simplified model with abstract objects, their attributes, and their operations. The model introduces a Printer object and a Job object. The Job object supports multiple documents per job. The security document covers potential threats and proposed counters to those threats. The protocol specification is formal document which incorporates the ideas in all the other documents into a concrete mapping using clearly defined data representations and transport protocol mappings that real implementers can use to develop interoperable client and server side components. Finally, the directory schema document shows a generic schema for directory service entries that represent instances of IPP Printers.

This document is the "Internet Printing Protocol/1.0: Model and Semantics" document.

Table of Contents

1.

Introduction

The Internet Printing Protocol (IPP) is an application level protocol that can be used for distributed printing on the Internet. The protocol is heavily influenced by the printing model introduced in the Document Printing Application (ISO/IEC 10175 DPA) standard. Although DPA identifies both end user and administrative features, the first version of IPP is focused only on end user functionality.

Section 2. Terminology introduces the terminology used within this document.

Section 0 introduces the simplified IPP model. The IPP model is made simple by exposing only the objects, attributes, and operations that are essential for end user access and control of the print subsystem. When future versions of IPP include features which satisfy operator and administrator requirements, the model can be extended to support the appropriate objects, attributes, and operations.

Section 0 introduces the full semantics of the Printer, Job, and Document objects in the IPP model. It covers how instances of these objects are identified, named, and related to each other.

Section 0 covers the operations that are part of the IPP model. These operations include: the Create-Job, Send-Document, Print-Job, Cancel, Get-Attributes, and Get-Jobs operations.

Section 0 describes the attributes, their syntaxes, and semantics which are part of the IPP model. Each object's attributes are described, and the attributes are grouped into logical groups to help clarify their relationships and meaning. These groups are also used to simplify queries that request multiple attributes.

Section 7. Conformance is a review of conformance issues and clarifies requirements that apply to client side and server side implementations.

Sections 8. IANA Considerations, Registered Extensions, Private Extensions-11. Author's Address cover extensibility, security, technical references, and author information.

2. Terminology

This specification uses the terminology defined in this section.

2.1 Conformance Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [25].

2.1.1 MUST

This word, or the terms "REQUIRED", "SHALL" or "MANDATORY", mean that the definition is an absolute requirement of the specification.

2.1.2 MUST NOT

This phrase, or the phrase "SHALL NOT", mean that the definition is an absolute prohibition of the specification.

2.1.3 SHOULD

This word, or the adjective "RECOMMENDED", mean that there may exist valid reasons in particular circumstances to ignore a particular item, but the full implications must be understood and carefully weighed before choosing a different course.

2.1.4 SHOULD NOT

This phrase, or the phrase "NOT RECOMMENDED" mean that there may exist valid reasons in particular circumstances when the particular behavior is acceptable or even useful, but the full implications should be understood and the case carefully weighed before implementing any behavior described with this label.

2.1.5 MAY

This word, or the adjective "OPTIONAL", mean that an item is truly optional. One vendor may choose to include the item because a particular marketplace requires it or because the vendor feels that it enhances the product while another vendor may omit the same item. An implementation which does not include a particular option MUST be prepared to interoperate with another implementation which does include the option, though perhaps with reduced functionality. In the same vein an implementation which does include a particular option MUST be prepared to interoperate with another implementation which does not include the option (except, of course, for the feature the option provides.)

2.1.6 CONDITIONALLY MANDATORY

This term means that an item MUST be implemented in a conforming implementation if the item corresponds to a feature or behavior that the implementation is capable of realizing. It is also true, that a conforming implementation NEED NOT implement the items that correspond to features or behaviors that the implmentation is not capable of realizing.

2.1.7 NEED NOT

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.

2.2 Model Terminology

2.2.1 Keyword

Keywords are used within this document as identifiers of semantic entities within the abstract model. These entities are often attribute names, attribute values, attribute syntaxes, and attribute groups. In this document, a keyword is a sequence of characters (length of 1 to 255) which consists of the following ASCII characters: letters, digits, hyphen ("-"), and underscore ("_"). A keyword MUST start with a letter. In the actual protocol, these keywords will be represented using an appropriate protocol encoding (strings, enumerated values, constants, operation codes, identifiers, etc.).

2.2.2 Attributes

An attribute is an item of information consisting of an attribute name and attribute value(s) using a specific syntax for that attribute. All attributes are defined in section 6. Object Attributes. Attributes are identified as being "MANDATORY", "CONDITIONALLY MANDATORY", or "OPTIONAL".

2.2.3 Attribute Name

Each attribute is uniquely identified in this document by its attribute name which 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 (").

2.2.4 Group Name

Related attributes are grouped into named groups. The name of the group is a keyword. It can be used wherever an attribute name is used in

place of naming all the attributes in the group explicitly. Attribute groups are defined in section 6. Object Attributes.

2.2.5 Attribute Value

Each attribute shall have one or more values. Attribute values shall be 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 syntax types are keyword, integer, text, etc.

2.2.6 Attribute Syntax

Each attribute is defined using an explicit syntax. In this document, each syntax type is defined as a keyword with specific meaning. The protocol specification document [23] shall indicate the actual representation for each syntax type that shall be used for the actual protocol. Attribute syntaxes are defined in section 6.1 Attribute Syntaxes.

2.2.7 Implements

By definition, an attribute is implemented if, in response to a query for that attribute, an implementation responds with both the attribute and a current value (or values) for that attribute. . A conforming implementation SHALL implement all MANDATORY attributes and it SHALL implement all CONDITIONALLY MANDATORY attributes whose possible values correspond to the behaviors that the implementation is capable of realizing.

2.2.8 Supports

By definition, a job processing behavior or selectable feature is supported by Printer only if that Printer implements the corresponding "supported" attribute populated with the value representing that behavior or feature. A given implementation may exhibit a behavior that corresponds to the value of some supported attribute, but if the implementation, when queried for that attribute, doesn't respond with the supported attribute populated with that specific value, then as far as IPP is concerned, that Printer does not support that feature. Section 6. Object Attributes describes all of the Printer object's supported attributes. Most of the Printer object's supported attributes are OPTIONAL or CONDITIONALLY MANDATORY, therefore conformance to IPP

does not mandate that all implementations support all possible value 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 SHALL implement the "document-format-supported" attribute and that attribute SHALL be populated with a set of values, possibly only one, taken from the entire set of possible values defined in this model document. This implemented set of values represent the Printer's set of supported document formats. Another example is the "finishings-supported" attribute. If a Printer is not physically capable of stapling (there is no stapler in the output device itself), the "finishings-supported" attribute MUST NOT be implemented with the value of 'staple'. Note: The supported attributes are set (populated) by some administrative process or automatic sensing mechanism which is outside the scope of IPP.

3. Simplified Printing Model

In order to a achieve its goal of realizing a workable printing protocol for the Internet, IPP is based on a simplified printing model which abstracts the many (often complex) components of real world printing solutions. Many of these systems include features, interfaces, and relationships that are beyond the scope of IPP. IPP has to run in a distributed computing environment where requesters of print services (clients, applications, PC drivers, etc.) cooperate and interact with print service providers. Although the underlying configuration may be a complex n-tier client/server system, an important simplifying step in the IPP model is to expose only the key objects and interfaces required for printing. The IPP model encapsulates these important elements into three simple objects:

Printer (Section 0)
Job (Section 0)
Document (Section 0)

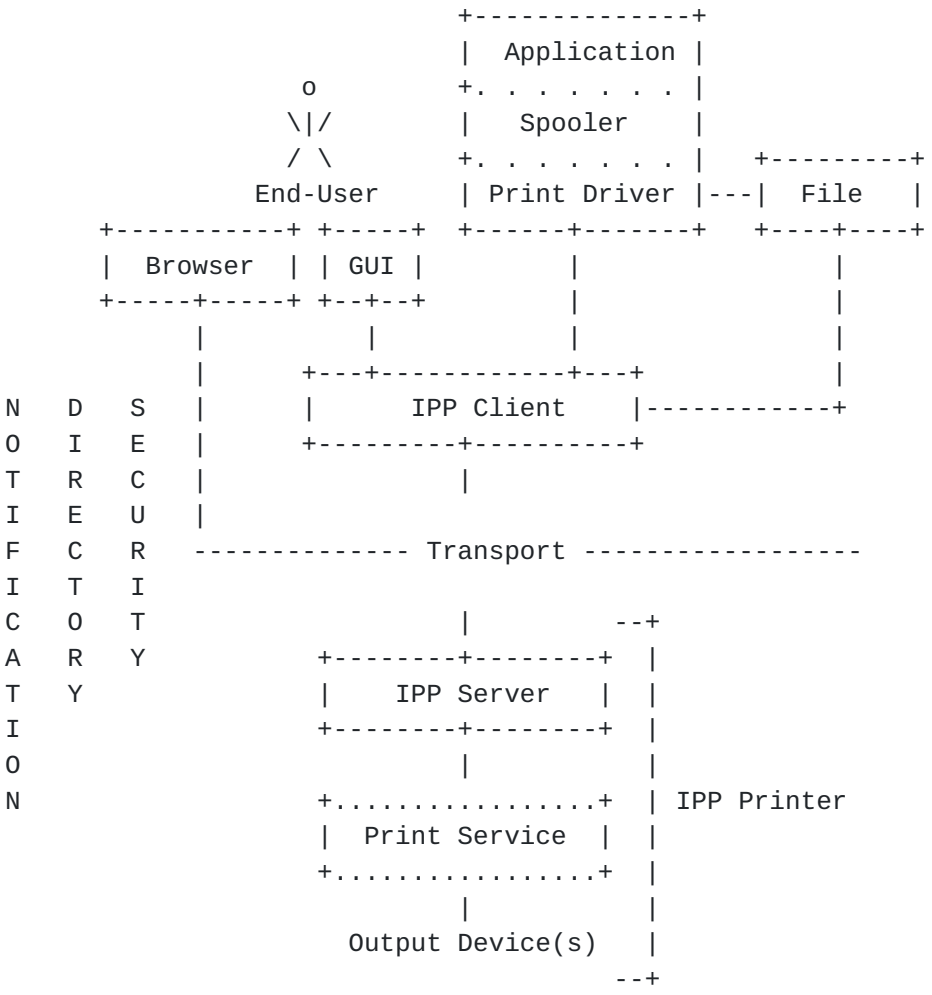
Each of these objects has a set of operations associated with it. These include:

Printer:
   Create-Job (Section 0)
   Print-Job (Section ???)
   Get-Attributes (Section 0)
   Get-Jobs (Section 0)
Job

Send-Document(section ???)
Get-Attributes (Section 0)
Cancel Job (Section 0)

There are no operations defined for a Document object. All document information is accessed through a Job object and its operations.

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



IPP Printers encapsulate the functions normally associated with physical output devices along with the spooling, scheduling and multiple device management functions associated with a print server. Printers may be

registered as entries in a directory where end users find and select them based on some sort of filtered and context based searching. 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 (name, context, printer capabilities, etc.)

IPP clients implement the IPP protocol on the client side and give end users or programs the ability to query an IPP Printer and submit and manage their print jobs. An IPP server is just that part of the IPP Printer that implements the protocol. The rest of the IPP Printer implements the application semantics of the print service itself. All information about the Printer, both static and dynamic information, can be accessed directly from the Printer itself. The more dynamic information associated with a Printer includes state, currently loaded and ready media, number of jobs on the Printer, errors, warnings, etc.. When a job is submitted to the Printer, a Job object is created. The end user then interacts with this new Job to query its status and monitor the progress of the job. End users may also cancel the Job. The end user is able to register to receive certain events which are then routed using the notification service(s).

4. IPP Objects

4.1 Printer Object

A major component of the IPP model is the Printer object.

The capabilities and state of an IPP Printer are described by its attributes. Printer attributes are defined in the following categories:

Job Template Attributes (section 6.5.1 Printer Job Template Attributes)
Printer Description Attributes (section 6.5.2 Printer Description Attributes)

Operations which are invoked on a printer include:

Create-Job (section 0)
Print-Job (section ???)
Get-Attributes (section 0)
Get-Jobs (section 0)

An instance of a Printer object implements IPP. Using the protocol, end users may query the attributes of the Printer, submit jobs to the Printer, determine subsequent states of submitted and queued jobs, and cancel their own print jobs. The realization of a Printer object may take on different forms for any given configuration of real components. However, the details of the configuration of real components must be

transparent to the end user.

Since a Printer object is an abstraction of a generic document output device or print service provider, an IPP Printer object could be used to represent any real or virtual device with semantics consistent with the Printer object. For example, an instance of a Printer object could be used to front end a fax-out device, any kind of 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 might or might not be capable of spooling jobs
   3b) The associated output devices might or might not support IPP

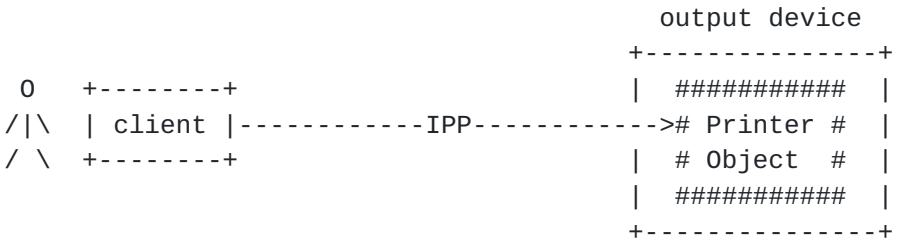
See the following figures for some examples on how to view Printer objects on top of several print system configurations. The embedded case below represents configurations 1 and 2. The hosted and fan-out figures below represent configuration 3.

Legend:

indicates a Printer object which is
either embedded in an output device or is
hosted in a server. The implementation
might or might not be capable of queuing/spooling.

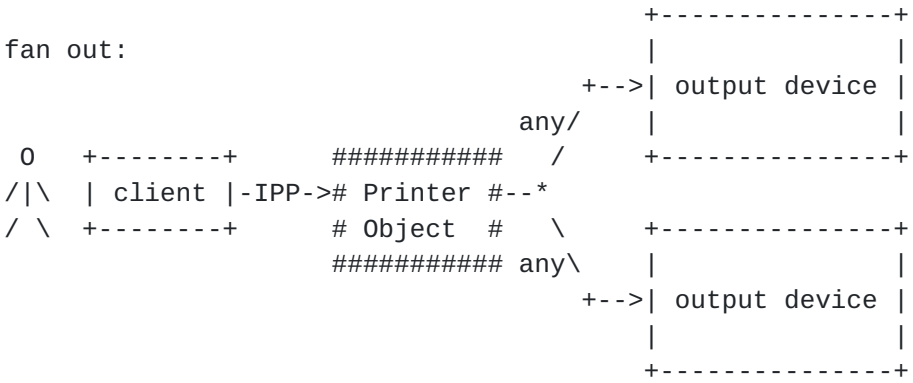
any indicates any network protocol or direct
connect, including IPP

embedded printer:



hosted printer:

fan out:



4.2 Job Object

A Job object is used to model a job. A job can contain one or more documents. The information required to create a Job object is sent in a create request from the end user via an IPP client to a Printer. A

create request can be either a Create-Job Request or a Print-Job Request. The Printer may perform some validation checks to verify that the job may indeed be processed. For example, the create request may specify that the documents within the job are to be printed duplex (on both sides of the media). However, the Printer might not support such a feature. Once the Printer validates the submitted information, a Job object is created. The instance of the Job object is initialized with information from the create request. If a Create-Job operation is used to create the Job object, subsequent Send-Document operations are used to transfer the document data from the client to the IPP Printer.

This model specification defines rules for what MUST be done when:

- optional attributes are missing
- there are conflicts between what is supported and what is requested
- there are conflicts between what the client requests via external attributes in the IPP operation and what the client requests in embedded instructions in the document page description language (PDL).

Job attributes are broken up into the following groups:

Job Template Attributes (optionally supplied by the client/end user, section 6.3.1 Job Template Attributes
Job Description Attributes (set by the Printer, section 6.3.2 Job Description Attributes)

The following operations can be invoked on Jobs:
Send-Document(section 5.1.3 Send-Document Operation)
Cancel Job (section 0)
Get-Attributes (section 0)

4.3 Document Object

Documents consist of printable data and attributes that describe the data to be printed. In this version of the protocol only the attributes in section 6.4 Document Attributes are defined for individual documents. Documents are sent in a Send-Document operation or in a Print-Job operation.

Document attributes include:

Document Attributes (section 0)

Currently no operations are defined on documents.

4.4 Object Relationships

Instances of objects within the system have relationships that must be maintained persistently along with the persistent storage of the object attributes. A Printer can represent one or more output devices. An output device can be represented by at most one Printer object. A Printer can contain zero or more Job objects. A Job object is contained in exactly one Printer object. A Job object contains one or more Documents. If the Document is simply a reference to some print data stream, the reference may be used in multiple documents in the same Job or even in different Jobs. If the Document is not just a reference, but an actual stream of print data, it shall only be contained in one Document, although there can be copies of it in other Documents.

4.5 Object Attributes

Each object type is defined by a set of attributes which describe the realization of each instance of an object. That is, a Printer object is defined as set of attributes that are associated with each instance of a Printer object. In the same manner, a Job object is defined by defining the set of attributes that are associated with each instance of a Job object. Some attributes are OPTIONAL, some are MANDATORY, and some are CONDITIONALLY MANDATORY (see section 2). Object attributes are defined in section 0 of this document.

4.5.1 Job Template Attribute Overview

Attributes that a client may optionally include in a create request are called Job Template attributes. These are described in detail in section 0. The Printer object has associated attributes which define supported and default values for the Printer.

- When a Job Template attribute is supplied by a client in a create request, the attribute and its value describe the desired job processing behavior.

- The Printer object's supported attribute describes what behaviors are possible.

- The Printer object's default value attribute describes what will be done when no other job processing information is supplied by the client.

4.5.2 The "best-effort" Job Attribute Overview

A client supplies Job Template attributes to 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, within the Page Description Language (PDL) of the document data. If there is a conflict between the value of one of these IPP Job Template attributes, and a corresponding instruction in the document (either implicit or explicit), it is desirable that the value of the attribute shall take precedence over the document instruction. Until companies that supply interpreters for PDLs, such as PostScript and PCL allow a way to external attributes (such as IPP attributes) to take precedence over internal job production instructions, a Printer might not be able to implement the semantics that IPP attributes override (take on a higher precedence) the embedded PDL instructions. Therefore, IPP introduces a special Job Template attribute named "best-effort". This attribute gives the end user some ability to influence, or at least understand, how a particular Printer implementation handles these conflicts.

This attribute takes on the following values:

- 'shall-honor-ipp-attributes': If a Printer supports this value and a client requests this value, the Printer guarantees that all IPP attribute values take precedence over embedded instructions in the job data (the PDL of the job's documents).
- 'should-honor-ipp-attributes': If a Printer supports this value, and a client requests this value, the Printer should try to make sure that IPP attribute values take precedence over embedded PDL instructions, however there is no guarantee

ISSUE: Should these be 'shall-honor-attribute-precedence' and 'should-honor-attribute-prcedence'?

A Printer SHALL implement the "best-effort-supported" attribute. Notice that since 'should-honor-ipp-attributes' does not offer any type of guarantee, a Printer may not do a very "good" job of implementing the semantics of "should", but it would still be a conforming implementation.

If there is ever a conflict between what a Printer supports and what an IPP client requests, the Printer shall reject the print request. A client should query the printer to find out what is supported before making a request. This ensures that all requested attribute values are supported.

ISSUE: Should this be called "effort-level" rather than "best-effort"?

4.6 Object Identity

All instances of Printer and Job objects have an identifier attribute whose value is globally unique so that they can persistently and unambiguously referenced. The IPP model requires that these values be URIs as defined by RFC 1738 and RFC 1808. In addition to an identifier attribute, instances of Printer and Job objects may have a name. An object name need not be unique across all instances of all objects. The Printer name is chosen and set by an administrator. The Job name is created by the Printer using the name of the first document in the job. In all cases, the name only has local meaning, and is not constrained to be globally unique.

To summarize, each instance of Printer and Job objects will have two identifying attributes:

- "xxx-URI": The globally unique identifier for this object instance
- "xxx-name": The non-globally unique name for this object instance

Document objects only have names, no identifiers. The "document-name" attribute is used to store the name of the Document. This name is just of interest within the context of a Job. It need not be globally unique.

If Documents are printed by reference, they are identified by URIs.

5. IPP Operations

Jobs and Printers each have a set of associated operations. End users or programs invoke these operations using an IPP client. The operations are:

For a Printer object:
- Create-Job (section 0)
- Print-Job (section ???)
- Get-Attributes (section 0)
- Get-Jobs (section 0).

For a Job object:
- Send-Document (section ??)
- Cancel-Job (section 0)
- Get-Attributes (section 0)

IPP Job and Printer objects are identified by URIs. When a client communicates with a remote IPP object, it sends an operation request to

the URI for that object. Each request carries along with it the input parameters and data required to perform the specified operation. Each request requires a response from the object indicating success or

failure of the operation including response data and/or error messages. The representation and encoding of the IPP protocol are contained in "Internet Printing Protocol: Protocol Specification."[23]

It is assumed that URIs for IPP Printers are available to end users or programs that wish to invoke Printer operations. Although NOT MANDATORY, it is RECOMMENDED that Printers be registered in a directory service which end users and programs can interrogate. "Internet Printing Protocol: Directory Schema"[24] defines the attributes to be associated with a Printer entry in a directory service.

5.1 Operation Semantics

In this section, the IPP operations are described in terms of their contents and semantics including both the request and the response.

In order to create a new Job object, a client MAY use one of two operations: either the Create-Job operation or the Print-Job operation. If the client wants to create a Job with only a single Document, the client MAY use the Print-Job operation or a Create-Job operation followed by a single Send-Document operation. For performance reasons, the client SHOULD use the Print-Job operation for all single Document Jobs. If the client wants to create a Job with more than one Document, the client SHALL use the Create-Job operation followed by as many Send-Document operations as needed (on Document per Send-Document operation). The Print-Job operation is a convenience operation for creating a Job with only one Document. Throughout this model specification, the term create request is used to refer to either a Create-Job Request or a Print-Job Request.

5.1.1 Print-Job Operation

When an end user desires to submit a print job with only one document, the client sends a Print-Job Request to a Printer and receives a Print-Job Response from that Printer. The information in a Print-Job Request (along with any default information associated with the Printer) is sufficient for the Printer to create a Job object and then process that Job.

5.1.1.1 Print-Job Request

The following elements are part of the Print-Job Request:

Job Template Attributes:
An optional set of Job Template attributes as defined in section

6.2 Job Template Attributes. If the client supplies no Job Template attributes in the Create-Job Request, the Printer uses its default value attributes when processing the job.

Document Content
The client either supplies the raw document data or a URI reference to the data but not both.

The simplest Print-Job Request consists of just the Document Content and nothing else. This means that the Printer SHALL create a new Job object with no Job Template attributes and a single contained Document.

When a Printer receives a Print-Job Request, the Printer SHALL either accept or reject the request. The Printer SHALL accept the Print-Job Request and SHALL create a Job object if it is able to accept all attributes in the request. The Printer SHALL reject the request and SHALL NOT create a Job object if the Printer rejects any attribute in the request. There are six cases to consider with respect to each Job and Document attributes:

1. The client supplies a Job Template attribute named "xxx" and the value supplied by the client is among the values supported by the Printer (i.e., is among the values of the Printer's "xxx-supported" attribute): The "xxx" Job Template attribute is accepted. If the "best-effort-supported" attribute contains the value 'shall-honor-ipp-attributes' the Printer SHALL guarantee the behavior represented by the value in the "xxx" attribute (i.e., the IPP attribute has precedence over any other embedded job instruction). If the value of the "best-effort-supported" is 'should-honor-ipp-attributes' then the Printer SHOULD try to realize the behavior requested by the client, but NEED NOT guarantee the behavior. The Printer creates the Job object and implements the "xxx" attribute in the new Job object and uses the value supplied by the client.

2. The client supplies an attribute value but the attribute is syntactically bad: The Printer shall reject the job and return the ??? error code.

3. The client supplies an attribute value and the attribute value is not among the values supported by the Printer: The Printer SHALL reject the job and return the 'attribute-unsupported' error code.

4. The client supplies an attribute value and the Printer does not implement the attribute: The attribute is ignored. The Printer behaves as if the attribute was never supplied by the client in the Print-Job Request. In the response, the Printer SHALL return the names of all ignored attributes. The final result of the Job is undefined for an ignored attribute (that is the desired behavior

might or might not be realized).

ISSUE: Should the printer just reject the job?

5. The client does not supply an attribute, but the Printer implements the attribute: The attribute is accepted and when the Printer creates the Job object, the Printer SHALL NOT implement the attribute in the Job object. When the Printer processes that Job, the Printer SHOULD attempt to use the behavior implied by the default value Printer attribute as set at the time of Job processing (not Job creation). In other words, these rules allow for a Job object to be created without implementing some of the Job Template attributes. As the Printer processes the Job, if the Printer implements a default value attribute for the missing Job Template attribute, the Printer does its best to realize the behavior of the default value. If the Printer does not implement the default value attribute, the results are undefined.

   Note: For each Job Template attribute, this specification REQUIRES that a Printer to implement the CONDITIONALLY MANDATORY attributes.

6. The client does not supply an attribute, and the Printer does not implement the attribute: The Printer accepts the Job and creates a Job object. When the Job is processed, the actual behavior realized with respect to the missing Job Template attribute is undefined.

5.1.1.2 Print-Job Response

The Printer shall return to the client the following output parameters as part of the Print-Job Response:

Job Identifier:
A URI which the client shall use for all other operations on this Job

Job Status:
The following Job attributes: job-name, job-state, and job-state-reasons. The value of each attribute shall be from a snapshot taken sometime after the time the Printer receives the print request. The "job-state-message" attribute is OPTIONAL.

Note: Since any job affecting printer state information is reflected in the "job-state" and "job-state-reasons" job attributes, it is sufficient to return only job status attributes and no printer status attributes at Job creation time.

Ignored Attributes:
A list of attribute names which were ignored in the creation of the Job object.

Unsupported Attributes:
A list of attribute names which are unsupported. Any attributes in this list imply that the Job object was not created.

Bad Attributes:
A list of attribute names which were syntactically incorrect. Any attributes in this list imply that the Job object was not created.

Status
Status information including error status

The simplest response shall consist of the job identifier, the Job Status attributes, and an operation status that is either an "ok" status or an "error" status.

5.1.2 Create-Job Operation

When an end user desires to submit a print job, the client sends a Create-Job Request to a Printer and receives a Create-Job Response from that Printer. The information in a Create-Job Request along with any default information associated with the Printer is sufficient for the Printer to create a Job object. An instance of a Job object contains all the information needed by the Printer to print one or more documents as a print job. If the client follows the Create-Job operations with as many Send-Document operations as needed.

5.1.2.1 Create-Job Request

The following elements are part of the Create-Job Request:

Number of Documents:
An optional integer value specifying the number of Documents for this Job. The document data is transferred in a series of subsequent Send-Document operations (one document per Send-Document operation). If this value is not supplied by the client, the Printer waits to receive an empty Send-Document operation signaling the end of Documents for this Job. If the client wants to create a Job with only a single document, the client MAY use the Print-Job operation. This is a convenience operation for creating a Job with only one Document. The Print-Job Operation is semantically the same as a Create-Job operation followed by one Send-Document operation.

Job Template Attributes:

An optional set of Job Template attributes as defined in section 6.2 Job Template Attributes. If the client supplies no Job

Template attributes in the Create-Job Request, the implemented Printer defaults are used.

The simplest Create-Job Request has no data which means that the Printer SHALL create a new Job object with no Job Template attributes and the number of Documents is yet to be determined.

When a Printer receives a Create-Job Request, the Printer SHALL either accept or reject the request. The rules for accepting or rejecting a Create-Job Request are the same as the rules for accepting or rejecting the Print-Job Request (see section 5.1.1.1 Print-Job Request).

5.1.2.2 Create Job Response

The Printer shall return to the client the following output parameters as part of the Create-Job Response:

Job Identifier:
A URI which the client shall use for all other operations on this Job

Job Status:
The following Job attributes: job-name, job-state, and job-state-reasons. The value of each attribute shall be from a snapshot taken sometime after the time the Printer receives the print request.

Note: Since any job affecting printer state information is reflected in the "job-state" and "job-state-reasons" job attributes, it is sufficient to return only job status attributes and no printer status attributes at Job creation time.

Ignored Attributes:
A list of attribute names which were ignored in the creation of the Job object.

Unsupported Attributes:
A list of attribute names which are unsupported. Any attributes in this list imply that the Job object was not created.

Bad Attributes:
A list of attribute names which were syntactically incorrect. Any attributes in this list imply that the Job object was not created.

Status
   Status information including error status

The simplest response shall consist of the job identifier, the Job Status attributes, and an operation status that is either an "ok" status or an "error" status.

5.1.3 Send-Document Operation

Once a Job object has been created using a Create-Job operation, a client uses the Send-Document operation to transport the documents to be printed and add them to the named Job object. A document MUST be sent in a single Send-Document operation.

5.1.3.1 Send-Document Request

The client submits the request to a Job URI.

The following abstract data types are part of the Send-Document Request:

Document Attributes:
An optional set of Document Description attributes (section 6.4 Document Attributes).

Document Content:
The client either supplies the raw document data or a URI reference to the data but not both.

5.1.3.2 Send-Document Response

The following abstract data types are part of the Send-Document Response:

Job Status:
The following Job attributes: job-name, job-state, and job-state-reasons. The value of each attribute shall be from a snapshot taken sometime after the time the Printer receives the print request.

Note: Since any job affecting printer state information is reflected in the "job-state" and "job-state-reasons" job attributes, it is sufficient to return only job status attributes and no printer status attributes at Job creation time.

Ignored Attributes:

A list of attribute names which were ignored in the creation of the Job object.

Unsupported Attributes:
A list of attribute names which are unsupported. Any attributes in this list imply that the Job object was not created.

Bad Attributes:
A list of attribute names which were syntactically incorrect. Any attributes in this list imply that the Job object was not created.

Status:
Status information including error status

5.1.4 Cancel Job Operation

This operation allows a user to cancel one specific Print Job any time after the print job has been established on the Printer. Some pages may be printed before a job is terminated if printing has already started when the Cancel Job operation is received. Only the end user who is also the job originator ("job-originating-user" Job attribute) can cancel the job using IPP 1.0.

5.1.4.1 Cancel-Job Request

The client submits the request to a Job URI.

The following abstract data types are part of the Cancel Job Request:

Message:
Optional message to the operator

5.1.4.2 Cancel-Job Response

The following information is part of the Cancel Job Response:

Status:
Status information including error status

5.1.5 Get-Attributes Operation

The Get-Attributes operation allows client to obtain information from a Printer or Job object. The client supplies the set of attributes names and/or attribute group names that the requester is interested in as operation input parameters. The Printer shall return a corresponding attribute list in the response with the appropriate attribute values

filled in for each attribute (explicitly named or implicitly included in an attribute group) that the client supplied in the request.

5.1.5.1 Get-Attributes Request

The client shall submit the Get-Attributes request to a Job URI or Printer URI.

The following input parameters shall be part of the Get-Attributes Request:

Document Format:
The client shall supply this input parameter only when requesting attributes of the Printer object. The Printer shall reject this request, if this input parameter is supplied for a Job object.

This input parameter conditions the Printer attributes and values that might depend on the document format. The Printer shall return only (1) those attributes that are implemented and (2) the attribute values that are supported for the specified document format. By specifying the document format, the client can eliminate the attributes that are not implemented and values that are not supported for the document format that the client has or is able to generate.

If the client omits this input parameter, the effect shall be the same as if the value of the Printer's document format attribute were supplied. It is recommended that the client always supply a value for document-format, since the Printer's default value for document-format may be 'auto-sense', in which case the returned attributes and values are for the union of the document formats that the Printer supports in its 'auto-sense' support."

Requested Attributes:
An optional set of attribute names (without values) or attribute group names in whose values the requester is interested. If the client omits this input parameter, the effect shall be the same as if the "all" attribute group were supplied.

Attributes may be requested by name or by group name. For Jobs, the attribute groups include:

- 'job-template': the attributes specified in Section 6.3.1 Job Template Attributes.
- 'job-description': the attributes specified in Section 6.3.2 Job Description Attributes.

For Printers, the attribute groups include:

- 'printer-job-template': the attributes specified in Section 6.5.1 Printer Job Template Attributes.

- 'printer-description': the attributes specified in Section 6.5.2 Printer Description Attributes.

There are also special groups:

- 'none': no attributes of the specified object. Note: none is primarily useful in Get-Jobs, but can be used as a "ping" with the Get-Attributes operation.
- 'all': all attributes of the specified object

5.1.5.2 Get-Attributes Response

The Printer shall return the following output parameters as part of the Get-Attributes Response:

Result Attributes:
The requested attributes of the object with their current values, if the requester supplied any Requested Attributes. If the request did not supply any attribute names, the Printer shall assume that the client is implicitly requesting the default group of "all" and shall return all attributes implemented for the specified Job or Printer object.

Unimplemented Attributes:
A list of attribute names which are unimplemented.

Unknown:
A list of attribute names which are unknown.

Status:
Status information including error status

A Printer may choose, for security reasons, not to return all attributes that a client requests. It may even return none of the requested attributes. In such cases, the status returned is the same as if the Printer had returned all requested attributes. The client cannot tell by such a response whether the requested attribute was present or absent on the Printer.

In response to a "Get-Attributes" (or a "Get-Jobs") operation the following requirements apply to the Printer:

1. If the client supplies an attribute name in the Requested

Attribute input parameter and that attribute is implemented by the Printer, the printer shall respond with all current values for that attribute. If the value of an implemented attribute is unknown for

some reason, the Printer shall respond with the attribute name in the "unknown attribute list" response parameter.

2. If the client supplies an attribute name in the Requested Attribute input parameter that attributed is not implemented by the Printer, the Printer shall respond with the attribute name in the "unimplemented attribute list" response parameter.

3. If the client supplies an attribute group that is implemented by the Printer, the Printer shall respond with all current values for each implemented attribute in the group. It shall not respond for unimplemented attributes in the group. If the value of an attribute is unknown for some reason, the Printer shall respond with the attribute name in the "unknown attribute list" response parameter.

4. If the client supplies an attribute group keyword that is not implemented, the Printer has no means for determining if it is an unimplemented attribute or attribute group. In this case, the Printer assumes that it is an unimplemented attribute and responds as if it is an unimplemented attribute (the Printer responds with the attribute name in the "unknown attribute list" response parameter).

5.1.6 Get-Jobs Operation

The Get-Jobs operation allows a client to retrieve Printer attributes and a list of print jobs belonging to the target Printer object. A list of Job attribute names or attribute group names that the client is interested in seeing may be included in the request.

This operation is like Get-Attributes, except that Get-Jobs operation returns attributes from more than one object.

5.1.6.1 Get-Jobs Request

The client shall submit the Get-Jobs request to a Printer URI.

The following input parameters are part of the Get-Jobs Request:

Job Owner
: This is the user-name. If the value is non-null, then the requester wants only those jobs whose job-originating-owner is the same as the specified user-name. If the value is null, then the requester wants all jobs.

Limit

This is an integer value which indicates a limit to the number of Jobs returned. The limit is a "stateless limit" in that if the

limit is n then only the first n jobs are returned in the Get-Jobs Response; there is no mechanism to allow for the "next" n jobs. The limit applies across all Job States requested. For example, if the limit if 50, and there are 75 jobs in the 'completed' state and 25 in the 'pending state' and the client requests first 'completed jobs' and then 'pending' jobs, only the first 50 'completed' jobs are returned. The other 25 'completed' jobs are not returned and neither are any of the 'pending' jobs returned.

Job States

A possibly empty set of job state values. If the set is not empty, then the requester wants only those jobs whose job-state is the same as one of the specified job state values. If this operation parameter has more than one value, the Printer SHALL return the jobs grouped by state with each group being in the same order as supplied by the client in this parameter. Within each group, the jobs are ordered from oldest to newest with respect to completion time (either actual or expected). For example, if the client requests all 'pending' and 'completed' jobs, first all jobs in the 'pending' state are returned (ordered from oldest to newest) and then all jobs in the 'completed' state are returned (ordered from oldest to newest). If the client request all 'completed' and 'processing' jobs, first all jobs in the 'completed' state are returned (ordered from oldest to newest) and then all jobs in the 'processing' state are returned (oldest to newest). If the client does not supply this operation parameter, the value SHALL be assumed to be (by both the client and the Printer) first 'pending' and then 'processing'.

Requested Job Attributes:

A optional set of attribute names (without values) or attribute groups names in whose values the requester is interested from each of the jobs on the specified Printer. The attribute group names are the same as for the Get-Attributes operation for the Job object. If the client omits this input parameter, the effect shall be the same as if the 'none' attribute group were supplied.

5.1.6.2 Get-Jobs Response

The Printer shall return the following output parameters as part of the Get-Jobs Response:

Result Attributes:

The result includes zero or more objects each with zero or more attributes. Each Job is returned in chronological order. This

order is explicitly defined to be: oldest to newest with respect to completion time, either actual or expected.

If the client did not supply any Job attributes, the Printer shall assume that the client is implicitly requesting the 'none' group (that is no Job attributes are returned, just the Job URI for each Job).

Status
Status information including error status

A Printer may choose, for security reasons, not to return all attributes that a client requests. It may even return none of the requested attributes. In such cases, the status returned is the same as if the Printer had returned all requested attributes. The client cannot tell by such a response whether the requested attribute was present or absent on the Printer.

5.2 Operation Status and Messages

The Status code provides information on the results of a request. The Message provides a short textual description of the Status. The Status is intended for use by automata and the Message is intended for the human user. An IPP application (i.e. a browser, GUI, print driver or gateway) is not required to examine or display the Message.

5.3 Status Codes (type2 keyword)

Each Status is described below, including a description of which operation(s) it can follow and any meta-information required in the response.

ISSUE: Keith's doc still need to go here.

6. Object Attributes

This section describes the attributes with their corresponding 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:

- ISO/IEC 10175 DPA (Final, June 1996)
- RFC 1759 Printer MIB (Proposed Standard, May 1995)
- Internet-Draft: Printer MIB (Draft Standard in progress, December

1996)
- Internet-Draft: Job Monitoring MIB (I-D in progress, March 1997)

Each attribute is uniquely identified in this document using a "keyword" in the section header describing that attribute. A keyword is a sequence of characters (length of 1 to 255) which consists of just letters, digits, hyphen ("-"), and underscore ("_"). With these restrictions, there will be a straight forward encoding of these keywords onto real values in the protocol specification. Not only are attributes uniquely identified with keywords, some attributes take on a syntax which is a set of keywords. This set of keywords represents the domain of the attribute.

6.1 Attribute Syntaxes

The following table shows the basic syntax types that a client and server shall be able to handle.

Table 1 - Attribute Syntaxes

Basic Type	Description	Comments
text	a sequence of characters; length: 0 to 4095; characters: any	For free form human readable text intended for human consumption.
name	a sequence of characters; length: 1 to 255; characters: any	For referencing some object via a user-friendly string, such as a Printer name, a document name, a user name, or a host name. May include the SPACE character. Note: The protocol may need to provide means to quote the SPACE character if the protocol uses SPACE for other purposes.
fileName	a sequence of	For referencing some

	characters; length: 1 to 1024;	file. The limit is the same as POSIX

Basic Type	Description	Comments
	characters: any	and NT.
keyword	a sequence of characters; length: 1 to 255; characters: letters, digits, hyphen ("-"), underscore ("_")	identifiers of entities in the abstract protocol (specified in this document). These entities can be For semantic attribute names or values of attributes, When a keyword is used to represent an attribute (its name), it must be unique within the full scope of 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, a keyword can not be used for two different values of the same attribute to mean two different semantic ideas. However, the same keyword can be used across two or more attributes, representing different semantic ideas for each attribute.

uri	a sequence of characters as defined in rfc1738 and	Universal Resource Identifier

Basic Type	Description	Comments
	rfc1808	
uriScheme	a sequence of characters representing the URI Scheme	"http" for HTTP schemed URIs (e.g., http://...). "ftp" for FTP schemed URIs (e.g., ftp://...).
locale	a standard identifier for language and character set	ISSUE: What standard values will be used for locale?
octetString	a sequence of octets	Used for opaque data, such as the document-content.
boolean	two values of 'true' and 'false'	like an keywordSet, but there are only two values. Note: An application might use a checkbox for such a value.
integer	an integer value that is in the range from -2**31 to 2**31 - 1	each attribute specifies the range constraint explicitly.
dateTime	a value that holds the date and time to the nearest second.	absolute date and time
seconds	a non-negative integer with implicit units of seconds	a relative time
milliseconds	a non-negative interger with	a relative time

implicit units
of milliseconds

Basic Type	Description	Comments
integerUnits	an integer with explicit units	an integer value expressed in units ISSUE: we have two types with implicit units and one with explicit units where the units are specific for one attribute: "printer-speed".
1setOf X	1 or more values of type X.	for sets of values
rangeOf X	a range of value of type X	for a range of values, such as integers

6.1.1 Attribute Extensibility

This document uses prefixes to the keyword basic syntax type in order to communicate extra information to the reader through its name. This extra information need not be represented in an implementation because it is unimportant to a client or Printer. The table below describes the prefixes and their meaning.

ISSUE: There are some references to the Printer Working Group (PWG). How do we describe in this document the process for the PWG to approve type 2 extensions?

Basic Type	Prefix	Comments
keyword	type1	someone must revise the IPP standard to add a new name. No private names are allowed.
keyword	type2	implementers can, at any time, add new

values by proposing them to the PWG
for registration (or an IANA-appointed
registry advisor after the PWG is no

Basic Type	Prefix	Comments
		longer certified) where they are reviewed for approval. IANA keeps the registry. Implementers can support private (unregistered) with a suitable distinguishing prefix, such as -xxx- where xxx is the company name registered with IANA for use in domain names.
keyword	type3	implementers can, at any time, add new values by submitting a registration request directly to IANA, no PWG or IANA-appointed registry advisor review is required. Implementers can support private (unregistered) names with a suitable distinguishing prefix, such as -xxx- where xxx is the company name registered with IANA for use in domain names.
keyword	type4	system administrators can, at any time, add new installation-defined names to a local system. Care should be taken by the administrator to see that keywords do not conflict with other keywords defined by the standard or as defined by the implementing product. There is no registration or approval procedure for type 4 keywords. ISSUE: Since the standard specifies some of the type 4 values, shouldn't it be possible to register additional type 4 values after the standard is approved?

Note: This standard defines keyword values for all of the above types.

6.2 Job Template Attributes

Job Template attributes describe job processing behavior. Take for example, a generic Job Template attribute called "xxx":

1. "xxx" is optionally supplied by the client in a create request. If "xxx" is supplied, the client is specifying that the Printer will apply a specific job processing behavior to this job while processing the Job. When "xxx" is not supplied, the client expects the Printer will apply the default job processing behavior.

2. "xxx-supported" is a Printer attribute that describes what is behaviors are supported by that Printer. "xxx-supported" is a CONDITIONALLY MANDATORY attribute which means that the Printer only implements the attribute if it is capable of realizing one or more of the behaviors associated with the attribute and its values. A client can query the Printer and find out what behaviors are supported by inspecting at the values in the "xxx-supported" attribute.

3. The Printer also implements a default value attribute named "xxx". This default value attribute describes what will be done 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 job data). Along with the supported attribute, the default value attribute is also CONDITIONALLY MANDATORY. However, if the Printer implements the "xxx-supported" attribute, the Printer MUST implement the corresponding default value attribute and vice versa.

4. The Printer OPTIONALLY implements the "xxx-available" attribute. This attribute describes a state of readiness for each supported attribute. The supported and available attributes have the same number of attributes and are ordered so that there is a set of ordered value pairs between the two attributes. The availability state of a supported attribute value indicates the level of effort required by the Printer to actually use resource indicated by the supported value. The following values are used in the "xxx-available" attributes:

   'ready' - the resource can be used by a Job without human intervention (the resource is not exhausted)
   'not-ready' - the use of the resource requires human intervention (the resource may be exhausted or not automatically available)

   Note: The "xxx-available" attribute only applies to the "media" and "finishings" attributes.

5. If a client application wishes to present an end user with a list of supported and default values from which to choose, the client

program should query the supported, default values, and possibly the available attributes. The values that the client then sends in the create request will all fall within the supported values at the

Printer. When querying the Printer, the client MAY enumerate each attribute by name in the Get-Attributes Request, or the client MAY just name the "printer-job-template" group in order to get the complete set of supported, default value, and available attributes which are implemented.

The "job-priority" attribute is an example of a Job Template attribute. It is an integer in the range from 1 to 100. A client can query the Printer for the "job-priority-supported" attribute and the "job-priority" default value attribute. The supported attribute contains a set of supported priority values (a range). The default value attribute contains job priority value that will be used for a new job if the client does not supply one in the create request. If the client does supply the "job-priority" attribute, the Printer validates the value to make sure that it falls within the range of supported values. If the client-supplied value is supported, the Job object is created and the "job-priority" attribute is populated with that value. The Job object, when queried, returns the value supplied by the client. If the client does not supply a "job-priority" value in the create request, the Job object is created, but no "job-priority" attribute is implemented for the Job. The client queries the Printer's default value "job-priority" value to find out at what priority the job will be processed.

The following table summarize the names, relationships, and conformance requirements for all Job Template attributes. The following general rules apply to implementation requirements:

1. In a create request, all Job Template attributes are OPTIONAL.

2. In a Printer Object, all supported attributes are CONDITIONALLY MANDATORY.

3. All Printer default value attributes are CONDITIONALLY MANDATORY. However, if the supported attribute is implemented then the default value attribute MUST be implemented and vice versa.

4. All "available" attributes are OPTIONAL.

The table only shows exceptions to the above rules. The first column of the table (Job) shows the name and syntax for each Job Template

attribute in the Job object (in the create request, the same name and syntax is used). The next three columns show the name and syntax for each Job Template attribute in the Printer object (the default value attribute, the supported attribute, and the available attribute). A "No" in the table means the Printer SHALL NOT implement the attribute.

Job	Printer Default Value	Printer Supported	Printer Available
job-name (name, MAN))	No	No	No
job-sheets (type4 keyword)	job-sheets (type4 keyword)	job-sheets-supported (1setOf type4 keyword)	No
notificaiton-events (1setOf type2 keyword)	notification-events (1setOf type2 keyword)	events (1setOf type2 notification-keyword)	No
notification-addresses (1setOf uri)	No	notification-addresses-supported (1setOf uri scheme)	No
job-priority (int)	job-priority (int)	job-priority-supported (rangeOf int)	No
job-hold-until (type4 keyword)	job-hold-until (type4 keyword)	job-hold-until-supported (1setOf type4 keyword)	No

multiple-documents-are (type2 keyword)	multiple-documents-are (type2 keyword)	multiple-documents-are-supported (1setOf type2	No

Job	Printer Default Value	Printer Supported	Printer Available
		keyword)	
best-effort (type2 keyword)	best-effort (type2 keyword, MAN)	best-effort-supported (1setOf type2 keyword, MAN)	No
media (type4 keyword)	media (type4 keyword)	media-supported (1setOf type2 keyword)	media-available (1setOf avail keyword)
number-up (type3 keyword)	number-up (type3 keyword)	number-up-supported (1setOf type3 keyword)	
sides (type2 keyword)	sides (type2 keyword)	sides-supported (1setOf type2 keyword)	
printer-resolution (type2 keyword)	printer-resolution (type2 keyword)	printer-resolution-supported (1setOf type2 keyword)	
print-quality (type2 keyword)	print-quality (type2 keyword)	print-quality -supported (1setOf type2 keyword)	
finishings (setOf type2 keyword)	finishings (setOf type2 keyword)	finishings-supported (setOf type2 keyword)	finishings-available (setOf avail keyword)
copies	copies	copies-supported	No

```
(int)             (int)           (rangeOf int)
```

Job	Printer Default Value	Printer Supported	Printer Available
compression (type3 keyword)	compression (type3 keyword)	compression-supported (1setOf type3 keyword)	No
job-k-octets (int)	No	job-k-octets-supported (rangeOf int)	No
job-impressions (int)	No	job-impressions-supported (rangeOf int)	No
job-media-sheets (int)	No	job-media-sheets-supported (rangeOf int)	No

6.2.1 job-name (name)

This attribute defines the name of the job. It is a name that is more user friendly than the job-URI.

If "job-name" is not supplied in the in the create request, the Printer, on creation of the Job, shall generate a name which is the name of the first document in the job. This name comes from the "document-name" attribute or "document-URI" attribute depending on which attribute is supplied in the Create-Job Request for the first document. If "job-

name" is supplied in the Create-Job Request, the Printer uses its value as the name of the created Job.

6.2.2 job-sheets (type4 keyword)

This attribute determines which of any banner page(s) shall be printed with a job.

Standard values are:

'none': no job sheet is printed
'standard': a site specific standard job sheet is printed
extensions: names the specific job sheet (banner page)

To force no job sheets, the system administrator SHALL set the only supported value to 'none'.. To force the use of banner pages, the supported values shall not include 'none'. If a client requests 'none' in the create request, the request is rejected.

6.2.3 notification-events (1setOf type2 keyword)

This attribute specifies the events for which the end user desires some sort of notification. The "notification-addresses" attribute is used to describe the destination addresses for these events.

Standard values are:

'none': the Printer shall not notify
'all': the Printer shall notify when any of event occurs.
'job-completion': the Printer shall notify when the job containing this attribute completes with or without errors.
'job-canceled': the Printer shall notify when the job containing this attribute is canceled by the end-user or by the operator, or aborts before completion.
'job-problems': the Printer shall notify when this job has a problem while this job is printing. Problems include any of the "job-state-reasons" or "printer-state-reason" values
'printer-problems': the Printer shall notify when any job, including this job, is affected by a Printer problem (the printer has moved to the stopped state and there is a reason in the printer-state-reasons attribute) while this job is waiting to print or printing. Problems include any of the "job-state-reasons" or "printer-state-reason" values

6.2.4 notification-addresses (1setOf uri)

This attribute describes both where (the address) and how (the mechanism for delivery) notification events are to be delivered. The Printer

shall use this attribute as the set of addresses and methods for sending messages when an event occurs that the end user (job submitter) has registered an interest in.

Standard uri scheme values are:

'mailto': email is used
'http': an HTTP method is used to add HTML formatted events to the end of the specified HTML file.
'ftp': FTP is used to append a record at the end of a specified text file.

6.2.5 job-priority (integer(1:100))

This attribute specifies a priority for scheduling the print-job. A higher value specifies a higher priority. The value 1 is defined to indicate the lowest possible priority. The value 100 is defined to indicate the highest possible priority. Priority is expected to be evenly or "normally" distributed across this range. Among those jobs that are ready to print, a Printer shall print all jobs with a priority value of n before printing those with a priority value of n-1 for all n. The mapping of vendor-defined priority over this range is implementation-specific.

6.2.6 job-hold-until (type4 keyword)

This job attribute specifies the named time period during which the Job print job shall become a candidate for printing.

Standard values for named time periods are:

'no-hold': immediately, if there are not other reasons to hold the job.
'day-time': during the day.
'evening': evening
'night': night
'weekend': weekend (Saturday or Sunday)
'second-shift': second-shift
'third-shift': third-shift (after midnight)

An administrator shall associate allowable print times with a named time period (by means outside IPP 1.0). An administrator is encouraged to pick names that suggest the type of time period.

If the value of this attribute specifies a time period that is in the future, the Printer shall add the 'job-hold-until-specified' value to the job's "job-state-reasons" attribute and shall not schedule the print-job for printing until the specified time-period arrives. When

the specified time period arrives, the Printer shall remove the 'job-hold-until-specified' value from the job's "job-state-reason attribute"

and, if no other reasons remain, shall consider the job as a candidate for processing.

If this job attribute value is the named value "'no-hold'", or the time period has already started , the job shall be a candidate for processing immediately.

6.2.7 multiple-documents-are (type2 keyword)

This job attribute is relevant only if a job consists of two or more documents. It controls finishing operations, job-sheet placement, and the order of documents when the copies attribute exceeds 1.

Standard values are:

'single-document': If the files for the job are a and b, then files a and b are treated as a single document for finishing operations. Also, there will be no slip sheets between files a and b. If more than one copy is made, the ordering shall be a, b, a, b, ...

'separate-documents-uncollated-copies': If the files for the job are a and b, then each file is treated as a single document for finishing operations. Also, a client may specify that a slip sheet be placed between files a and b. If more than one copy is made, the ordering shall be a, a, b, b, ...

'separate-documents-collated-copies': If the files for the job are a and b, then each file is treated as a single document for finishing operations. Also, a client may specify that a slip sheet be placed between files a and b. If more than one copy is made, the ordering shall be a, b, a, b, ...

Both of the 'separate-xxx' values force each new document to start on a new media sheet.

6.2.8 best-effort (type2 keyword)

This attribute determines what to do if there is a conflict between what a client requests and what a Printer supports.

Standard values for this attribute are:

- 'shall-honor-ipp-attributes': If a Printer supports this value and a client requests this value, the Printer guarantees that all IPP attribute values take precedence over embedded instructions in the job data (the PDL of the job's documents).

- 'should-honor-ipp-attributes': If a Printer supports this value, and a client requests this value, the Printer should try to make

sure that IPP attribute values take precedence over embedded PDL instructions, however there is no guarantee

The client supplies Job Template attributes in the create request to 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, within the Page Description Language (PDL) of the document data. If there is a conflict between the value of one of these attributes, and a corresponding instruction in the document (either implicit or explicit), it is desireable that the value of the attribute shall take precedence over the document instruction. Many of these job template attributes provides a client with a way to request some feature at print time that might not have been embedded within the document data when the document was created. Job template attribute also provides a client with a way to override a feature at print time that was embedded within the document data when the document was created. Note: until companies that supply interpreters for PDLs, such as PostScript and PCL allow a way to specify overrides for internal job production instructions, a Printer might not be able to implement these attributes for some PDLs.

In order to solve this problem, the IPP Printer implements the MANDATORY best-effort-supported" attribute. If the value of this attribute is 'shall-honor-ipp-attributes', the implementation MUST guarantee that the IPP attribute values take precedence over any related job processing instructions in the PDL Job's document data. This can be done by modifying the interpreter within the output device itself to understand IPP attributes, or by mergeing theses Job Template attributes directly into the document data, or in any other implementation specific manner. In any case, the semantics of 'shall-honor-ipp-attributes' MUST be preserved.

Note: Since 'should-honor-ipp-attributes' does not offer any type of guarantee, a Printer may not do a very "good" job of implementing the semantics of "should", but it would still be a conforming implementation.

This "best-effort" attribute has nothing to do with conflict between what a Printer supports and what an IPP client requests. If there is such a conflict, the Printer shall reject the create request. A client SHOULD query the printer to find out what is supported before supplying specific values in a create request.

6.2.9 media (type4 keyword)

This job attribute identifies the medium that the Printer shall use for all pages of the document regardless of what media are specified within the document.

The values for medium include medium-names, medium-sizes, input-trays and electronic forms so that one attribute specifies the media. If a printer allows a client to specify a medium name as the value of this attribute, such a medium name implicitly selects an input-tray that contains the specified medium. If a printer allows a client to specify a medium size as the value of this attribute, such a medium size implicitly selects a medium name which in turn implicitly selects an input-tray that contains the medium with the specified size. If a printer allows a client to specify 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-freed input-trays. If a printer allows a client to specify an electronic form as the value of this attribute, such an electronic form implicitly selects a medium-name which 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 shall merge with the data from the document as its prints each page.

Standard values are (taken from ISO DPA and the Printer MIB):

default:
iso-a4-white:
...

6.2.10 number-up (type3 keyword)

This job attribute specifies the number of source page-images to impose upon a single side of an instance of a selected medium.

Standard values are:

'none': The Printer shall not include any embellishments and shall place one logical page on a single side of an instance of the selected medium without any translation, scaling, or rotation.
'one': The Printer shall place one logical page on a single side of an instance of the selected medium but is allowed to add embellishments or some sort of translation, scaling, or rotation.
'two':
'four':

This attribute primarily controls the translation, scaling and rotation of page images, but a site may choose to add embellishments, such as borders to each logical page.

6.2.11 sides (type2 keyword)

This attribute specifies how source page-images are to be imposed upon the sides of an instance of a selected medium.

The standard values are:

'one-sided': imposes each consecutive source page-image upon the same side of consecutive media sheets.
'two-sided-long-edge': imposes each consecutive pair of source page-image upon front and back sides of consecutive media sheets, such that the orientation of each pair of source-pages on the medium would be correct for the reader as if for binding on the long edge. This imposition is sometimes called "duplex".
'two-sided-short-edge': imposes each consecutive pair of source page-image upon front and back sides of consecutive media sheets, such that the orientation of each pair of source-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' and 'two-sided-short-edge' work the same for portrait or landscape. That is, "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.

6.2.12 printer-resolution (type2 keyword)

This job attribute specifies the resolution that the Printer should use.

The values are type2 keywords which represent single integers or pair of integers. The latter are to specify the resolution when the x and y dimensions differ. When two integers are specified, the first is in the x direction, i.e., in the direction of the shortest dimension of the medium, so that the value is independent of whether the printer feeds long edge or short edge first.

The standard values are:

normal:
res-100:
res-300x300:
...

6.2.13 print-quality (type2 keyword)

This job attribute specifies the print quality that the Printer should use.

The standard values are:

  draft: lowest quality available on the printer

normal: normal or intermediate quality on the printer
high: highest quality available on the printer

6.2.14 copies (integer(1:2**31 - 1))

This job attribute specifies the number of copies of the job to be printed. Note: The effect of this attribute on jobs and documents is controlled by the multiple-files-are job attribute.

6.2.15 finishings (1setOf type2 keyword)

This job attribute identifies the finishing operation that the Printer should apply to each copy of each printed document in the job where the definition of a copy is controlled by the "multiple-documents-are" Job attributes.

Standard values are:

none:
staple:
...

6.2.16 compression (type3 keyword)

This attribute identifies compression algorithms used for compressed document data.

Standard values for this attribute are:

'none':
'zip':
'tar':
...

6.2.17 job-k-octets (integer(0:2**31 - 1))

This Job attribute specifies the total size of the job in K octets, i.e., in units of 1024 octets. The value shall be rounded up, so that a job between 1 and 1024 octets shall be indicated as being 1K, 1025 to 2048 shall be 2, etc. This attribute is not intended to be a counter as

in the Job Monitoring MIB; it is intended to be useful routing and scheduling information if known. The Printer might not be able to

compute this value (if not supplied by the client in the request) at the time the Job is created. If not, the Printer may implement this attribute at any later time as it is able to compute the total size of the Job.

6.2.18 job-impressions (integer(0:2**31 - 1))

This job attribute specifies the total size of the job in impressions. This attribute is not intended to be a counter as in the Job Monitoring MIB; it is intended to be useful routing and scheduling information if known. The Printer shall try to compute the value if it is not supplied in the create request. The Printer might not be able to compute this value (if not supplied by the client in the request) at the time the Job is created. If not, the Printer may implement this attribute at any later time as it is able to compute the total size of the Job.

6.2.19 job- media-sheets (integer(0:2**31 - 1))

This job attribute specifies the total size of the job in media-sheets. This attribute is not intended to be a counter as in the Job Monitoring MIB; it is intended to be useful routing and scheduling information if known. The Printer shall try to compute the value if it is not supplied in the Create-Job Request. The Printer might not be able to compute this value (if not supplied by the client in the request) at the time the Job is created. If not, the Printer may implement this attribute at any later time as it is able to compute the total size of the Job.

6.3 Job Attributes

The attributes in this section are associated with a Job.

6.3.1 Job Template Attributes

The Job Template attributes which apply specifically to a Job object are described in section 6.2 Job Template Attributes. These Job specific attributes form the attribute group called "job-template".

6.3.2 Job Description Attributes

These attributes form the attribute group called "job-description".

Attribute	Man ?

job-URI (uri)	Man
job-originating-user (name)	Man
job-originating-host (name)	
user-locale (type2 keyword)	
job-state (type1 keyword)	Man
job-state-reasons (1setOf type2 keyword)	Man
job-state-message (text)	
output-device-assigned (name)	
time-since-submission (milliseconds)	Man
number-of-intervening-jobs (int)	Man
job-message-from-operator (text)	
time-since-completion (milliseconds)	Man
job-k-octets-	

```
completed
(int)
```

job-impressions-
completed
(int)

job-media-sheets-
completed
(int)

6.3.2.1 job-URI (uri)

This attribute contains the URI for the job. The Printer, on receipt of a new job, shall generate a URI which identifies the job on the Printer. The Printer, shall return the value of the URI job attribute as part of the PrintResult in the Print operation. The precise format of a job URI shall be implementation dependent.

6.3.2.2 job-originating-user (name)

This attribute specifies the user name of the person submitting the print job. The Printer shall set this attribute to the most authentic name that it can obtain from the protocol over which the operation was received from the client.

6.3.2.3 job-originating-host (name)

This attribute identifies the originating host of the job. The Printer shall set this attribute to the most authentic host name it can obtain from the protocol over which the operation was received from the client.

6.3.2.4 user-locale (type3 keyword)

This attribute identifies the locale of the job, i.e, the country, language, and coded character set. The Printer sets this attribute to the most authentic value it can obtain from the protocol over which the Print operation was received from the client.

The Printer shall use this attribute to determine the locale for notification messages that it sends.

6.3.2.5 job-state (type1 keyword)

This attribute identifies the current state of the job.

The Printer may provide additional information about each state value by supplying one or more values of the job's companion "job-state-reasons" attribute depending on implementation. While the job states cannot be added to 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 without impacting such deployed clients. In other words, the "job-state-reasons" attribute is intended to be extensible.

Standard values are:

unknown	The job state is not known, or its state is indeterminate.
pending	The job is a candidate to start processing, but is not yet processing..
pending-stopped	The job is not a candidate for processing for any number of reasons and will resume pending as soon as the reasons are no longer present. The job's "job-state-reason" attribute shall indicate why the job is no longer a candidate for processing.

processing Either:
1. the job is using, or is attempting to use, one or more document transforms which include (1) purely software processes that are interpreting a PDL, and (2) hardware devices that are interpreting a PDL, making marks on a medium, and/or performing finishing, such as stapling
OR
2. the server 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.
ISSUE: The "job-state-reasons" specification is that the job is in 'pending' state not 'processing' state.
When the job is in the 'processing' state, the entire job state includes the detailed status represented in the printer's "printer-state", "printer-state-reasons", and "printer-state-message attributes.
Implementations may include additional values in the job's "job-state-reasons" attribute to indicate the progress of the job, such as adding the 'job-printing' value to indicate when the output device is actually making marks on paper. Most implementations won't bother with this nuance.

processing -stopped	The job has stopped while processing for any number of reasons and will continue processing as soon as the reasons are no longer present. The job's "job-state-reason" attribute shall indicate why the job has stopped processing. If the output device is stopped, the 'printer-stopped' value shall be included in the job's "job-state-reasons" attribute. When the output device is stopped, the device usually indicates its condition in human readable form locally at the device. A client can obtain more complete device status remotely by querying the printer's "printer-state-reasons" attribute.
canceled	The job has been canceled by a Cancel-Job operation and is either (1) in the process of terminating or (2) has completed terminating. The job's "job-state-reasons" attribute shall contain either the 'canceled-by-user' or 'canceled-by-operator' value.
aborted	The job has been aborted by the system, usually while the job was in the 'processing' state.
completed	The job has completed successfully or with warnings or errors and all of the job media sheets have been properly stacked in the appropriate output tray or bin. The job's "job-state-reasons" attribute shall contain one of: 'completed-successfully', 'completed-with-warnings', or 'completed-with-errors'.

The length of time that jobs remain in the 'canceled', 'aborted', and 'completed' states depends on implementation.

The following figure shows the normal job state transitions. Other transitions are unlikely, but are not forbidden.

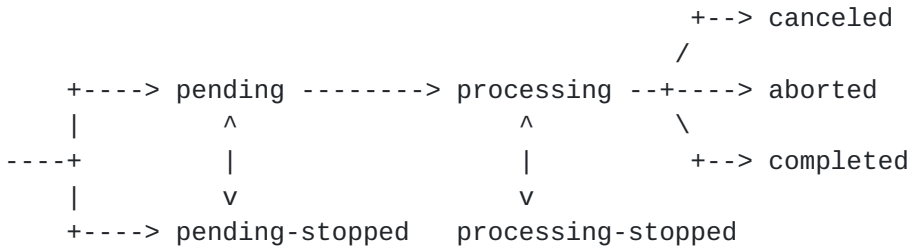


Figure 1 - Normal job state transitions

Normally a job progresses only from left to right through three states.

Even though the IPP protocol defines seven values for job states, Printers SHALL only implement those states which are appropriate for the particular implementation.

6.3.2.6 **job-state-reasons (1setOf** type2 keyword)

This attribute identifies the reason or reasons that the job is in the state that it is in (e.g., 'pending', 'processing', 'completed', etc.). The Printer shall indicate the particular reason(s) by setting the value of the job's "job-state-reasons" attribute.

The following standard values are defined for use with the job states indicated as applicable.

NOTE - For easy of understanding the order of the reasons is presented in the normal order of job state progression:

none	Mandatory: There are no reasons for the job's current state. Applicable job states: 'pending', 'processing', 'aborted'.
job-incoming	Mandatory. The PrintJob or CreateJob operation has been accepted by the Printer, but the Printer is waiting for additional SendDocument operations and/or the transfer of the remainder of the job or document data. Applicable job states: 'pending-stopped'
job-outgoing	Optional. The Printer is transmitting the job to the

output device.
Applicable job states: 'pending'

job-hold-until-specified	Conditionally mandatory. Mandatory if the "job-hold-until" job attribute is implemented. The value of the job's "job-hold-until" attribute has specified a time period that has not yet arrived, so that the Printer shall not consider the job as a candidate for processing. Applicable job states: 'pending-stopped'.
job-hold-until-resources-are-ready	Optional. 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. Usually such a condition is detected while the job is in the 'pending' state. If a job starts processing and encounters a resource that is not ready, there are two possible implementations: (1) the device is stopped and no jobs can run until the resource(s) are made ready, in which case the Printer shall keep the job in the 'processing' state and shall add the 'printer-stopped' reason to the job's "job-state-reasons" attribute (not the 'job-hold-until-resources-are-ready' value) OR (2) another job is found to use the device while the current job goes back to waiting for its turn and for the resources to be made ready, in which case the Printer shall change the job's "job-state" attribute to 'pending' and add the 'job-hold-until-resources-are-ready' value to the job's "job-state-reasons" attribute.

Applicable job states: 'pending-
stopped'.

printer-stopped-partly	Optional. This reason appears in all jobs in the 'pending' state when the value of the Printer's "printer-state-reasons" attribute contains the value 'stopped-partly'. Applicable job states: 'pending'
printer-stopped	Mandatory. The output device is stopped. This reason appears in all jobs in the 'pending' and 'processing' states when the value of the Printer's "printer-state" attribute is 'stopped'. Applicable job states: 'pending-stopped', 'processing-stopped'
job-printing	Optional. The output device is marking media. This value is useful for Printers which spend a great deal of time processing when no marking is happening and then want to show that marking is now happening. Applicable job states: 'processing', 'processing-stopped'
job-cancelled-by-user	Level 2 Mandatory. The job was cancelled by the user using the CancelJob request. Applicable job states: 'canceled'.
job-cancelled-by-operator	Level 2 Mandatory. The job was cancelled by the operator using the CancelJob request. Applicable job states: 'canceled'.
logfile-pending	Optional. The job's logfile is pending file transfer. Applicable job states: 'canceled', 'aborted', and 'completed'.
logfile-transferring	Optional. The job's logfile is being transferred. Applicable job states: 'canceled', 'aborted', and

'completed'.

job-completed-successfully	Mandatory. The job completed successfully. Applicable job states: 'completed'.
job-completed-with-warnings	Mandatory. The job completed with warnings. Applicable job states: 'completed'.
job-completed-with-errors	Mandatory. The job completed with errors (and possibly warnings too). Applicable job states: 'completed'.

6.3.2.7 job-state-message (text)

This attributes specifies suplimental information about the Job State in human readable text. It SHALL be set by the Printer.

6.3.2.8 output-device-assigned (uri)

This attribute identifies the Output Device to which the Printer has assigned this job. If an output device implements an embedded IPP Printer, the Printer NEED NOT set this attribute. If a Print Server implements a Printer, the value MAY be empty until the Printer assigns an output device to the job.

ISSUE: Why is this a uri?

6.3.2.9 time-since-submission (milliseconds)

This attribute indicates the amount of time that has passed since the Job was first created.

6.3.2.10 time-since-processing (milliseconds)

This attribute indicates the amount of time that has passed since the Job first entered the processing state.

6.3.2.11 number-of-intervening-jobs (integer(0:2**31 - 1))

This attribute indicates the number of jobs that are "ahead" of this job in the current scheduled order. For efficiency, it is only necessary to

calculate this value when an operation if performed that requests this attribute.

Note: This attribute is necessary since an end user may request just their own jobs and they need some relative position indicator if there are other jobs interspersed in the waiting list which are not returned in the response or cannot be because of site security policy restrictions.

6.3.2.12 job-message-from-operator (text)

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.

6.3.2.13 time-since-completion (milliseconds)

This attribute indicates the amount of time that has passed since the Job was first created.

6.3.2.14 job-k-octets-completed (integer(0:2**31 - 1))

This attribute specifies the number of octets completed in K octets, i.e., in units of 1024 octets. The value shall be rounded up, so that a job between 1 and 1024 octets shall be indicated as being 1K, 1025 to 2048 **shall be 2, etc. This attribute is intended to be a counter as in** the Job Monitoring MIB.

6.3.2.15 job-impressions-completed (integer(0:2**31 - 1))

This job attribute specifies the number of impressions completed. This attribute is intended to be a counter as in the Job Monitoring MIB.

6.3.2.16 job-media-sheets-completed (integer(0:2**31 - 1))

This job attribute specifies the media-sheets completed. This attribute is intended to be a counter as in the Job Monitoring MIB.

6.4 Document Attributes

This group of attributes describes the document data for the job. They are supplied in the Send-Document request.

Attribute	Man ?
document-name (name)	Man
document-format (type2 keyword	
document-URI (uri)	

6.4.1 document-name(name, Mandatory)

This attribute contains the name of the document used by the client to initially identify the document. When a client prints by reference, i.e. includes the document-URI attribute and no document content, this attribute shall be absent. When a document's contents are spread across multiple Send Document operations "document-name" is ignored by the Printer in all but the first Send-Document operation.

6.4.2 document-format (type2 keyword)

This attribute defines the document format of this document. When a client prints by reference, i.e. includes the document-URI attribute and no document content, this attribute shall be absent. When a document's contents are spread across multiple Send-Document operations, document-format is ignored by the Printer in all but the first Send Document operation for that document.

This attribute identifies the document format of this document. One possible supported and default value is 'auto-sense'. However, 'auto-sense' shall not be sent in the Send-Document Request.

The following standard values have been reviewed with the Printer Working Group and are registered with IANA as part of the IETF Printer MIB project. The standard value assigned by the PWG starts with the four letters: "lang", in order to follow SNMP ASN.1 rules that all enum symbols shall start with a lower case letter. The keyword values in IPP

shall be the same as the PWG standard values registered with IANA with the "lang" removed. The MIB (integer) value is included here for

reference only, the MIB integer value shall not be used in IPP; the keyword value shall be used instead:

Note: In the protocol specification [22], these keywords will be encoded as MIME types.

6.4.3 document-URI (uri)

This attribute contains the URI of the document when the document content is not included in the Send Document operation. Document-number is the only other attribute allowed when a document-URI attribute is present in a Send Document operation.

6.5 Printer Attributes

The attributes in this section are associated with a Printer object.

6.5.1 Printer Job Template Attributes

The Job Template attributes which apply specifically to a Printer object are described in section 6.2 Job Template Attributes.

These Printer attributes form the attribute group named "printer-job-template".

6.5.2 Printer Description Attributes

These attributes form the attribute group called "printer-description".

A Printer object may be realized in either a print server or output device. Note: How these attributes are set by an Administrator is outside the scope of this specification.

Attribute	Man?
printer-URI (uri)	Man
printer-name (name)	Man

printer-location (text)	
printer-description (text)	
printer-more-info-site (uri)	
printer-driver-installer (uri)	
printer-make-and-model (text)	
maximum-printer-speed (integerUnits)	
printer-more-info-manf (uri)	
printer-state (type1 keyword)	Man
printer-state-reasons (type2 keyword)	Man
printer-is-accepting-job (boolean)s	Man
printer-state-message (text)	
queued-job-count (int)	

printer-message-
from-operator

(text)

printer-locale Man
(locale)

printer-locales- Man
supported
(1setOf locale)

multiple-documents-
are-supported
(boolean)

6.5.2.1 printer-URI (uri)

This attribute contains the URI for the printer. An administrator shall determine a printer's URI and shall set this attribute to that URI. The precise format of a printer URI shall be implementation dependent.

6.5.2.2 printer-name (name)

This attribute contains the name of the printer. It is a name that is more user friendly than the printer-URI. An administrator shall determine a printer's name and shall set 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.

6.5.2.3 printer-location (text)

This attribute identifies the location of this printer.

6.5.2.4 printer-description (text)

This attribute identifies the descriptive information about the this Printer. This could include things like: "This printer can be used for printing color transparencies for HR presentations", or "Out of courtesy for others, please print only small (1-5 page) jobs at this printer", or even "This printer is going away on July 1, 1997, please find a new printer".

6.5.2.5 printer-more-info-site (uri)

This attribute contains a URI used to obtain more information about this specific printer. 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 services (e.g. job pricing, services offered, end user assistance). The manufacturer may initially populate this attribute.

6.5.2.6 printer-driver-installer (uri)

This attribute contains a URI to use to locate the driver installer for this printer. This attribute is intended for consumption by automata. The mechanics of print driver installation is outside the scope of IPP. The manufacturer may initially populate this attribute.

6.5.2.7 printer-make-and-model (text)

This attribute identifies the make and model of the printer.

ISSUE: Several comments that we should break this up into two attributes. It is just a text string, so it could really be anything.

6.5.2.8 maximum-printer-speed (integerUnits)

This attribute indicates the maximum printer speed of the Printer in units of pages per minute, impressions per minute, lines per minute, and characters per second. A job cannot control a Printer's speed, but a Printer Browser can use printer speed as a criteria.

The standard units are a type2 setOf keyword : ppm, ipm, spm, lpm, and cps. These mean pages per minute, impressions per minutes, sides per minutes, lines per minute, and characters-per-second, respectively.

6.5.2.9 printer-more-info-manf (uri)

This attribute contains a URI used to obtain more information about this type of printer. 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). The information is intended to be germane to this printer without regard to site specific modifications or services.

6.5.2.10 printer-state (type1 keyword)

This attribute identifies the current state of the printer. The printer-state reasons attribute augments the printer-state attribute to give more detailed information about the Printer is in the given printer state.

The protocol shall support all values for printer states. A Printer shall continually keep this attribute set to the value in the table below which most accurately reflects the state of the Printer.

The following standard values are defined:

unknown	The Printer state is not known, or is indeterminate. A Printer shall use this state only if it cannot determine its actual state.
idle	If a Printer receives a job (whose required resources are ready) while in this state, such a job shall transit into the processing state immediately. If the printer-state-reasons attribute contains any reasons, they shall be reasons that would not prevent a job from transiting into the processing state immediately, e.g., toner-low. Note: 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.

processing — If a Printer receives a job (whose required resources are ready) while in this state, such a job shall transit into the pending state immediately. Such a job shall transit into the processing state only after jobs ahead of it complete printing.

If the printer-state-reasons attribute contains any reasons, they shall be reasons that do not prevent the current job from printing, e.g. toner-low.

Note: 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.

stopped — If a Printer receives a job (whose required resources are ready) while in this state, such a job shall transit into the pending state immediately. Such a job shall transit into the processing state only after some human fixes the problem that stopped the printer and after jobs ahead of it complete printing.

The printer-state-reasons attribute shall contain at least one reason, e.g. paper-jam, which prevents it from either processing the current job or transiting a pending job to the processing state.

Note: 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.

Note: In the case where a Printer controls more than one output device, 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 idle to processing without human intervention, even though the Printer is stopped.

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

This attribute supplies additional detail about the printer's state.

Each value shall have an adornment to indicate its level of severity. The three levels are: report (least severe), warning and error (most severe).

- 'report': it has the adornment of "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 shall contain nothing that could affect the printed output.
- 'warning': it has the adornment of "warning". An implementation may choose to omit some or all warnings. Warnings serve as a precursor to an error. A warning shall contain nothing that prevents a job from completing, though in some cases the output may be of lower quality.
- 'error': it has no adornment. An implementation shall include all errors. If this attribute contains one or more errors, printer shall be in the stopped state.

ISSUE: How do we indicate report, warning, or error without adornments? Proposal add a parallel attribute to print-state-reasons and name it "printer-state-reasons-severity-level". The ith element of the reasons attribute has the severity level of the ith element of the severity level attribute. Another proposal: an implementation may add 'error-', 'warning-', or 'report-' to any of the reasons to indicate the level of severity.

ISSUE: Toner-low should be a warning because it allows printing to proceed, but in some printers, toner-low may also produce degraded output. Do we want a fourth category, perhaps severe-warning which allows a job to continue printing but with reduced quality?

If a Printer controls more than one output device, each value of this attribute shall apply to one more of the output devices. An error on one output device that does not stop the Printer as a whole appears as a warning in the Printer's printer-state-reasons attribute. Such a Printer's printer-state value may be stopped even with no printer-state-

reasons that are errors.

The following standard values are defined:

stopped-partly	When a Printer controls more than one output device, this reason indicates that one or more output devices are stopped. If the reason is a report, fewer than half of the output devices are stopped. If the reason is a warning, fewer than all of the output devices are stopped.
media-needed	A tray has run out of media
paper-jam	The printer has a paper jam.
paused	Someone has paused the Printer. In this state, a Printer shall not produce printed output, but it shall perform other operations requested by a client. If a Printer had been printing a job when the Printer was paused, the Printer shall 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 from service, and it may be powered down or physical removed. In this state, a Printer shall not produce printed output, and unless the Printer is realized by a print server that is still active, the Printer shall perform no other operations requested by a client, including returning this value. If a Printer 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-printer	The server has scheduled a job on the Printer 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 in the time specified by the printer's printer-timeout-period attribute.
stopping	The printer will be stopping in a while and will change its reason to printer-stopped. This reason is a non-critical, even for a Printer with a single output device. When an output-device ceases accepting jobs, the Printer will have this state while the output device completes printing.

6.5.2.12 printer-is-accepting-jobs (boolean)

This attribute determines whether the printer is currently accepting job. If the value is true, the printer is accepting jobs. If the value is false, the printer is currently rejecting any jobs submitted to it.

Note: This value is independent of the printer state and printer-state-reasons 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 to accepts jobs when the printer-state is stopped.

6.5.2.13 printer-state-message (text)

This attribute specifies the additional informaion about the printer state in human readable text and it shall be set by the Printer (or the

Administrator by some mechanism outside the scope of IPP). When a Printer returns the value of this attribute to a client, the Printer

shall localize the value of this attribute to be in the locale of the user, as specified by the Get-Attributes or Get-Jobs operation.

6.5.2.14 queued-job-count (integer(0:2**31 - 1))

This attribute contains a count of the number of jobs that are either pending and/or processing and shall be set by the Printer.

6.5.2.15 printer-message-from-the-operator (text)

This 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.

6.5.2.16 printer-locale (locale)

This attribute specifies the current locale that the Printer is operating in.

6.5.2.17 printer-locales-supported (1setOf locale)

This attribute specifies the locales that the Printer operates in.

7. Conformance

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

7.1 Conditionally Mandatory

For example, a conditionally mandatory attribute means that a Printer implementation need not implement the attribute if the attribute controls a feature that the output device does not implement or expose. For example, for an output device that can only print on one side, a Printer need not implement the "sides" attribute. For an output device that does not support any of the finishing attribute values, a Printer need not implement the "finishing" attribute. An implementation that does not allow for 'not-ready' supported values in addition to 'ready' values, need not implement the corresponding

"xxx-availability" Printer attribute. For an output device with only a single input tray with only one media type in that tray, a Printer need not implement the "media" attribute.

7.2 Client Conformance Requirements

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 application might not.

When sending a Get-Attributes or Create-Job request, an IPP client need not supply any attributes.

A client shall be able to accept any of the attribute syntaxes defined in Section 0 that may be returned to it in a response from a Printer

ISSUE: Are there any attributes that are mandatory for a client to set in a Print request?

A query response may contain attributes and values that the client does not implement or expect. Therefore, a client implementation must gracefully handle such responses and not refuse to interoperate with a conforming Printer that is returning extended registered or private attributes and/or attribute values that conform to Section 8. IANA Considerations, Registered Extensions, Private Extensions. Some clients may choose to ignore any attributes that it does not implement.

7.3 Printer Object Conformance Requirements

This section specifies the conformance requirements for conforming Printer object implementations with respect to objects, operations, and attributes.

7.3.1 Objects

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

Section 0 4.1 Printer Object
Section 0 4.2 Job Object
Section 0 4.3 Document Object

7.3.2 Operations

Conforming Printer implementations shall implement all of the model operations, including mandatory responses, as defined in this specification in the indicated sections:

Section 0
Section Job 0 5.1.4 Cancel Job Operation
Section 0 5.1.5 Get-Attributes Operation
Section 0 5.1.6 Get-Jobs Operation

7.3.3 Attributes

ISSUE: Some Printer attributes are purely software, so that Conditionally Mandatory doesn't apply, such as Printer Description Attributes. For example, what does it mean for the "printer-location" to be Conditionally Mandatory? Should we add a "OPTIONAL" in the header of the few attributes for which Conditionally Mandatory doesn't make sense?

Conforming Printer implementations shall implement all of the mandatory attributes, as defined in this specification in the indicated sections. Mandatory attributes are indicated as "MANDATORY" in the header of each sub-section of Section 0 that specifies the attribute which is copied into the Table of Contents:

ISSUE: What if this list differs from the headers, which wins? Wouldn't it be safer and more compact to replace the above list with a reference to section 6 attributes that are flagged as MANDATORY in the header line (and automatically copied to the Table of Contents)?

Conforming Printer implementations shall implement all conditionally mandatory attributes in Section 0, i.e., the additional Job and Printer attributes that represent features that the output device implements.

It is not required that a conforming Printer implement or support all values of all implemented attributes. For example, if a Printer supports some of the "finishing" attribute values in this document, it is not required that a Printer implement or support all finishing methods indicated by all the values of the "finishing" attribute contained in this document.

If a Printer implements a "xxx-supported" attribute it must implement the corresponding "xxx" default value attribute and vice versa.

7.3.4 Default Value

If the output device itself does not support the concept of a default

value or the output device's default value is unpredictable, or unknown, the implementation of the Printer shall always supply its default value to the physical device each time the Printer submits

the job to the output device. The default value specified by a Printer shall not be 'unknown'.

7.3.5 Availability

Conforming implementation need not implement the "xxx-available" attributes since if an "xxx-supported" printer attribute does not have have an associated "xxx-available" Printer attribute, all supported values shall be 'ready'.

7.3.6 Printer extensions

A conforming Printer may implement registered extensions and private extensions, as long as they meet the requirements specified in Section 8. IANA Considerations, Registered Extensions, Private Extensions.

7.3.7 Attribute Syntaxes

A Printer shall be able to accept any of the attribute syntaxes defined in Section 0 in any operation in which a client may supply attributes. Furthermore, a Printer shall return attributes to the client in operation responses that conform to the syntax specified in Section 0.

7.4 Security Conformance Requirements

The security mechanisms being considered for IPP fall outside the scope of the application layer protocol itself. There are two mechanisms used to begin secure communications using IPP:

1. Information in the directory entry for an IPP Printer (or from additional information at a Web site hosting the IPP Priner) indicate which, if any, security protocols are used in conjunction with IPP.

2. The URI for the IPP Printer contains the security protocol information (https://..., etc.)

In either case, the security protocol (if any) is initiated first which allows for the negotiation of security features. IPP is then run as an application protocol on top of the security protocols. One cannot "bootstrap" the security features from IPP itself.

8. IANA Considerations, Registered Extensions, Private Extensions

IPP is explicitly designed to be extensible. Additional attributes can be proposed to be registered by going through the type 2 or type 3 keyword process which will register their specification after

approval with IANA. In addition specific implementation instances may support not only the basic protocol as defined in this specification, but may add vendor-specific private extensions by prefixing attribute-names with their company name registered with IANA for use in domains. See attribute syntax section. However, such private extensions shall not duplicate attribute semantics already in this specification.

9. Security Considerations

There is another Internet-Draft called "Internet Printing Protocol/1.0: Security" [22]. That document is being drafted and reviewed in parallel with this document. The mapping of IPP on top of appropriate security protocols will be described in that document. IPP does not introduce any new, general purpose security mechanisms for authentication and encryption.

A Printer may choose, for security reasons, not to return all attributes that a client requests. It may even return none of the requested attributes. In such cases, the status returned is the same as if the Printer had returned all requested attributes. The client cannot tell by such a response whether the requested attribute was present or absent on the Printer.

10. References

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

[2] Berners-Lee, T, Fielding, R., and Nielsen, H., "Hypertext Transfer Protocol - HTTP/1.0", RFC 1945, August 1995.

[3] Crocker, D., "Standard for the Format of ARPA Internet Text Messages", RFC 822, August 1982.

[4] Postel, J., "Instructions to RFC Authors", RFC 1543, October 1993.

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

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

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

[8] Borenstein, N., and Freed, N., "MIME (Multi-purpose Internet Mail Extensions) Part One: Mechanism for Specifying and Describing the Format of Internet Message Bodies", RFC 1521, September, 1993.

[9] Braden, S., "Requirements for Internet Hosts - Application and Support", RFC 1123, October, 1989,

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

[11] Berners-Lee, T., Masinter, L., McCahill, M. , "Uniform Resource Locators (URL)", RFC 1738, December, 1994.

[20] Internet Printing Protocol: Requirements

[21] Internet Printing Protocol/1.0: Model and Semantics (This document)

[22] Internet Printing Protocol/1.0: Security

[23] Internet Printing Protocol/1.0: Protocol Specification

[24] Internet Printing Protocol/1.0: Directory Schema

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

11. Author's Address

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

Phone: 801-861-7366
Fax: 801-861-4025
EMail: scott_isaacson@novell.com

Tom Hastings
Xerox Corporation
701 S. Aviation Blvd.
El Segundo, CA 90245

Phone: 310-333-6413
Fax: 310-333-5514

EMail: hastings@cp10.es.xerox.com

Robert Herriot
Sun Microsystems Inc.
2550 Garcia Ave., MPK-17
Mountain View, CA 94043

Phone: 415-786-8995
Fax: 415-786-7077
Email: robert.herriot@eng.sun.com

Roger deBry
HUC/003G
IBM Corporation
P.O. Box 1900
Boulder, CO 80301-9191

Phone: (303) 924-4080
Fax: (303) 924-9889
Email: debry@vnet.ibm.com

Patrick Powell
San Diego State University
9475 Chesapeake Dr. Suite D
San Diego, CA 92123

Phone: (619) 874-6543
Fax: (619) 279-8424
Email: papowell@sdsu.edu

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

Other Participants:

Chuck Adams - Tektronix
Jeff Barnett - IBM
Ron Bergman - Data Products
Keith Carter, IBM Corporation
Jeff Copeland - QMS
Andy Davidson - Tektronix
Mabry Dozier - QMS
Lee Farrell - Canon Information Systems
Steve Gebert - IBM
David Kellerman - Northlake Software

Rick Landau - Digital
Harry Lewis - IBM
Pete Loya - HP

Ray Lutz - Cognisys
Mike MacKay, Novell, Inc.
Carl-Uno Manros, Xerox, Corp.
Jay Martin - Underscore
Stan McConnell - Xerox
Pat Nogay - IBM
Bob Pentecost - HP
Rob Rhoads - Intel
David Roach - Unisys
Hiroyuki Sato - Canon
Bob Setterbo - Adobe
Devon Taylor, Novell, Inc.
Mike Timperman - Lexmark
Randy Turner - Sharp
Atsushi Yuki - Kyocera
Lloyd Young - Lexmark
Bill Wagner - DPI
Jim Walker - DAZEL
Chris Wellens - Interworking Labs
Rob Whittle - Novell
Don Wright - Lexmark
Peter Zehler, Xerox, Corp.

Change History

This section will be deleted when the document is forwarded to the IESG to be considered for becoming an RFC. The changes are summarized in reverse chronological order.

12.1 Changes made to version 970512, dated 12-May-1997 to make version 970603, dated 03-June-1997.

1. Removed the idea of "default beahvior"

2. Removed the idea of "best-effort" 'substitute-as-needed' and 'do-not-substitute'

3. Deleted all of the definitions from section 3 and changed the figures

4. Added references to RFC 2119 (standard conformance language) and started to capitalize all of the implementation requirement words such as SHALL, NEED NOT, MUST, etc.

5. Removed font Printer attributes and scheduling algorithm

6. Reformated all the of 6.2 (Job Template)

7. Consolidated the 3 sections on Job Template down into just one section.

8. Added attribute tables to be beginning of each first level section in 6.0. These tables show attribute name, syntax, and MAND/OPT/etc.

9. Fixed the definitions for Implements and Supports

10. Added Create-Job, Print-Job, and Send-Document operations

11. Reformatted the whole Operations section

12. Reformatted most of the standard values sections for keyword attributes. This needs to be finished for the next rev of the document.

13. Changed URL to URI.

12.2 Changes made to version 970509, dated 9-May-1997 to make version 970512, dated 12-May-1997.

1. Replaced Print with Create-Job and Send-Document

2. Added status codes (ok, error, and messages)

3. Removed all "may not" language (replace with might not)

4. Fixed attribute sections to show more detail

5. Misc. typos and edits

12.3 Changes made to version 2.2, dated 5-May-1997 to make version 970509, dated 9-May-1997.

1. Too many to mention.

12.4 Changes made to version 2.1, dated 24-April-1997 to make version 2.2, dated 5-May-1997.

Added terminology to the conformance section for keyword, attribute, attribute name, attribute value(s), default, default behavior, availabilty.

Removed most discussion about adornments and tags, except for embedded tag and job-state-reasons, report, warning, error tags.

Moved Conformance description to the appropriate operation sections, so that the Conformance section is a checklist with references to the sections where the semantics are described.

Changed the job-size attributes into xxx-requested, xxx-completed Job attributes, and xxx-supported Printer attributes.

12.5 Changes made to version 2.0, dated 26-March-1997 to make version 2.1, dated 22-April-1997.

1. Added terminology to the conformance section for supported and implemented.

12.6 Changes made to version 1.8, dated 24-March-1997 to make version 2.0, dated 26-March-1997.

The following changes were made to version 1.8 to make version 2.0:

1. Minor editing fixes

2. Submitted 2.0 as draft-ietf-ipp-model-00.txt

12.7 Changes made to version 1.7, dated 24-Mar-1997 to make version 1.8,

dated 24-March-1997.

The following changes were made to version 1.7 to make version 1.8:

1. Minor editing fixes

12.8 Changes made to version 1.6, dated 12-Mar-1997 to make version 1.7, dated 24-March-1997.

The following changes were made to version 1.6 to make version 1.7:

1. Removed Text Handling attributes

2. Removed best-effort tag, added best-effort Job attribute

3. Moved job-name from Job Template to Job Identification

4. Fixed many typos, spelling errors, etc.

5. Removed enum and added keyword

6. Print Response returns Job Status not Printer Status (since Job status includes interesting Printer

Status)

7. Removed file type, added format to Get-Attributes

8. Added Patrick Powell as author

9. Move conformance to section 1, added client and server considerations

10. Added a IANA considerations section

11. Added a type4 keyword

12. Removed the notion of "secondary tags"

13. Added more section headers

14. Merged the Job Production Attributes and Document Production Attributes section

12.9 Changes made to version 1.5, dated 11-Mar-1997 to make version 1.6, dated 12-March-1997.

The following changes were made to version 1.5, dated 11-Feb-1997 to make version 1.6, dated 12-March-1997 from the miscellaneous e-mail messages, suggestions, and agreements:

1. Added a new abstract with includes a reference to other IPP documents

2. Added Roger deBry's "new intro" document.

3. Added a modified text diagram showing IPP mapped onto a generic distributed printing model in section 2.0

4. Added some new summary and introductory comments to 2.0 to clarify that this section is not only meant to introduce the IPP model but to show a mapping onto the various architecture and product solutions on which IPP will be implemented.

5. Added a new section (3.4) summarizing the role of object attributes and introduced the notion of Job Template (Printer and Job) attributes along with "adornments"

6. Fixed up section 3.3 on Object relationships.

7. Fixed Section 3.5 on object identity. This section now shows that Jobs and Printers have both URLs and Names. Documents have Names and Ids

8. Added the correct references to Printer attributes in section 3.1

9. Fixed the references to Job attributes in section 3.2

10. Added a high level over view of the IPP operations and their purpose and interactions to section 4.1 (Roger deBry worked on this section)

11. Replaced the security section with a reference to a new IPP Security document which will eventually be merged back into this document

12. In section 5.2 I added subheading for each attribute to more easily locate the text associated with the attribute as a Job object attribute and then the text associated with the attribute as a Printer object attribute.

13. Reorganized section 3 to include a Document object.

14. Added IPP Mailing list info to contact list

12.10 Changes made to version 1.4, dated 27-Feb-1997 to make version 1.5, dated 9-March-1997.

The following changes were made to version 1.4, dated 27-Feb-1997 to make version 1.5, dated 9-March-1997 from the 3/6/97 telecon agreements:

1. Replaced the data types with the ones discussed on the 3/6/97 telecon. Added back octetString, type1Enum, type2Enum, type3Enum types as well.

2. Added range constraints to the integer data type.

3. Changed the xxx-locale attributes to be type3Enum.

4. Added standard printer-resolution enum values.

5. Added standard document-format enum values from the Printer MIB and IANA registries.

6. Added back job-hold-until attribute using named periods only and added back the job-hold-until-specified value to the job-state-reasons attribute.

7. Changed the job state from printing to processing to agree with the Printer state and changed the optional job-processing job-state-reasons value to job-printing for use with Printers that take a lot of time processing while not marking.

8. Deleted the second occurrence of the job-name attribute, since we removed any attributes that are common to both Job and Printer.

9. Changed job-priority from a type1Enum to integer(1:100) so as to have more levels so as to be able to match existing systems.

10.

Changed job-retention-period to use integerSeconds, instead of minutes.

11. Changed the Text Formatting attributes to take only the computer points as units.

12. Changed job-octets to job-K-octets, in order to be able to print large documents. Aligns with the IETF Job Monitoring MIB.

13. Clarified that the job-incomplete value of job-state-reasons is waiting for document data, as opposed waiting for the job to be closed.