R. deBry
IBM Corporation
T. Hastings
Xerox Corporation
R. Herriot
Sun Microsystems
S. Isaacson
Novell, Inc.
P. Powell
San Diego State University
June 23, 1997

Internet Printing Protocol/1.0: Model and Semantics
draft-ietf-ipp-model-02.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 introduces the terminology used within this document.

Section 3 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 system. 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 4 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 5 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 6 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 is a review of conformance issues and clarifies requirements that apply to client side and server side implementations.

Sections 8-11 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.

ISSUE: There are too many "SHALLs" in the document right now. Carl-Uno makes the comment: "I think we have gone overboard in the use of SHALL in this version. Every time we need a verb it is preceded by a SHALL, also in places where it does not serve any purpose. I think that this only detracts from the places where SHALL is really useful." These do need to be cleaned up. Also, many of our conformance requirements in the form of MANDATORY, CONDITIONALLY MANDATORY and SHALL statements are too stringent, and needs quite a bit of relaxation.

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 is not required to implement the items that correspond to features or behaviors that the implementation is not capable of realizing.

ISSUE: Should we use the following definition and define an explicit condition for each attribute labeled as CONDITIONALLY MANDATORY?

> This term means that an item MUST be implemented in a conforming implementation if the specified condition is true. Furthermore, a conforming implementation NEED NOT implement the item if the specified condition is false.

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.

ISSUE: Occasionally in the document the terms MANDATORY and OPTIONAL get used for what a requester shall supply in a request or what a provider shall return in a response. I think that it is confusing to say that such and such input parameter is OPTIONAL meaning that the requester NEED NOT supply it in a request because it sounds like we are saying that the requester NEED not implement the input parameter and that the provide need not support the input parameter. Same for responses. Instead of labeling input parameters and output parameters as MANDATORY and OPTIONAL, just say something like "the requester SHALL supply the xxx input-parameter in the yyy operation" or "the requester MAY supply the zzz input-parameter in the yyy operation" or "the requester NEED NOT supply the zzz input-parameter in the yyy operation". In headings for parameters, how about: "(SHALL be supplied)" and "(MAY be omitted)"? In the description of each parameter we need to also specify whether support of the input parameter is MANDATORY for the

provider or whether support is OPTIONAL.

2.2 Model Terminology

2.2.1 Keyword

Keywords are used within this document as identifiers of semantic entities within the abstract model. Attribute names, attribute values, attribute syntaxes, and attribute groups are represented as keywords. In this document, a keyword is a sequence of characters (length of 1 to 255) which consists of the following ASCII characters: lower-case letters, digits, hyphen ("-"), and underscore ("_"). A keyword MUST start with a lower-case 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 Parameters

A parameter is an item of information supplied in an operation consisting of a parameter name and parameter value(s) using a specific syntax for that parameter. Clients supply input parameters in an operation request and Printers return output parameters in an operation response. Most parameters have corresponding object attributes, some do not. All parameters are defined in section 5. Parameterare identified as being "MANDATORY", "CONDITIONALLY MANDATORY", or "OPTIONAL" for implementation and "SHALL be supplied" or "MAY be omitted" in operation requests and responses.

2.2.2.1 Parameter Name

Each parameter is uniquely identified in this document by its parameter name which is a keyword. The keyword parameter name is given in the section header describing that parameter. In running text in this document, parameter names are indicated inside double quotation marks (").

2.2.2.2 Parameter Value

Each parameter has one or more values. Parameter values are represented in the syntax type specified for that parameter. In running text in this document, parameter values are indicated inside single quotation marks ('), whether their parameter syntax is keyword, integer, text, etc.

2.2.2.3 Parameter Syntax

Each parameter 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] indicates the actual representation for each parameter syntax that SHALL be used for the actual protocol.

Parameter syntaxes are the same as attribute syntaxes which are defined in section 6.1.

2.2.3 Attributes

An attribute is an item of information that is associated with an object instance consisting of an attribute name and attribute value(s) using a specific syntax for that attribute. A requester sets an attribute by supplying an input parameter in an operation request which has the same syntax as the attribute. A provider returns an attribute by supplying an output parameter in an operation response which has the same syntax as the attribute. The attributes that can be set by a client have a corresponding representation as an input parameter. The attributes that can be queires by a client have a corresonding representation as an output parameter. All attributes are defined in section 6. Attributes are identified as being "MANDATORY", "CONDITIONALLY MANDATORY", or "OPTIONAL".

2.2.3.1 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 (").

2.2.3.2 Attribute Group Name

Related attributes are grouped into named attribute groups. The name of the group is a keyword. It MAY be used as the value of an input parameter in place of naming all the attributes in the group explicitly. Attribute groups are defined in section 6.

2.2.3.3 Attribute Value

Each attribute has one or more values. Attribute values are represented in the syntax type specified for that attribute. In running text in this document, attribute values are indicated inside single quotation marks ('), whether their attribute syntax is keyword, integer, text, etc.

2.2.3.4 Attribute Syntax

Each attribute is defined using an explicit attribute syntax. In this document, each attribute syntax is defined as a keyword with specific meaning. The protocol specification document [23] indicates the actual

representation for each attribute syntax that SHALL be used for the actual protocol. Attribute syntaxes are defined in section 6.1.

2.2.4 Supports

By definition, a job processing behavior or selectable feature is supported by a Printer only if that Printer responds with the corresponding attribute and the associated value in a response to a query for that attribute. 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. A conforming implementation SHALL support all MANDATORY attributes and all CONDITIONALLY MANDATORY attributes whose possible values correspond to the behaviors that the implementation is capable of realizing. Therefore conformance to IPP does not mandate that all implementations support all possible values representing all possible job processing behaviors and features.

For example, if a given instance of a Printer supports only certain document formats, then that Printer SHALL respond with the "document-format-supported" attribute populated with a set of values, possibly only one, taken from the entire set of possible values defined in this model document. This 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 populated 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 4.1)
Job (Section 4.2)
Document (Section 4.3)

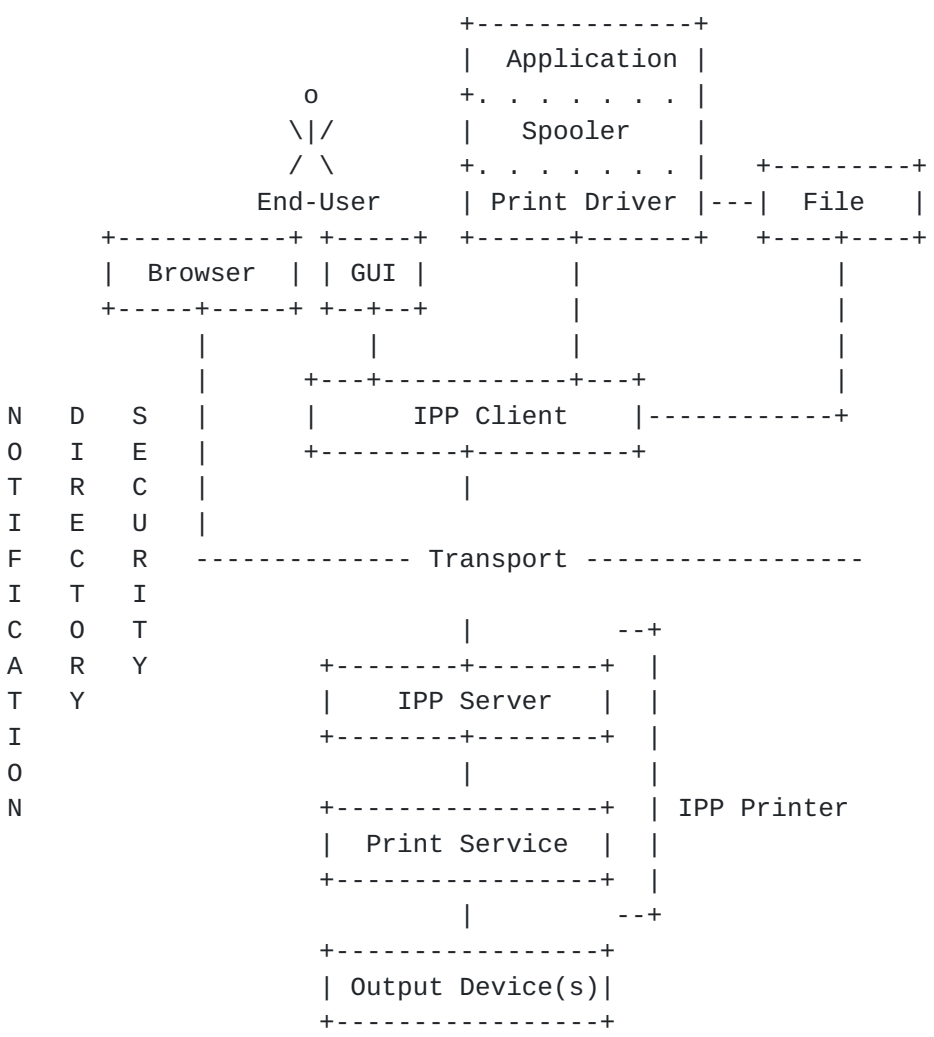
Each of these objects has a set of operations associated with it. These include:

Printer:
- Get-Operations (Section 5.1.1)
- Print-Job (Section 5.1.2)
- Print-URI (Section 5.1.3)
- Validate-Job (Section 5.1.4)
- Create-Job (Section 5.1.5)
- Get-Attributes (Section 5.1.9)
- Get-Jobs (Section 5.1.10)

Job
- Send-Document(Section 5.1.6)
- Send-URI (Section 5.1.7)
- Cancel-Job (Section 5.1.8)
- Get-Attributes (Section 5.1.9)

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. The IPP Printer MAY be embedded in an output device or MAY be embedded in a host on the network that communicates with the output device. 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, the Printer SHALL create a Job object. 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

An IPP object is defined as set of attributes that can be potentially supported by each instance of the object. The attributes for each object type are identified as MANDATORY, CONDITIONALLY MANDATORY, or OPTIONAL (see section 2). Each instance of an IPP object supports an appropriate set of attributes (with values for each of the attributes) that describe that instance. That is, an IPP Printer object is defined as set of attributes that can potentially be implemented by some entity claiming to be an IPP Printer. In the same manner, a Job object is defined as a set of attributes that are potentially associated with each instance of a Job object.

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 grouped as follows:

"job-template" attributes (section 6.2)
"printer-description" attributes (section 6.5)

Operations which are invoked on a printer include:

Get-Operations (Section 5.1.1)
Print-Job (section 5.1.2)
Print-URI (Section 5.1.3)
Validate-Job (Section 5.1.4)
Create-Job (section 5.1.5)
Get-Attributes (section 5.1.9)
Get-Jobs (section 5.1.10)

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 are 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

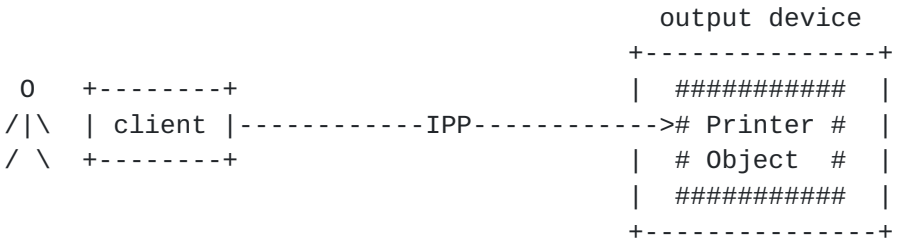
See the following figures for some examples on how to view Printer objects on top of several print system configurations. The embedded case below represents configurations 1 and 2. The hosted and fan-out figures below represent configuration 3.

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.

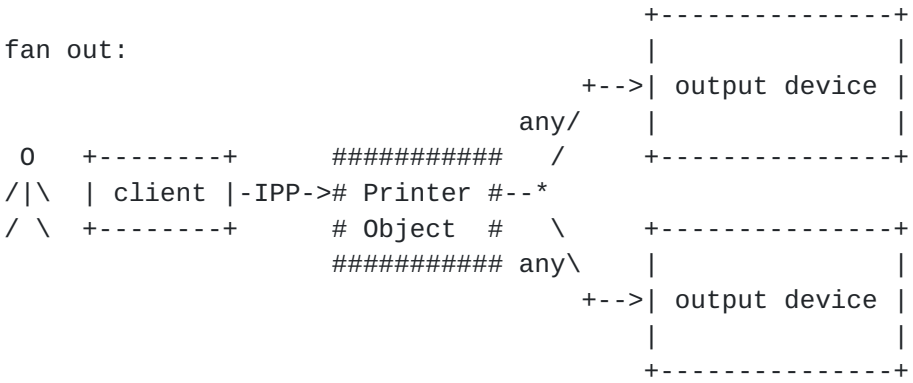
any indicates any network protocol or direct
connect, including IPP

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 Print-Job Request, a Print-URI request, or a Create-Job Request. The Printer MUST perform validation checks to verify that the job may indeed be processed. A client MAY send a Validate-Job Request (with no document data) so that the Printer performs all validation checks without the overhead of transferring all of the document data. As an example of some of the validation checks that are performed, 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 grouped as follows:

"job-template" attributes (optionally supplied by the client/end user, section 6.2)
"job-description" attributes (set by the Printer, section 6.3)

The following operations can be invoked on Jobs:

Send-Document (section 5.1.6)
Send-URI (Section 5.1.7)Cancel Job (section 5.1.8)
Get-Attributes (section 5.1.9)

4.3 Document Object

A Document object consists of printable data and Document Attributes (see section 6.4). These Document Attributes only describe the data to be printed; they do not include any specialized document processing instructions that apply to only one Document in a multi-document Job. All Job Template attributes (those attributes that describe desired job

processing behavior) are defined as part of the Job object, therefore, they apply equally to all Documents within a Job. Currently there are no operations defined for Document objects.

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, the stream SHALL contain only one document, although there can be copies of the same document data in other Documents in the same or different Jobs.

ISSUE: Does "An output device can be represented by at most one Printer object" kill "fan-in" too much?

4.5 Object Attributes

Each object type is defined by a set of possible attributes which describe the realization of each instance of an object. That is, a Printer object is defined as set of attributes which each instance of a Printer object might potentially support. In the same manner, a Job object is defined as a 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 6 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 6.2. 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

Client supplied Job Template attributes affect the rendering, production, and finishing of the documents in a job. Similar types of instructions may also be contained within the Page Description Language (PDL) of the document to be printed. The "best-effort" attribute, described in detail in section 6.2.8 is provided to help manage the conflicts between values supplied in IPP Job Template attributes and corresponding instructions contained within the body of the document itself. The "best-effort" attribute SHALL take one of 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

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.

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. If not supplied by the client, tthe Job name is created by the Printer. In all cases, the name only has local meaning, and is not constrained to be 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 unique name for this object instance

Document objects sent to an IPP Printer 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 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:	
Get-Operations (section 5.1.1)	MANDATORY
Print-Job (section 5.1.2)	MANDATORY
Print-URI (section 5.1.3)	OPTIONAL
Validate-Job (section 5.1.4)	OPTIONAL
Create-Job (section 5.1.5)	OPTIONAL
Get-Jobs (section 5.1.8)	OPTIONAL
Get-Attributes (section 5.1.9)	OPTIONAL
For a Job object:	
Send-Document (section 5.1.6)	OPTIONAL
Send-URI (section 5.1.7)	OPTIONAL
Cancel-Job (section 5.1.8)	MANDATORY
Get-Attributes (section 5.1.9)	OPTIONAL

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 three operations:

- The Print-Job operation: This operation is used if the client wants to create a Job with only a single Document and the document data is included in the request. In this case, the client "pushes" the document data to the Printer.

- The Print-URI operation: This operation is used if the client wants to create a Job with only a single Document and only a URI reference to the document data (not the document data itself) is included in the request. In this case, the Printer "pulls" the document data from the location identified by the URI.

- The Create-Job operation: This operation is used if the client wants to create a Job with one or more Documents. This operation is followed by an arbitrary number of Send-Document or Send-URI operations (each creating another Document for this Job). The Send-Document operation includes the document data with the operation request (client "pushes" the document data to the printer), and the Send-URI operation includes only a reference (a URI) to the document data (the Printer "pulls" the document data from the referenced location).

A Create-Job operation followed by a only one Send-Document operation is semantically equivalent to a Print-Job operation, however, for performance reasons, the client SHOULD use the Print-Job operation for all single Document Jobs. Throughout this model specification, the term "create request" is used to refer to any of the three operation requests that creates a new job object (a Print-Job request, a Print-URI request, or a Create-Job request).

5.1.1 Get-Operations Operation

Since some of the IPP operations defined in this specification are OPTIONAL and therefore some implementations MAY choose to not implement support them, this operation, Get-Operations, is a simple, MANDATORY operation that all implementations MUST support. The client uses this operation to query a specific implementation for a list of supported OPTIONAL operations.

5.1.1.1 Get-Operations Request

The Get-Operations Request has no parameters.

5.1.1.2 Get-Operations Response

The Printer SHALL return to the client the following output parameters as part of the Get-Operations Response:

Supported Operations:
A list of the OPTIONAL operations that this implementation supports. This set of OPTIONAL operations are 'Create-Job', 'Print-URI', 'Submit-Document', 'Submit-URI', 'Get-Jobs', and 'Get-Attributes'.

Status
Status information including error status

5.1.2 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. A Print-Job operation differs from a Print-URI operation in that a Print-Job operation contains the document data to be printed and a Print-URI operation only contains a reference to the document data.

5.1.2.1 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. If the client supplies no Job Template attributes in the Create-Job Request, the Printer uses its default value attributes when processing the job. Since a Print-Job operation is used for a Job with only one Document, the Document attributes "document-name" and "document-format" are also supplied by the client. "document-name" is MANDATORY; "document-format" is OPTIONAL.

Document Content

The client supplies the document data.

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 when accepting or rejectingJob 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 associates the "xxx" attribute with the new Job object using the value supplied by the client.

2. The client supplies a Job Template attribute but the attribute is syntactically bad: The Printer SHALL reject the job and return the 'attribute-unsupported' error code and the name of the badly formed attribute (if known) in the "unsupported-attributes" response parameter.

3. The client supplies a Job Template attribute and the attribute value is not among the values supported by the Printer: The Printer SHALL reject the Job and return the 'attribute--value-unsupported' error code and the name of the unsupported attribute in the "unsupported-attribute-values" response parameter.

4. The client supplies a Job Template attribute and the Printer does not support the attribute: The Printer rejects the attribute. The Printer returns the 'attribute-unsupported' error code and the name of the rejected attribute in the "unsupported-attributes" response parameter.

5. The client does not supply a Job Template attribute, but the Printer supports the attribute: The attribute is accepted and when the Printer creates the Job object, the Printer SHALL NOT associate

the attribute with the new 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 supports a corresponding default value attribute for the missing Job Template attribute, the Printer uses the default value. Depending on the value of the Printer's "best-effort" attribute, the Printer either guarantees the behavior corresponding to the default value or it does its best to realize the behavior of the default value. The results of processing a Job are undefined if the Printer does not support the default value attribute and the client does not supply a value in the create request.

6. The client does not supply an attribute, and the Printer does not support the attribute: The Printer accepts the Job but how the Job is finally processed (with respect to the missing Job Template attributes) is undefined.

5.1.2.2 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 printer state information which affects a job's state is reflected in the "job-state" and "job-state-reasons" attributes, it is sufficient to return only these attributes and no specific printer status attributes.

ISSUE: Randy suggest that the following are optional returns in a response: job-originating-user job-originating-host user-locale job-state job-state-reasons job-state-message output-device-assigned time-since-submission time-since-processing number-of-intervening-jobs

job-message-from-operator time-since-completion job-k-octets-completed
job-impressions-completed job-media-sheets-completed

Unsupported Attributes:
A list of attribute names which are unsupported. The existence of any attribute name in this list implies that the Job was rejected.

Unsupported Attribute Values:
A list of attribute names whose client supplied values are unsupported. The existence of any attribute name in this list implies that the Job was rejected.

ISSUE: Should we call both of these "attribute-syntax-invalid'?

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 Print-URI Operation

This operation is identical to the Print-Job operation (section 5.1.2) except that a client supplies a reference (a URI) to the document data to be printed rather than the document data itself. It is up to the IPP server to interpret the URI and subsequently "pull" the document from the source referenced by the URI string.

5.1.3.1 Print-URI Request

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

Job Template Attributes:
(see section 5.1.2.1)

Document Reference:
The client supplies the a URI reference to the document data.

5.1.3.2 Print-URI Response

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

Job Identifier:

(see section 5.1.2.2)

Job Status:
(see section 5.1.2.2)

Unsupported Attributes:
(see section 5.1.2.2)

Unsupported Attribute Values:
(see section 5.1.2.2)

Status
(see section 5.1.2.2)

5.1.4 Validate-Job Operation

This operation is identical to the Print-Job operation (section 5.1.2) except that a client supplies no document data or any reference to document data and the Printer allocates no resources (i.e., a new Job object) to process the job. The VALIDATE request is only used to verify capabilities of a printer object against whatever input parameters are supplied in the Validate-Job request.

5.1.4.1 Validate-Job Request

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

Job Template Attributes:
(see section 5.1.2.1)

5.1.4.2 Validate-Job Response

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

Job Identifier:
(see section 5.1.2.2)

Job Status:
(see section 5.1.2.2)

Unsupported Attributes:

(see section 5.1.2.2)

Unsupported Attribute Values:
(see section 5.1.2.2)

Status
(see section 5.1.2.2)

5.1.5 Create-Job Operation

This operation is similar to the Print-Job operation (section 5.1.2) except that a client supplies no document data or any reference to document data in the Create-Job request. This operation is followed by one or more Send-Document or Send-URI operations. It is possible for a given implementation to only support either Send-Document or Send-URI but not both. In that case, a client SHOULD NOT use an unsupported operation. If a Printer supports the Create-Job operation, it MUST also support one of the Send-Document or Send-URI operations or both.

5.1.5.1 Create-Job Request

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

Job Template Attributes:
(see section 5.1.2.1)

5.1.5.2 Create Job Response

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

Job Identifier:
(see section 5.1.2.2)

Job Status:
(see section 5.1.2.2)

Unsupported Attributes:
(see section 5.1.2.2)

Unsupported Attribute Values:

(see section 5.1.2.2)

Status
(see section 5.1.2.2)

(see section 5.1.2.2)

5.1.6 Send-Document Operation

Once a Job object has been created using a Create-Job operation (returning a "job-uri"), a client directs a Send-Document operation to the newly create Job object (identified by the returned "job-uri"). The operation adds a new Document to the Job object. An entire document MUST be sent in a single Send-Document operation.SEND-DOCUMENT requests are directed towards the job object referenced by the "job_URI" string returned in a successful CREATE-JOB-RESP message.

5.1.6.1 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:
A set of Document Description attributes (section 6.4).
Last Document Flag
This is a boolean flag that is set if this is the last Document for the Job.

Document Content:
The client supplies the document data.

5.1.6.2 Send-Document Response

The following output parameters are part of the Send-Document Response:

Job Status:
(see section 5.1.2.2)

Unsupported Attributes:
(see section 5.1.2.2)

Unsupported Attribute Values:
(see section 5.1.2.2)

Status:

(see section 5.1.2.2)

5.1.7 Send-URI Operation

This operation is identical to the Send-Document operation (see section 5.1.6) except that a client supplies a reference (a URI) to the document data to be printed rather than the document data itself. It is up to the IPP server to interpret the URI and subsequently "pull" the document from the source referenced by the URI string.

5.1.7.1 Send-URI Request

The client submits the request to a Job URI.

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

Document Attributes:
(see section 5.1.6.1)

Last Document Flag
(see section 5.1.6.1)

Document Reference:
The client supplies a URI reference to the document data.

5.1.7.2 Send-URI Response

The following output parameters are part of the Send-URI Response:

Job Status:
(see section 5.1.6.2)

Unsupported Attributes:
(see section 5.1.6.2)

Unsupported Attribute Values:
(see section 5.1.6.2)

Status:
(see section 5.1.6.2)

5.1.8 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.

5.1.8.1 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

5.1.8.2 Cancel-Job Response

The following information is part of the Cancel Job Response:

Status:
Status information including error status

ISSUE: Randy suggests that the following might be optionally returned in a response: job-state job-state-reasons job-state-message job-k-octets-completed job-impressions-completed job-media-sheets-completed time-since-submission time-since-processing job-originating-user job-originating-host

5.1.9 Get-Attributes Operation

The Get-Attributes operation allows a client to obtain information from a Printer or Job object. The client supplies as an operation parameter the set of attribute names and/or attribute group names that the requester is interested in. 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.

ISSUE: Should this be broken up into two sections - one for Printer one for Job?

5.1.9.1 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 supported and (2) the attribute values that are supported for the specified document format. By specifying the document format, the client can eliminate the attributes and values that are not supported.

If the client omits this input parameter, the effect SHALL be the same as if the value of the Printer's default value 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': all of the Job Template attributes that apply to a Job object (the first column of the table in Section 6.2).
- 'job-description': the Job Description attributes in Section 6.3.

For Printers, the attribute groups include:

- 'job-template': all of the Job Template attributes that apply to a Printer object (the last two columns of the table in Section 6.2).
- 'printer-description': the attributes specified in Section 6.5.

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

5.1.9.2 Get-Attributes Response

The Printer SHALL return the following response parameters as part of the Get-Attributes Response:

Result Attributes:
The requested attributes of the object with their current valuesSHALL

Unknown Attributes:
A list of attribute names included in the Get-Attributes request which are unknown by the Printer.

Status:
Status information including error status

A Printer MAY be configured, 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 supported by the Printer, the printer SHALL respond with all current values for that attribute.

2. If the client supplies an attribute name in the Requested Attributes input parameter and that attribute is not supported by the Printer, the Printer SHALL respond with the attribute name in the "unsupported attributes" response parameter.

3. If the client supplies an attribute group that is supported by the Printer, the Printer SHALL respond with all current values for each supported attribute in the group. It SHALL not respond for unsupported attributes in the group.

4. If the client supplies an attribute group keyword that is not unsupported, the Printer assumes that it is an unknown attribute and responds group name in the "unknown attribute list" response parameter.

5.1.10 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.

5.1.10.1 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:

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 oldest 50 'completed' jobs are returned. The other 25 'completed' jobs are not returned and neither are any of the 'pending' jobs returned.

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 "job-uri" attribute were supplied.

5.1.10.2 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. Jobs that are in the 'pending-held' state SHALL appear in their position as if they were 'pending' (otherwise, a user might be deceived by jobs that move

from 'pending-held' to 'pending' as seeming to jump ahead in the queue).

If the client did not supply any Job attributes, the Printer SHALL assume that the client is implicitly requesting the "job-uri" attribute (that is no other Job attributes are returned, but the Job URI for each Job).

Status
Status information including error status

A Printer MAY be configured, 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.

ISSUE: Some people still have complaints about this security statement.

5.2 Operation Status Codes and Messages

An operation status code provides information on the processing of a request. A message provides a short textual description of the Status. The status code is intended for use by automata and a 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. Status codes and suggested corresponding messages are included in section 12 "APPENDIX A - Status Codes".

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" (see section 2.2.1). The keyword in included in the section header

describing that attribute. Not only are attributes uniquely identified with keywords, some attributes take on a syntax which is a set of keywords.

6.1 Attribute Syntaxes

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

text: a sequence of characters, length: 0 to 4095, any characters. This syntax type is used for free form human readable text intended for human consumption.

name: a sequence of characters, length: 1 to 255, any characters. This syntax type is used for referencing some object or entity via a user-friendly string, such as a Printer name, a document name, a user name, or a host name.

fileName: a sequence of characters, length: 1 to 1024, any characters. This syntax type is used for referencing some file. The limit is the same as in POSIX and Microsoft NT.

keyword: a sequence of characters, length: 1 to 255, containing only the characters ASCII letters, ASCII digits, hyphen ("-"), underscore ("_"). The first character MUST be an ASCII letter. This syntax type is used for enumerating semantic identifiers of entities in the abstract protocol (specified in this document). These entities can be 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, the same keyword can not be used for two different values within 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 rfc1808. This syntax type is used for carrying Universal Resource Identifiers.

uriScheme: a sequence of characters representing the URI Scheme. These include 'http' for HTTP schemed URIs (e.g., http://...), and 'ftp' for FTP schemed URIs (e.g., ftp://...).

locale: a standard identifier for country and language. The values for this syntax type are taken from RFC 1766 [26].

octetString: a sequence of octets. This syntax type is used for opaque data, such as the document-content.

boolean: two values of 'true' and 'false'. This syntax type is like a keywordSet, but there are only two values. Note: An application might use a checkbox for an attribute with this syntax type.

integer: an integer value that is in the range from -2**31 to 2**31 - 1. Each attribute specifies the range constraint explicitly if the range is different from the full range of possible integer values (e.g., 0 - 100 for the "job-priority" attribute).

dateTime: a standard, fixed length representation of date and time (to the nearest second) as defined in RFC 1123 [27]. For example, Sun, 06 Nov 1994 08:49:37 GMT. This is a fixed-length subset of that defined by RFC 1123 (an update to RFC 822). All values MUST be represented in Greenwich Mean Time (GMT). This is indicated by the inclusion of "GMT" as the three-letter abbreviation for time.

seconds: a non-negative integer with implicit units of seconds. This is used for relative time.

milliseconds: a non-negative integer with implicit units of milliseconds. This is used for relative time.

1setOf X: 1 or more values of type X. This syntax type is used for multi-valued attributes, whose value is a set of values. Note: The syntax type is called "1setOf" to indicate that set of values SHALL NOT be empty (a set of size 0).

rangeOf X: a range of value of type X. This syntax type is used for ordered values (numeric, lexical, etc.) 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.

"type1": The editor MUST revise the IPP standard to add a new name. No private names are allowed.

"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 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.

"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.

"type4": Anyone (system administrators, system integrators, site managers, etc.) can, at any time, add new installation-defined names to a local system. Care SHOULD be taken by the implementers 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.

Each of the four types above assert some sort of registry or review process in order to be valid. "type1" values are only valid if the specification is updated, "type2" values are only valid if the PWG or an IANA approved review process approves them, "type3" values are only valid if IANA registers the value with no review process required, and "type4" values are always valid (there is no review or registration process required). Any typeN value MAY be registered using a process for some typeM where M is less than N, however such registration is NOT REQUIRED. For example, a type4 value MAY be registered in a type 1 manner (by being included in a future version of an IPP specification) however it is NOT REQUIRED.

Note: This specification defines keyword values for all of the above types, including type4 keywords.

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 which behaviors are supported by a Printer. "xxx-supported" is a CONDITIONALLY MANDATORY attribute which means that the Printer only supports 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 supports 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 supports the "xxx-supported" attribute, the Printer MUST support the corresponding default value attribute and vice versa.

4. 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 and default value attributes. The values that the client then supplies 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 and default value attributes which are supported.

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 the 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 associated with 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 Printer implements that "supported" attribute then the Printer MUST also implement the default value attribute as well. and vice versa.

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). Section All of the attributes in the first column make up the groupThe last two columns show the name and syntax for each Job Template attribute in the Printer object (the default value attribute and the supported attribute). A "No" in the table means the Printer SHALL NOT support the attribute. A "MAN" indicates that the attribute is MANDATORY.

Job	Printer: Default Value	Printer: Supported
job-name (name, MAN)	No	No
job-sheets (type4 keyword)	job-sheets (type4 keyword)	job-sheets-supported (1setOf type4 keyword)
notify-events (1setOf type2 keyword)	notify-events (1setOf type2 keyword)	notify-events-supported (1setOf type2 keyword)
notify-addresses (1setOf uri)	No	notify-addresses-supported (1setOf uri scheme)
job-priority (integer 1-100)	job-priority (integer 1-100)	job-priority-supported (rangeOf integer 1-100)
job-hold-until (type4 keyword)	job-hold-until (type4 keyword)	job-hold-until-supported (1setOf type4 keyword)
multiple-documents-are (type2 keyword)	multiple-documents-are (type2 keyword)	multiple-documents-are-supported (1setOf type2 keyword)
best-effort (type2 keyword)	best-effort (type2 keyword, MAN)	best-effort-supported (1setOf type2 keyword, MAN)
media (type4 keyword)	media (type4 keyword)	media-supported (1setOf type4 keyword)
number-up (type3 keyword)	number-up (type3 keyword)	number-up-supported (1setOf type3 keyword)
sides	sides	sides-supported

(type2 keyword)	(type2 keyword)	(1setOf type2 keyword)
resolution	resolution	resolution-supported

(type2 keyword)	(type2 keyword)	(1setOf type2 keyword)
quality (type2 keyword)	quality (type2 keyword)	quality-supported (1setOf type2 keyword)
finishings (1setOf type2 keyword)	finishings (1setOf type2 keyword)	finishings-supported (1setOf type2 keyword)
copies (integer: 0 - MAX)	copies (integer: 0 - MAX)	copies-supported (rangeOf integer 0- MAX)
document-format (type2 keyword)	document-format (type2 keyword)	document-format-supported (1setOf type3 keyword)
compression (type3 keyword)	No	compression-supported (1setOf type3 keyword)
job-k-octets (integer)	No	job-k-octets-supported (rangeOf integer)
job-impressions (integer)	No	job-impressions-supported (rangeOf integer)
job-media-sheets (integer)	No	job-media-sheets-supported (rangeOf integer)

6.2.1 job-name (name)

This attribute is the name of the job. It is a name that is more user friendly than the "job-uri" attribute value. It does not need to be unique.

If "job-name" is not supplied in the create request, the Printer, on creation of the Job, SHALL generate a name. The name MAY be generated

using the name of the first Document in the Job ((the "document-name" attribute).. If "job-name" is supplied in the create request, the Printer SHALL use its value as the name of the created Job.

6.2.2 job-sheets (type4 keyword)

This attribute determines which if 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

To force no job sheets, the system administrator SHALL set the supported value to only '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 notify-events (1setOf type2 keyword)

This attribute specifies the events for which the end user desires some sort of notification. The "notify-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.

ISSUE: Should these correspond with Job states job-completed, job-completed-aborted, job-completed-canceled?

6.2.4 notify-addresses (1setOf uri)

This attribute describes both where (the address) and how (the mechanism for delivery) 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. 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
'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.

ISSUE: Change name to "finishing-for-multiple-documents"??

Standard values are:

'single-document': If the files for the job are a and b, then files a and b SHALL be treated as a single document for finishing operations. Also, there SHALL 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 SHALL be 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 SHALL be 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)

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 support the semantics that IPP attributes override (take on a higher precedence) the embedded PDL instructions. Therefore, 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"?

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 merging 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.

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 "media" 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-feed 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) and are listed in section 14.

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 (MAY add some sort of translation, scaling, or rotation).

'two': The Printer SHALL place two logical page on a single side of an instance of the selected medium (MAY add some sort of translation, scaling, or rotation).
'four': The Printer SHALL place four logical page on a single side of an instance of the selected medium (MAY add some sort of translation, scaling, or rotation).

This attribute primarily controls the translation, scaling and rotation of 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', 'two-sided-short-edge', 'tumble'. 'duplex', and 'head-to-toe' all 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:

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

'normal':
'res-100':
'res-200':
'res-240':
'res-300':
'res-600':
'res-800':
'res-1200':
'res-1800':
'res-100x200':
'res-300x600':
'res-600x300':
'res-400x800':
'res-800x400':
'res-600x1200':
'res-1200x600':
'res-1800x600':

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-documents-are" job attribute (section 6.2.7).

6.2.15 finishing (1setOf type2 keyword)

This job attribute identifies the finishing operations 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 document-format (type2 keyword)

This optional attribute defines the document format for each Document in a Job. The standard values for this attribute are keywords. Since the complete list is rather long, the full enumeration of standard values is found in section 13 APPENDIX B - "document-format" Values.

6.2.17 compression (type3 keyword)

This attribute identifies compression algorithms used for compressed document data.

Standard values for this attribute are:

'none': no compression is used.
'zip':ZIP compression technology
'tar': UNIX TAR compression technology

6.2.18 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. If the client does not supply this attribute in the create request, the Printer might not be able to compute this value at the time the Job is created.

6.2.19 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 support this attribute at any later time as it is able to compute the total size of the Job.

6.2.20 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 creaet 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 support this attribute at any later time as it is able to compute the total size of the Job.

6.3 Job Description Attributes

The attributes in this section form the attribute group called "job-description". The following table summarizes these attributes. The third column indicates whether the attribute is a MANDATORY attribute. If it is not MANDATORY, then it is OPTIONAL.

Attribute	Syntax	MANDATORY?
job-uri	uri	MANDATORY
job-uri-user	uri	
job-originating-user	name	MANDATORY
job-originating-host	name	
user-locale	locale	
job-state	type1 keyword	MANDATORY
job-state-reasons	1setOf type2 keyword	
job-state-message	text	
output-device-assigned	name	
time-since-pending	milliseconds	MANDATORY
time-since-processing	milliseconds	MANDATORY
time-since-completed	milliseconds	MANDATORY
number-of-intervening-jobs	integer	MANDATORY
job-message-from-operator	text	
job-k-octets-completed	integer	
job-impressions-completed	integer	
job-media-sheets-completed	integer	

6.3.1 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 "job-uri" attribute as part of the response to a create request. The precise format of a job URI

SHALL be implementation dependent.

6.3.2 job-uri-user (uri)

Similar to "job-uri", this attribute contains the URI referencing an HTML page containing information about the Job.

6.3.3 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.

6.3.4 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.

6.3.5 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.

6.3.6 job-state (type1 keyword)

6.3.7 job-state-reasons (1setOf type2 keyword)

6.3.8 job-state-message (text)

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

6.3.9 output-device-assigned (name)

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.

6.3.10 time-since-pending (milliseconds)

This attribute indicates the amount of time that has passed since the Job was first put into the pending state..

6.3.11 time-since-processing (milliseconds)

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

6.3.12 time-since-completed (milliseconds)

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

6.3.13 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 is 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.

6.3.14 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.

6.3.15 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.

6.3.16 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.

6.3.17 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.

ISSUE: Should the above three attributes be regularly named as job-size-in-xxx-yyy where xxx is k-octets, impressions and media sheets and where yyy is a job state pending, processing and completed. So job-impressions becomes job-size-in-impressions-pending and job-impressions-completed is job-size-in-impressions-processing while printing and job-size-in-impressions-completed when the job completed. Thus job-impressions-completed doesn't serve two functions.

6.4 Document Attributes

This group of attributes describes the document data for the job. For single-Document Jobs, they are supplied in the Print-Job or Print-URI requests. For multi-Document Jobs, they are supplied in each Send-Document or Send-URI request.

Attribute	Syntax	MANDATORY?
document-name	name	MANDATORY
document-format	type 2 keyword	
document-uri	uri	

6.4.1 document-name (name)

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.

6.4.2 document-format (type2 keyword)

See section 6.2.16 which describes "document-format" as a Job Template

attribute.

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.

ISSUE: Now that we have Print-URI and Send-URI, do we still need this? Do we allow for the query of this attribute via Get-Attributes?

6.5 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. The following table summarizes these attributes, their syntax, and whether or not they are MANDATORY. If they are not MANDATORY, they are OPTIONAL.

Attribute	Syntax	MANDATORY?
printer-uri	uri	MANDATORY
printer-uri-user	uri	
printer-name	name	MANDATORY
printer-location	text	
printer-description	text	
printer-more-info-site	uri	
printer-driver-installer	uri	
printer-make-and-model	text	
printer-more-info-manufacturer	uri uri	
printer-state	type1 keyword	MANDATORY
printer-state-reasons	type2 keyword	
printer-state-message	text	
printer-is-accepting-jobs	boolean	MANDATORY
queued-job-count	integer	
printer-message-from-operator	text	
printer-locale	locale	MANDATORY
printer-locales-supported	1setOf locale	MANDATORY
color-supported	boolean	

6.5.1 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 printer-uri-user (uri)

Similar to "printer-uri", this attribute contains the URI for an HTML page with more information about this printer.

6.5.3 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.

6.5.4 printer-location (text)

This attribute identifies the location of this printer.

6.5.5 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".

6.5.6 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 printer manufacturer may initially populate this attribute.

6.5.7 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 printer manufacturer may initially populate this attribute.

6.5.8 printer-make-and-model (text)

This attribute identifies the make and model of the printer.

6.5.9 printer-more-info-manufacturer (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.

6.5.10 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 in the given printer state.

A Printer SHALL continually keep this attribute set to the value in the table below which most accurately reflects the state of the Printer. A Printer NEED NOT implement all values if they are not applicable to a given implementation.

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. 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. Also, 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.

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

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

Each MAY 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.

An implementation may add 'error-', 'warning-', or 'report-' to any of the reasons below to indicate the level of severity.

If a Printer controls more than one output device, each value of this attribute SHALL apply to one or 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:

'media-needed': A tray has run out of media.
'media-jam': The printer has a media 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.

ISSUE: Bob suggests: From an English point of view the suffixes should be -partly and -mostly rather than -report and -warning with all being -warnings. For example, paused-partly-warning or media-needed-mostly-warning

'connecting-to-device': 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.
'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.
'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.
'toner-low': The Printer is low on toner.

6.5.12 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.

6.5.13 printer-state-message (text)

This attribute specifies the additional information 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.

6.5.14 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.

6.5.15 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.

6.5.16 printer-locale (locale)

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

6.5.17 printer-locales-supported (1setOf locale)

This attribute specifies the locales that the Printer operates in.

6.5.18 color-supported (boolean)

This attribute identifies whether the Printer is capable of any type of

color printing at all. All document instructions having to do with color are embedded within the document PDL (none are external IPP attributes).

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 support 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 support the "sides" attribute. For an output device that does not support any of the finishing attribute values, a Printer need not support the "finishing" 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 implementation might not.

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

A client SHALL be able to accept any of the attribute syntaxes defined in Section 6.1 that may be returned to it in a response from a Printer

A query response may contain attributes and values that the client does not expect. Therefore, a client implementation MUST gracefully handle such responses and not refuse to interoperate with a conforming Printer that is returning extended registered or private attributes and/or attribute values that conform to Section 8. Clients may choose to ignore any attributes that it does not understand.

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 4.1 Printer Object
Section 4.2 Job Object
Section 4.3 Document Object

7.3.2 Operations

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

For a Printer object:

Get-Operations (section 5.1.1)	MANDATORY
Print-Job (section 5.1.2)	MANDATORY
Print-URI (section 5.1.3)	OPTIONAL
Validate-Job (section 5.1.4)	OPTIONAL
Create-Job (section 5.1.5)	OPTIONAL
Get-Jobs (section 5.1.8)	OPTIONAL
Get-Attributes (section 5.1.9)	OPTIONAL

For a Job object:

Send-Document (section 5.1.6)	OPTIONAL
Send-URI (section 5.1.7)	OPTIONAL
Cancel-Job (section 5.1.8)	MANDATORY
Get-Attributes (section 5.1.9)	OPTIONAL

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 support all of the MANDATORY attributes, as defined in this specification in the indicated sections.

Conforming Printer implementations SHALL support all CONDITIONALLY MANDATORY attributes as defined in this specification in the indicated sections that represent features and behaviors that the actual output

device implements.

It is not required that a conforming Printer support all values of all supported 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 Printer extensions

A conforming Printer may support registered extensions and private extensions, as long as they meet the requirements specified in Section 8.

7.3.5 Attribute Syntaxes

A Printer SHALL be able to accept any of the attribute syntaxes defined in Section 6.1 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 6.1.

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 Printer) 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.

ISSUE: The above is not quite correct. Waiting for better description from the security document [22].

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), 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

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

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 95123

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.

12. APPENDIX A - Status Codes

The Status keyword 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.

The prefix of the Status keyword defines the class of response as follows:

informational - Request received, continuing process

successful - The action was successfully received, understood, and accepted

redirection - Further action must be taken in order to complete the request

client-error - The request contains bad syntax or cannot be fulfilled

server-error - The server failed to fulfill an apparently valid request

IPP status codes are extensible. IPP applications are not required to understand the meaning of all registered status codes, though such understanding is obviously desirable. However, applications shall understand the class of any status code, as indicated by the prefix, and treat any unrecognized response as being equivalent to the first status code of that class, with the exception that an unrecognized response shall not be cached. For example, if an unrecognized status code of client-error-foo-bar is received by the client, it can safely assume that there was something wrong with its request and treat the response as if it had received a client-error-bad-request status code. In such cases, IPP applications should present the Message to the user, since that Message is likely to include human-readable information which will explain the unusual status.

12.1 Status Codes (type2 keyword)

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

12.1.1 Informational

This class of status code indicates a provisional response and is to be used for informational purposes only. There are no status codes defined in IPP 1.0 for this class of status code.

12.1.2 Successful Status Codes

This class of status code indicates that the client's request was successfully received, understood, and accepted.

12.1.2.1 successful-OK (IPPL1)

The request has succeeded.

Note: IPPL1 only includes OK. IPPL1 does not include Created nor No Content successful status codes. This is consistent with our agreement at the IPP Model telecon on May 9, but I believe we had this discussion before Bob Herriot joined the telecon.

12.1.3 Redirection Status Codes

This class of status code indicates that further action needs to be taken to fulfill the request. There are no status codes defined in IPP 1.0 for this class of status code.

12.1.4 Client Error Status Codes

This class of status code is intended for cases in which the client seems to have erred. The server should return a Message containing an explanation of the error situation and whether it is a temporary or permanent condition. IPP applications should display any returned Message to the end-user. Unless otherwise noted, these status codes apply to all operations.

12.1.4.1 client-error-bad-request (IPPL1)

The request could not be understood by the server due to malformed syntax. The IPP application should not repeat the request without modifications.

12.1.4.2 client-error-unauthorized

The request requires user authentication. The IPP client may repeat the request with suitable authorization credentials. If the request already

included authorization credentials, then this status code indicates that authorization has been refused for those credentials. If this response contains the same challenge as the prior response, and the user agent

has already attempted authentication at least once, then the user should be presented the Message in the response, since that Message may include relevant diagnostic information.

12.1.4.3 client-error-payment-required

This request requires payment by the end-user to perform the operation.

12.1.4.4 client-error-forbidden (IPPL1)

The server understood the request, but is refusing to fulfill it. Authorization will not help and the request should not be repeated. This status code is commonly used when the server does not wish to reveal exactly why the request has been refused or when no other response is applicable.

12.1.4.5 client-error-method-not-allowed

The operation specified in the request is not allowed for the object identified by the request URI.

12.1.4.6 client-error-timeout (NEW)

The client did not produce a request within the time that the server was prepared to wait. The client MAY repeat the request without modifications at any later time.

12.1.4.7 client-error-not-found

The server has not found anything matching the request URI. No indication is given of whether the condition is temporary or permanent.

In practice, an application should avoid this situation by presenting a list of valid Printer URIs and Job URIs to the end-user.

12.1.4.8 client-error-gone

The requested object is no longer available at the server and no forwarding address is known. This condition should be considered permanent. Clients with link editing capabilities should delete references to the request URI after user approval. If the server does not know or has no facility to determine,whether or not the condition is permanent, the status code client-error-not-found should be used instead. This response is cachable unless indicated otherwise.

This response is primarily intended to assist the task of web maintenance by notifying the recipient that the resource is intentionally unavailable and that the server owners desire that remote

links to that resource be removed. Such an event is common for limited-time, promotional services and for resources belonging to individuals no longer working at the server's site. It is not necessary to mark all permanently unavailable resources as "gone" or to keep the mark for any length of time -- that is left to the discretion of the server owner.

12.1.4.9 client-error-request-entity-too-large (IPPL1)

The server is refusing to process a request because the request entity is larger than the server is willing or able to process. An IPP Printer returns this status code when it limits the size of print jobs and it receives a print job that exceeds that limit.

12.1.4.10 client-error-request-URI-too-long

The server is refusing to service the request because the request URI is longer than the server is willing to interpret. This rare condition is only likely to occur when a client has improperly submitted a request with long query information (e.g. an IPP application allows an end-user to enter an invalid URI), when the client has descended into a URL "black hole" of redirection (e.g., a redirected URL prefix that points to a suffix of itself), or when the server is under attack by a client attempting to exploit security holes present in some servers using fixed-length buffers for reading or manipulating the Request-URI.

12.1.4.11 client-error-unsupported-media-type (IPPL1)

The server is refusing to service the request because the print data is in a format, as specified in document-format, not supported by the IPP Printer.

12.1.4.12 client-error-attribute-value-not-supported

For a Create-Job, Print-Job or Validate operation, if the IPP Printer does not support at least one attribute value specified in the request, the Printer shall return this status. For example, if the request requires A4 paper and that paper size is not supported by the Printer, the Printer shall return this status.

For a Get-Jobs operation, if the IPP Printer does not support at least one attribute value for Job Owner and/or Job States in the request, the Printer shall return this status.

In practice, an IPP application should avoid this situation by querying an IPP Printer for its valid attributes and values before performing an

operation on the Printer.

12.1.5 Server Error Status Codes

This class of status codes indicates cases in which the server is aware that it has erred or is incapable of performing the request. The server should include a Message containing an explanation of the error situation, and whether it is a temporary or permanent condition. IPP applications should display any included Message to the user. These response codes are applicable to any operation.

12.1.5.1 server-error-internal-server-error

The server encountered an unexpected condition which prevented it from fulfilling the request.

12.1.5.2 server-error-operation-not-implemented

The server does not support the functionality required to fulfill the request. This is the appropriate response when the server does not recognize an operation and is not capable of supporting it for any object.

12.1.5.3 server-error-service-unavailable

The server is currently unable to handle the request due to a temporary overloading or maintenance of the server. The implication is that this is a temporary condition which will be alleviated after some delay. If known, the length of the delay may be indicated in the Message. If no delay is given, the IPP application should handle the response as it would for a server-error-internal-server-error response.

12.1.5.4 server-error-timeout (NEW)

The server did not produce a response within the time that the client was prepared to wait. The client MAY repeat the request without modifications at any later time.

12.1.5.5 server-error-HTTP-version-not-supported (NEW)

The server does not support, or refuses to support, the HTTP protocol version that was used in the request message. The server is indicating that it is unable or unwilling to complete the request using the same major version as the client other than with this error message. The response SHOULD contain a message describing why the version is not supported and what other protocols are supported by that server.

12.1.5.6 server-error-IPP-version-not-supported

The server does not support, or refuses to support, the IPP protocol version that was used in the request message. The server is indicating that it is unable or unwilling to complete the request using the same major version as the client other than with this error message. The response should contain a Message describing why that version is not supported and what other versions are supported by that server.

A conforming IPP client shall specify the valid version for IPP 1.0 on each request. A conforming IPP server shall not return this status code to a conforming IPP 1.0 client. An IPP server shall return this status code to a non-conforming IPP client.

12.1.5.7 server-error-printer-error

A printer error, such as a paper jam, occurs while the IPP Printer processes a Print or Send operation. The response shall contain the Job Status with the specific error and should contain a Message describing the error. An IPP application should check the Job Status in the response for the specific error.

12.1.5.8 server-error-write-fault

A write error, such as a memory overflow (i.e. the document data exceeds the memory of the Printer) or a disk full condition, occurs while the IPP Printer processes a Print or Send operation.

12.2 Mapping of HTTP 1.1 Status Codes to IPP Status Keywords

HTTP 1.1 Status	IPP Keyword
100 **Continue**	none
101 **Switching Protocols**	none
200 **OK**	successful-OK
201 **Created**	successful-OK
202 **Accepted**	successful-OK
203 **Non-Authoritive Information**	successful-OK
204 **No Content**	successful-OK
205 **Reset Content**	none
206 **Partial Content (GET)**	none
300 **Multiple Choices**	none
301 **Moved Permanently**	none
302 **Moved Temporarily**	none
303 **See Other**	none
304 **Not Modified (GET)**	none
305 **Use Proxy**	none
400 **Bad Request**	client-error-bad-request
401 **Unauthorized**	client-error-unauthorized
402 **Payment Required**	client-error-payment-required
403 **Forbidden**	client-error-forbidden
404 **Not Found**	client-error-not-found
405 **Method Not Allowed**	client-error-method-not-allowed
406 **Not Acceptable**	client-error-bad-request
407 **Proxy Authentication Required**	client-error-unauthorized
408 **Request Timeout**	client-error-timeout
409 **Conflict (most likely PUT)**	client-error-bad-request
410 **Gone**	client-error-gone
411 **Length Required**	client-error-bad-request
412 **Precondition Failed**	client-error-bad-request
413 **Request Entity Too Large**	client-error-request-entity-too-large
414 **Request-URI Too Long**	client-error-request-URI-too-long
415 **Unsupported Media Type**	client-error-unsupported-media-type
none	client-error-attribute-value-not-supported
500 **Internal Server Error**	server-error-internal-server-error
501 **Not Implemented**	server-error-not-implemented
502 **Bad Gateway**	server-error-internal-server-error
503 **Service Unavailable**	server-error-service-unavailable
504 **Gateway Timeout**	server-error-timeout

505 HTTP Version Not Supported	server-error-HTTP-version-not-supported
none	server-error-IPP-version-not-supported

none	server-error-printer-error
none	server-error-write-fault

12.3 Status Keywords for IPP Operations

PJ = Print-Job, PU = Print-URI, CJ = Create-Job, SD = Send-Document
SU = Send-URI, V = Validate, GA = Get-Attributes, GJ = Get-Jobs
GO = Get-Operations, C = Cancel-Job

	IPP Operations									
IPP Status Keyword	PJ	PU	CJ	SD	SU	V	GA	GJ	GO	C
successful-OK	x	x	x	x	x	x	x	x	x	x
client-error-bad-request	x	x	x	x	x	x	x	x	x	x
client-error-unauthorized	x	x	x	x	x	x	x	x	x	x
client-error-payment-required	x	x	x	x	x	x				
client-error-forbidden	x	x	x	x	x	x	x	x	x	x
client-error-not-found	x	x	x	x	x	x	x	x	x	x
client-error-method-not-allowed	x	x	x	x	x	x	x	x	x	x
client-error-timeout	x	x	x	x	x	x	x	x	x	x
client-error-gone	x	x	x	x	x	x	x	x	x	x
client-error-request-entity-too-large	x			x						
client-error-request-URI-too-long	x	x	x	x	x	x	x	x	x	x
client-error-unsupported-media-type	x	x		x	x					
client-error-attribute-value-not-supported	x	x	x			x				
server-error-internal-server-error	x	x	x	x	x	x	x	x	x	x
server-error-service-unavailable	x	x	x	x	x	x	x	x	x	x
server-error-timeout	x	x	x	x	x	x	x	x	x	x
server-error-HTTP-version-not-supported	x	x	x	x	x	x	x	x	x	x
server-error-IPP-version-not-supported	x	x	x	x	x	x	x	x	x	x
server-error-printer-error	x	x	x	x	x					
server-error-write-fault	x	x	x	x	x					

13. APPENDIX B - "document-format" Values

The Printer Working Group has registered a set of values 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 is 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. In the IPP Protocol Specification [22], the keyword value SHALL be encoded as a MIME type.

The standard values are:

'other': 1 -
'PCL': 3 - PCL. Starting with PCL version 5, HP-GL/2 is included as part of the PCL language. PCL and HP-GL/2 are registered trademarks of Hewlett-Packard Company.
'HPGL': 4 - Hewlett-Packard Graphics Language. HP-GL is a registered trademark of Hewlett-Packard Company.
'PJL': 5 - Peripheral Job Language. Appears in the data stream between data intended for a page description language. Hewlett-Packard Co.
'PS': 6 - PostScript Language (tm) Postscript - a trademark of Adobe Systems Incorporated which may be registered in certain jurisdictions
'IPDS': 7 - Intelligent Printer Data Stream Bi-directional print data stream for documents consisting of data objects (text, image, graphics, bar codes), resources (fonts, overlays) and page, form and finishing instructions. Facilitates system level device control, document tracking and error recovery throughout the print process. Pennant Systems, IBM
'PPDS': 8 - IBM Personal Printer Data Stream. Originally called IBM ASCII, the name was changed to PPDS when the Laser Printer was introduced in 1989. Lexmark International, Inc.
'EscapeP': 9 - Epson Corp.
'Epson': 10 -
'DDIF': 11 - Digital Document Interchange Format Digital Equipment Corp., Maynard MA
'Interpress': 12 - Xerox Corp.
'ISO6429': 13 - ISO 6429. Control functions for Coded Character Sets (has ASCII control characters, plus additional controls for character imaging devices.) ISO Standard, Geneva, Switzerland
'LineData': 14 - line-data: Lines of data as separate ASCII or EBCDIC records and containing no control functions (no CR, LF, HT, FF, etc.). For use with traditional line printers. May use CR and/or LF to delimit lines, instead of records. See ISO 10175 Document Printing Application (DPA) ISO standard, Geneva, Switzerland
'MODCA': 15 - Mixed Object Document Content Architecture Definitions that allow the composition, interchange, and presentation of final form documents as a collection of data objects (text, image, graphics, bar codes), resources (fonts, overlays) and page, form and finishing instructions. Pennant Systems, IBM
'REGIS': 16 - Remote Graphics Instruction Set, Digital Equipment Corp., Maynard MA
'SCS': 17 - SNA Character String Bi-directional print data stream for

```
    SNA LU-1 mode of communications IBM
  'SPDL': 18 - ISO 10180 Standard Page Description Language ISO
    Standard
  'TEK4014': 19 - Tektronix Corp.
```

'PDS': 20 -
'IGP': 21 - Printronix Corp.
'CodeV': 22 - Magnum Code-V, Image and printer control language used to control impact/dot- matrix printers. QMS, Inc., Mobile AL
'DSCDSE': 23 - DSC-DSE: Data Stream Compatible and Emulation Bi-directional print data stream for non-SNA (DSC) and SNA LU-3 3270 controller (DSE) communications IBM
'WPS': 24 - Windows Printing System, Resource based command/data stream used by Microsoft At Work Peripherals. Developed by the Microsoft Corporation.
'LN03': 25 - Early DEC-PPL3, Digital Equipment Corp.
'CCITT': 26 -
'QUIC': 27 - QUIC (Quality Information Code), Page Description Language for laser printers. Included graphics, printer control capability and emulation of other well- known printer . QMS, Inc.
'CPAP': 28 - Common Printer Access Protocol Digital Equipment Corp
'DecPPL': 29 - Digital ANSI-Compliant Printing Protocol (DEC-PPL) Digital Equipment Corp
'SimpleText': 30 - simple-text: character coded data, including NUL, CR , LF, HT, and FF control characters. See ISO 10175 Document Printing Application (DPA) ISO standard, Geneva, Switzerlan
'NPAP': 31 - Network Printer Alliance Protocol (NPAP). This protocol has been superseded by the IEEE 1284.1 TIPSI standard. (ref. LangTIPSI(49)).
'DOC': 32 - Document Option Commands, Appears in the data stream between data intended for a page description . QMS, Inc
'imPress': 33 - imPRESS, Page description language originally developed for the ImageServer line of systems. A binary language providing representations for text, simple graphics (rules, lines, conic sections), and some large forms (simple bit-map and CCITT group 3/4 encoded).The language was intended to be sent over an 8-bit channel and supported early document preparation languages (e.g. TeX and TROFF). QMS, Inc.
'Pinwriter': 34 - 24 wire dot matrix printer for USA, Europe, and Asia except Japan. More widely used in Germany, and some Asian countries than in US. NEC
'NPDL': 35 - Page printer for Japanese market. NEC
'NEC201PL': 36 - Serial printer language used in the Japanese market. NEC
'Automatic': 37 - Automatic PDL sensing. Automatic sensing of the interpreter language family by the printer examining the document content. Which actual interpreter language families are sensed depends on the printer implementation.
'Pages': 38 - Page printer Advanced Graphic Escape Set IBM Japan

```
  'LIPS': 39 - LBP Image Processing System
  'TIFF': 40 - Tagged Image File Format (Aldus)
  'Diagnostic': 41 - A hex dump of the input to the interpreter
```

'PSPrinter': 42 - The PostScript Language used for control (with any PDLs) Adobe Systems Incorporated
'CaPSL': 43 - Canon Print Systems Language
'EXCL': 44 - Extended Command Language Talaris Systems Inc
'LCDS': 45 - Line Conditioned Data Stream Xerox Corporatio
'XES': 46 - Xerox Escape Sequences Xerox Corporation
'PCLXL': 47 - Printer Control Language. Extended language features for printing, and printer control. Technical reference manual # TBD. Hewlett-Packard Co.
'ART': 48 - Advanced Rendering Tools (ART). Page Description language originally developed for the Laser Press printers. Tehnical reference manual: "ART IV Reference Manual", No F33M. Fuji Xerox Co., Ltd.
'TIPSI': 49 - Transport Independent Printer System Interface (ref. IEEE Std. 1284.1)
'Prescribe': 50 - Page description and printer control language. It can be described with ordinary ASCII characters. Technical reference manual: "PRESCRIBE II Programming Manual"
'LinePrinter': 51 - A simple-text character stream which supports the control codes LF, VT, FF and CR plus Centronics or Dataproducts Vertical Format Unit (VFU). language is commonly used on many older model line and matrix printers.
'IDP': 52 - Imaging Device Protocol Apple Computer.
'XJCL': 53 - Xerox Corp.

One special value is 'auto-sense'. However a client SHALL NOT supply the value 'auto-sense' in a create request. If the "document-format" is unknown for a certain document, the client SHALL NOT supply the attribute in the create request or the Send-Document Request.

14. APPENDIX C - "media" Values

Standard values are taken from several sources.

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

'default': The default medium for the output device
'iso-a4-white': Specifies the ISO A4 white medium
'iso-a4-colored': Specifies the ISO A4 coloured medium
'iso-a4-transparent' Specifies the ISO A4 transparent medium
'iso-a3-white': Specifies the ISO A3 white medium
'iso-a3-colored': Specifies the ISO A3 coloured medium
'iso-a5-white': Specifies the ISO A5 white medium

```
  'iso-a5-colored': Specifies the ISO A5 coloured medium
  'iso-b4-white': Specifies the ISO B4 white medium
  'iso-b4-colored': Specifies the ISO B4 coloured medium
```

'iso-b5-white': Specifies the ISO B5 white medium
'iso-b5-colored': Specifies the ISO B5 coloured medium
'jis-b4-white': Specifies the JIS B4 white medium
'jis-b4-colored': Specifies the JIS B4 coloured medium
'jis-b5-white': Specifies the JIS B5 white medium
'jis-b5-colored': Specifies the JIS B5 coloured medium

The following standard values are defined for North American media:

'na-letter-white': Specifies the North American letter white medium
'na-letter-colored': Specifies the North American letter coloured medium
'na-letter-transparent': Specifies the North American letter transparent medium
'na-legal-white': Specifies the North American legal white medium
'na-legal-colored': Specifies the North American legal coloured medium

The following standard values are defined for envelopes:

'iso-b4-envelope': Specifies the ISO B4 envelope medium
'iso-b5-envelope': Specifies the ISO B5 envelope medium
'iso-c3-envelope': Specifies the ISO C3 envelope medium
'iso-c4-envelope': Specifies the ISO C4 envelope medium
'iso-c5-envelope': Specifies the ISO C5 envelope medium
'iso-c6-envelope': Specifies the ISO C6 envelope medium
'iso-designated-long-envelope': Specifies the ISO Designated Long envelope medium
'na-10x13-envelope': Specifies the North American 10x13 envelope medium
'na-9x12-envelope': Specifies the North American 9x12 envelope medium
'monarch-envelope': Specifies the Monarch envelope
'na-number-10-envelope': Specifies the North American number 10 business envelope medium
'na-7x9-envelope': Specifies the North American 7x9 inch envelope
'na-9x11-envelope': Specifies the North American 9x11 inch envelope
'na-10x14-envelope': Specifies the North American 10x14 inch envelope
'na-number-9-envelope': Specifies the North American number 9 business envelope
'na-6x9-envelope': Specifies the North American 6x9 inch envelope
'na-10x15-envelope': Specifies the North American 10x15 inch envelope

The following standard values are defined for the less commonly used media (white-only):

'executive-white': Specifies the white executive medium
'folio-white': Specifies the folio white medium
'invoice-white': Specifies the white invoice medium
'ledger-white': Specifies the white ledger medium
'quarto-white': Specified the white quarto medium
'iso-a0-white': Specifies the ISO A0 white medium
'iso-a1-white': Specifies the ISO A1 white medium
'iso-a2-white': Specifies the ISO A2 white medium
'iso-a6-white': Specifies the ISO A6 white medium
'iso-a7-white': Specifies the ISO A7 white medium
'iso-a8-white': Specifies the ISO A8 white medium
'iso-a9-white': Specifies the ISO A9 white medium
'iso-10-white': Specifies the ISO A10 white medium
'iso-b0-white': Specifies the ISO B0 white medium
'iso-b1-white': Specifies the ISO B1 white medium
'iso-b2-white': Specifies the ISO B2 white medium
'iso-b3-white': Specifies the ISO B3 white medium
'iso-b6-white': Specifies the ISO B6 white medium
'iso-b7-white': Specifies the ISO B7 white medium
'iso-b8-white': Specifies the ISO B8 white medium
'iso-b9-white': Specifies the ISO B9 white medium
'iso-b10-white': Specifies the ISO B10 white medium
'jis-b0-white': Specifies the JIS B0 white medium
'jis-b1-white': Specifies the JIS B1 white medium
'jis-b2-white': Specifies the JIS B2 white medium
'jis-b3-white': Specifies the JIS B3 white medium
'jis-b6-white': Specifies the JIS B6 white medium
'jis-b7-white': Specifies the JIS B7 white medium
'jis-b8-white': Specifies the JIS B8 white medium
'jis-b9-white': Specifies the JIS B9 white medium
'jis-b10-white': Specifies the JIS B10 white medium

The following standard values are defined for engineering media:

'a': Specifies the engineering A size medium
'b': Specifies the engineering B size medium
'c': Specifies the engineering C size medium
'd': Specifies the engineering D size medium
'e': Specifies the engineering E size medium

The following standard values are defined for input-trays (from ISO DPA and the Printer MIB):

```
  'top': The top input tray in the printer.
  'middle': The middle input tray in the printer.
  'bottom': The bottom input tray in the printer.
```

'envelope': The envelope input tray in the printer.
'manual': The manual feed input tray in the printer.
'large-capacity': The large capacity input tray in the printer.
'main': The main input tray
'side': The side input tray

The following standard values are defined for media sizes (from ISO DPA):

'iso-a0': Specifies the ISO A0 size: 841 mm by 1189 mm as defined in ISO 216
'iso-a1': Specifies the ISO A1 size: 594 mm by 841 mm as defined in ISO 216
'iso-a2': Specifies the ISO A2 size: 420 mm by 594 mm as defined in ISO 216
'iso-a3': Specifies the ISO A3 size: 297 mm by 420 mm as defined in ISO 216
'iso-a4': Specifies the ISO A4 size: 210 mm by 297 mm as defined in ISO 216
'iso-a5': Specifies the ISO A5 size: 148 mm by 210 mm as defined in ISO 216
'iso-a6': Specifies the ISO A6 size: 105 mm by 148 mm as defined in ISO 216
'iso-a7': Specifies the ISO A7 size: 74 mm by 105 mm as defined in ISO 216
'iso-a8': Specifies the ISO A8 size: 52 mm by 74 mm as defined in ISO 216
'iso-a9': Specifies the ISO A9 size: 37 mm by 52 mm as defined in ISO 216
'iso-a10': Specifies the ISO A10 size: 26 mm by 37 mm as defined in ISO 216
'iso-b0': Specifies the ISO B0 size: 1000 mm by 1414 mm as defined in ISO 216
'iso-b1': Specifies the ISO B1 size: 707 mm by 1000 mm as defined in ISO 216
'iso-b2': Specifies the ISO B2 size: 500 mm by 707 mm as defined in ISO 216
'iso-b3': Specifies the ISO B3 size: 353 mm by 500 mm as defined in ISO 216
'iso-b4': Specifies the ISO B4 size: 250 mm by 353 mm as defined in ISO 216
'iso-b5': Specifies the ISO B5 size: 176 mm by 250 mm as defined in ISO 216

'iso-b6': Specifies the ISO B6 size: 125 mm by 176 mm as defined in ISO 216
'iso-b7': Specifies the ISO B7 size: 88 mm by 125 mm as defined in ISO 216

'iso-b8': Specifies the ISO B8 size: 62 mm by 88 mm as defined in ISO 216
'iso-b9': Specifies the ISO B9 size: 44 mm by 62 mm as defined in ISO 216
'iso-b10': Specifies the ISO B10 size: 31 mm by 44 mm as defined in ISO 216
'na-letter': Specifies the North American letter size: 8.5 inches by 11 inches
'na-legal': Specifies the North American legal size: 8.5 inches by 14 inches
'executive': Specifies the executive size (7.25 X 10.5 in)
'folio': Specifies the folio size (8.5 X 13 in)
'invoice': Specifies the invoice size (5.5 X 8.5 in)
'ledger': Specifies the ledger size (11 X 17 in)
'quarto': Specifies the quarto size (8.5 X 10.83 in)
'iso-c3': Specifies the ISO C3 size: 324 mm by 458 mm as defined in ISO 269
'iso-c4': Specifies the ISO C4 size: 229 mm by 324 mm as defined in ISO 269
'iso-c5': Specifies the ISO C5 size: 162 mm by 229 mm as defined in ISO 269
'iso-c6': Specifies the ISO C6 size: 114 mm by 162 mm as defined in ISO 269
'iso-designated-long': Specifies the ISO Designated Long size: 110 mm by 220 mm as defined in ISO 269
'na-10x13-envelope': Specifies the North American 10x13 size: 10 inches by 13 inches
'na-9x12-envelope': Specifies the North American 9x12 size: 9 inches by 12 inches
'na-number-10-envelope': Specifies the North American number 10 business envelope size: 4.125 inches by 9.5 inches
'na-7x9-envelope': Specifies the North American 7x9 inch envelope size
'na-9x11-envelope': Specifies the North American 9x11 inch envelope size
'na-10x14-envelope': Specifies the North American 10x14 inch envelope size
'na-number-9-envelope': Specifies the North American number 9 business envelope size
'na-6x9-envelope': Specifies the North American 6x9 envelope size
'na-10x15-envelope': Specifies the North American 10x15 envelope size
'monarch-envelope': Specifies the Monarch envelope size (3.87 x 7.5 in)
'jis-b0': Specifies the JIS B0 size: 1030mm x 1456mm

```
  'jis-b1': Specifies the JIS B1 size: 728mm x 1030mm
  'jis-b2': Specifies the JIS B2 size: 515mm x 728mm
  'jis-b3': Specifies the JIS B3 size: 364mm x 515mm
  'jis-b4': Specifies the JIS B4 size: 257mm x 364mm
```

'jis-b5': Specifies the JIS B5 size: 182mm x 257mm
'jis-b6': Specifies the JIS B6 size: 128mm x 182mm
'jis-b7': Specifies the JIS B7 size: 91mm x 128mm
'jis-b8': Specifies the JIS B8 size: 64mm x 91mm
'jis-b9': Specifies the JIS B9 size: 45mm x 64mm
'jis-b10': Specifies the JIS B10 size: 32mm x 45mm