Subj: IPP Bake Off 2 Issues
From: Peter Zehler, Tom Hastings, and Bob Herriot
File: Issues-raised-at-Bake-Off2.doc
Version: 2.0
Date: 6/10/1999

This version incorporates the discussion on the mailing list and three telecons held 3/24/99, 3/31/99, and 4/7/99 and the New Orleans meeting, 4/14-4/15 and the 4/21/99, 4/28/99, 5/5/99, 5/12/99, and 5/19/99 telecons, and the Philadelphia meeting, 5/26-5/27 on resolving the IPP/1.1 issues raised at Bake Off 2. The revision marks show changes since the 4/12/1999 version. In the suggested text, the revision marks show changes from the existing text in the IPP/1.0 Model and Semantics document (RFC 2566).

We've taken the issues that Peter published in the Bake Off 2 Summary and started a separate file. We've add some additional information that we gathered at the Bake Off with the people raising the issues. We've also added to each issue, either a list of "possible alternatives" or a "suggested clarification", "suggested change", or "suggested addition" for the discussion, so that we can reach agreement as soon as possible. Finally, we've added "suggested text" with proposed resolutions. This text is what is to be published in the June 11 Internet Draft.

Status of Issues and Summary

This section lists the status of each issue and a brief summary. The next section is the detailed description of the issue and the resolution. Please review this status and the detailed issues to see if you agree or disagree with the status so far. Silence will be interpreted as agreement.

Status codes:

AGREED - agreement on the suggested clarification, suggested change, or suggested. Subsequence silence on the DL will be interpreted as agreement. If you disagree, please indicate this to the ipp@pwg.org DL with the subject line containing: "MOD - Issue nn ...", where nn is the Issue number, and ... is the brief description of the issue.

OPEN - All 36 issues have been closed.

OPEN issues remaining: none.

1) ISSUE: Is 'application/octet-stream REQUIRED?

Suggested change: AGREED - no, change 1.1 back to agree with 1.0.

2) ISSUE: How can client force identified (authenticated) mode?

Possible alternatives: AGREED - Add a "uri-authentication-supported (1setOf type2 keyword)" REQUIRED Printer Description attribute that identifies the authentication mechanism associated with each URI listed in the "printer-uri-supported" attribute. Also add this attribute as a RECOMMENDED directory schema attribute in the Directory Appendix E.

IIG: Add examples that show using suffixes to the URL to make multiple URLs, when distinct URLs are needed..

3) ISSUE: How reject down stream auto-sensed unsupported PDL?

Suggested addition (similar addition for "compression" in Issue 6): AGREED - add 'unsupported-document-format' and 'document-format-error' job state reasons.

IIG: Add an example showing a PostScript Level 3 job being aborted by a PostScript Level 2 printer.

4) ISSUE: Client (desktop or server) closes slow channel

Suggested clarification (same as Issues 5 and 20): AGREED that client SHOULD NOT close channel, unless the layer that initiated the submission does the close.

IIG: Suggest that a client implementer avoid using synchronous writes, since they automatically close the channel. Use asynchronous writes instead, so that the lower layer doesn't time out the channel.

5) ISSUE: Client (desktop or server) closes stopped device

Suggested clarification (same as Issues 4 and 20): AGREED that client SHOULD NOT close channel, unless user indicates or policy..

IIG: Add examples.

6) ISSUE: What error if wrong compressed data supplied?

Suggested addition (similar addition for document-format in Issue 3; see related Issue 28): AGREED - add 'client-error-compression-error' status code and 'compression-error' and 'unsupported-compression' job state reasons.

7) ISSUE: Please implement Manufacturer make and model printer attribute and send the .INF file model name of the printer.

AGREED - Leave the description of "make" ambiguous in the Model.

Suggested clarification for the IIG: Document what Microsoft does with "printer-make-and-model". Document what any other platform does with this or similar attributes as suggested by participants.

8) ISSUE: In Model and Semantics 3.2.6.1, the definition for "limit", "which-jobs" and "my-jobs" is contradicting each other.

Suggested clarification: AGREED - clarify the "limit" limits the number so that the other two don't have to return ALL.

9) ISSUE: Customers become very unhappy when they go to the printer to pick up their job and a ream of PostScript source code is sitting in the output bin.

Suggested clarification: AGREED - clarify that application/octet-stream (auto-sense) can happen at submit time and/or processing time, depending on implementation. If auto-sense detects an unsupported document format at submit time, it returns the 'client-error-document-format-not-supported' error status code and rejects the create request.

10) ISSUE: How distinguish between submit vs processing auto-sense?

Suggested clarification in [ipp-mod] and [ipp-iig]: AGREED - clarify in [ipp-mod] that auto-sense MAY happen at either submit-time and/or processing-time. In IIG explain that with compression, it is much harder to auto-sense at submit time, since some compression methods require processing the entire file. Do NOT add a way for the client to determine whether auto-sensing happens at submit time or processing time.

11) ISSUE: Return what attributes with 'client-error-document-format-not-supported'?

Suggested clarification (see also Issues 18 and 23): AGREED - IPP/1.1 NEED NOT return "document-format=xxx" in Unsupported Attribute Group even though a special error status code, to make this error consistent with the rules for unsupported attributes.

12) ISSUE: length fields for the "UNSUPPORTED" tag

Suggested clarification (same as Issue 15): AGREED - clarify [ipp-mod] to agree with [ipp-pro] that the length MUST be 0 and no value is returned.

13) ISSUE: What job-state value should be returned in the Create-Job response?

Suggested clarification: AGREED - can be 'pending-held', 'pending', or 'processing' (the latter for a non-spooling printer that doesn't implement the 'pending' job state). Add 'job-data-insufficient' job-state-reason for use in any of the three job states if actual ripping or marking cannot begin until sufficient data has arrived.

Suggested clarification to IIG: AGREED - Explain the difference between the two job state reasons 'job-incoming' and 'job-data-insufficient', since both are likely to be meaningful for a spooling server.

14) ISSUE: Job-state for a forwarding server that can't get status from the device or system?

Suggested clarified and addition: AGREED - 'completed' is ok, but also add 'queued-in-device' job state reason which MUST be supported.

15) ISSUE: 'unknown' and 'unsupported' Out of band values.

Suggested clarification (same clarification as Issue 12): AGREED - clarify [ipp-mod] to agree with [ipp-pro] that the length MUST be 0 and no value is returned.

16) ISSUE: Get-Printer-Attributes Polling

Suggested clarification in the IIG: AGREED - Add to IIG that clients SHOULD request only the attributes needed, rather than always asking for all.

17) ISSUE: Client display of absolute time for job attributes?

Suggested change: Change "time-at-creation (integer(0:MAX))", "time-at-processing (integer(0:MAX))", and "time-at-processing (integer(0:MAX))" Job Description attributes range from 0:MAX to MIN:MAX so that negative times (or 0) MAY be used to indicate job events that happened before the most recent power-up. Change "time-at-created(integer(MIN:MAX))", "time-at-processing(integer(MIN:MAX))", and "time-at-completed(integer(MIN:MAX))" Job Description attributes from OPTIONAL to REQUIRED. Add REQUIRED job-printer-up-time(integer(1:MAX)) Job Description attribute. Add OPTIONAL "date-time-at-creation(dateTime)", "date-time-at-processing(dateTime), and "date-time-at-completed(dateTime). If a Printer resets its "printer-up-time" to 1 on power-up, it MUST change all persistent job time attributes to 0 or negative.

IIG: Indicate how any network printer can get time from NTP Time server. See RFC 1305. Also DHCP option 32 in RFC 2132 returns the IP address of the NTP server.

18) ISSUE: Return all Job Template errors on Print-Job fidelity=true

Suggested clarification (same clarification as Issue 27): AGREED - all unsupported Job Template attributes MUST be returned, not just the first, to agree with June IPP/1.0 draft. (In the November draft this requirement was moved to the IIG, which seems to have been a mistake).

19) ISSUE: User Performing the Send-Document Operation

Suggested clarification: AGREED - same user MUST do Send-Document as did Create-Job. Same security level or higher for subsequent operations on the job. Introduce the terms: "job owner" and "authenticated user".

20) ISSUE: Non-spooling printers accept/reject additional jobs

Suggested clarification (same as Issues 4 and 5): AGREED that IPP object MAY accept an implementation-defined number of subsequent create operations, including NONE.

IIG: Add warning to clients that an IPP Printer MAY either reject subsequent jobs and/or may accept some, but flow control them down.

21) ISSUE: Does 'none' "uri-security-supported" mean Basic/Digest?

Suggested clarification: AGREED - "uri-security-supported" does not cover this kind of HTTP authentication. Also add a note to refer to [ipp-pro] for authentication since some authentication is transport-dependent. And the new "uri-authentication-supported" attribute covers authentication. See Issue 2.

22) ISSUE: Status code on variable-length attributes that are 'too short'

Suggested clarification in the IIG: AGREED - clarify in IIG that no special processing is needed if a client supplied a keyword with 0 length, since the keyword will not match any "xxx-supported" keywords.

23) ISSUE: There seems to be some misunderstanding about the unsupported-attributes group.

Suggested clarification (related to Issues 11 and 18): AGREED - clarify that the IPP object MUST return only requested attributes that are unsupported.

24) ISSUE What status does Get-Jobs return when no jobs?

Suggested clarification: AGREED - MUST return 'successful-ok'.

25) ISSUE - MAY an IPP object return more Operation attributes?

Suggested clarification: AGREED - client MUST process or ignore additional operation attributes returned.

26) ISSUE: MAY an IPP object return additional groups?

Suggested clarification: AGREED - Yes, and a client MUST process or ignore additional attribute groups returned in any order.

27) ISSUE: Return first or all unsupported Job Template attributes in Unsupported Group?

Suggested clarification (same clarification as Issue 18): AGREED - all unsupported Job Template attributes MUST be returned, not just the first, to agree with June IPP/1.0 draft. (In the November draft this requirement was moved to the IIG, which seems to have been a mistake).

28) ISSUE: What if compression is supplied but not supported?

Suggested IPP/1.1 Change (related to Issues 3 and 6): AGREED - "compression" and "compression-supported" is REQUIRED for IPP/1.1 (with at least the 'none' value), even though it is OPTIONAL for IPP/1.0. Add the 'client-error-document-format-error' for error detected at request time with a supported document format, such as PostScript Level 3 not supported by a PostScript level 2 printer. Describe the priority between 'client-error-document-format-not-supported', 'client-error-compression-not-supported', 'client-error-document-format-error', and 'client-error-compression-error' status codes. Also add "compression-supported" to the Appendix E on directory schema as a RECOMMENDED attribute.

IIG only: IPP/1.0 implementations SHOULD at least check for the "compression" attribute being present and reject the create request, if they don't support "compression". Not checking is a bug, since the data will be unintelligible.

It was brought up that we need to check what compression HTTP supports and whether that would allow us to drop the "compression" attribute in IPP altogether (or use it only in Print-URI and Send-URI). The HTTP compression would have to work on POST.

29) ISSUE: Should "queued-job-count" be REQUIRED?

Suggested change: AGREED - The "queued-job-count" is REQUIRED for IPP/1.1; it is a SHOULD in the IPP/1.0 document.

30) ISSUE: Should "job-state-reasons" and "printer-state-reasons" be REQUIRED for an IPP/1.1 Printer?

Suggested change: AGREED - The "job-state-reasons" and "printer-state-reasons" will be REQUIRED for IPP/1.1; they are OPTIONAL in IPP/1.0.""

31) ISSUE: How indicate a ripped job that is waiting for the marker?

Suggested addition: AGREED - An implementation MAY use any of the following: job stays in 'processing', job moves to 'pending', job moves to 'pending-held' job states. Any of the alternatives MAY use a new 'queued-for-marker' job state reason to indicate that the job has been ripped but is waiting for the marker in a high end system. The 'pending-held' state is used by systems where the Operator explicitly does a Release-Job to schedule the next job to be marked, while the 'pending' or 'processing' state is used by systems that choose the next job to mark automatically. The 'processing' state is typically used by systems that tend not to have much time between ripping and marking.

Also need to clarify that more than one job can be in the 'processing' state at the same time when some are being ripped while one is being marked.

32) ISSUE: Is Digest REQUIRED for an IPP client and an IPP Printer to support?

Suggested change to Encoding and Transport document: AGREED -

1) Require an IPP Printer to at least implement either or both of:

a) HTTP Basic over a TLS secured channel (implementing TLS authentication is NOT REQUIRED), OR,
b) the client authentication part of HTTP Digest

2) Require clients to implement at least both of the above.

33) ISSUE: Include the IPP/1.0 conformance requirements in the IPP/1.1 document?

Suggested change: AGREED - No. The IPP/1.1 Model and Semantics document and the IPP/1.1 Encoding and Transport document will only cover IPP/1.1. They will NOT obsolete the experimental RFC that describes IPP/1.0.

The IPP/1.1 documents will say that for interoperability with IPP/1.0 clients, that an IPP Printer SHOULD accept IPP/1.0 requests and respond with IPP/1.0 responses.

The IPP/1.1 documents will NOT describe IPP/1.0 at all. However, the IPP/1.1 documents will contain an appendix that summarizes each difference from IPP/1.0 by section number and a brief description (see Issue 33 details below).

IIG: The IIG will discuss the advantages of a Printer supporting both IPP/1.0 and IPP/1.1 to maximize interoperability with clients. Also discuss the advantage of a client supporting both IPP/1.0 and IPP/1.1 to maximize interoperability with IPP Printers.'"'

34) ISSUE: Ok to REQUIRE "multiple-document-handling if Create-Job is supported?

Suggested change: Allow Create-Job and Send-Document to be supported even when only one document jobs are supported. Add a new "multiple-document-jobs-supported (boolean) Printer Description attribute to indicate whether or not multiple documents are supported.

35) ISSUE: What error code to return on Print-URI or Send-URI if document not accessible?

Suggested addition: Add both a new 'client-error-document-access-error' status code and a 'document-access-error' value for "job-state-reasons", just like we have done for compression and document format errors for Issue 3, 6, and 28.

36) ISSUE: Don't require 1.0 support and add REQUIRED "version-numbers-supported" attribute

Suggested addition: RECOMMEND, rather than REQUIRE, conforming IPP/1.1 clients and the IPP/1.1 Printers to support IPP/1.0 requests and responses. Therefore, add an "ipp-versions-supported" Printer Description attribute. Also add this attribute as RECOMMENDED in the directory schema list in the Appendix.

Detailed Descriptions of Issues and Resolutions or Alternatives.

1) ISSUE: Is 'application/octet-stream REQUIRED?

Is application/octet-stream REQUIRED. IPP/1.0 appears not to require it, while IPP/1.1 indicates "REQUIRED".

Suggested change:

Change IPP/1.1 Model and Semantics document back to agree with IPP/1.0 not to require support of the 'application/octet-stream' document format.

2) ISSUE: How can client force identified mode?

If an IPP Printer supports both authenticated and unauthenticated access, there is no way for a client to force itself to be authenticated, i.e., be in identified mode, since it is the server that forces authentication by issuing a challenge to the client. It is very useful for a client to be able to get into identified mode as soon as possible. Today you have to wait to be challenged by the server, which may never happen – or happens at an unpredictable time. The security conformance requires that the authentication for operations be the same for all operations. So for authenticated Cancel-Job, the Print-Job has to be authenticated as well. We would like to add another operation that forces the server to generate a 401 authentication challenge which the client would submit before submitting the print job in the first place. Unless somebody has a different solution (Microsoft)

Possible alternatives:

1. Add the operation as an OPTIONAL operation to IPP/1.0 and IPP/1.1 that forces the IPP object to issue a challenge to the client.

2. Use two URLs for the same IPP Printer object, one requires authentication and the IPP server always issues a challenge and the other never does. So the client that wants to be authenticated submits requests to the URL that requires authentication. ISSUE: How does the client discover which URL to use, since "uri-security-supported" is about security, not authentication?

3. Use two IPP Printer objects that fan-in to the same device. One IPP Printer object requires authentication and always issues the challenge and the other never does. ISSUE: How does the client discover which IPP Printer to use for authenticated access?

4. Request that the HTTP WG add some kind of header that allows the client to request that the HTTP server issue a challenge. ISSUE: It is unlikely that the HTTP group would do such a thing, since it is not needed for the usual use of HTTP which is to access documents on a server.

5. Some say that it isn't a problem that the client cannot force authentication.

Suggested addition:

Add the following REQUIRED Printer Description attribute (alternative #2 above):

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

This REQUIRED Printer attribute MUST have the same cardinality (contain the same number of values) as the "printer-uri-supported" attribute. This attribute identifies the authentication mechanism associated with each URI listed in the "printer-uri-supported" attribute. The Printer object uses the specified mechanism to identify the authenticated user. The "i th" value in "uri-authentication-supported" corresponds to the "i th" value in "printer-uri-supported" and it describes the authentication mechanisms associated with the URI. See [IPP-PRO] for more details on Client Authentication.

The following standard keyword values are defined:

'none': There is no authentication mechanism associated with the URI. The Printer object assumes that the authenticated user is "anonymous".
'requesting-user-name': When a client performs an operation whose target is the associated URI, The Printer object assumes that the authenticated user is specified by the "requesting-user-name" Operation attribute. If this attribute is absent, the Printer object assumes that the authenticated user is "anonymous".
'basic': When a client performs an operation whose target is the associated URI, the Printer object challenges the client with HTTP basic authentication. The Printer object assumes that the authenticated user is the name received via the basic authentication mechanism.
'digest': When a client performs an operation whose target is the associated URI, the Printer object challenges the client with HTTP digest authentication. The Printer object assumes that the authenticated user is the name received via the digest authentication mechanism.
'certificate': When a client performs an operation whose target is the associated URI, the Printer object expects the client to provide a certificate. The Printer object assumes that the authenticated user is the textual name contained within the certificate.

3) ISSUE: How reject down stream auto-sensed unsupported PDL?

If auto-sensing happens AFTER the job is accepted (as opposed to auto-sensing at submit time before returning the response), what does the implementation do?

Presumably, it is similar to encountering a mal-formed PDL. So the implementation aborts the job, puts the job in the 'aborted' state and sets the 'aborted-by-system' value in the job's "job-state-reasons". ""The 'aborted-by-system' value seems appropriate, but it would be good to have a more specific reason to indicate the reason that the job was aborted by the system.

Suggested addition (similar addition for "compression" in Issue 6):

Add 'unsupported-document-format' as a "job-state-reasons" value for use when the job is aborted because the document format that is auto-sensed is not a supported document format. Also add a 'document-format-error' as a "job-state-reasons" value for use when the job is aborted because any kind of PDL error is encountered while processing the document.

Suggested text:

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

'document-format-error': The job was aborted by the system because the Printer encountered an error in the document-data while processing it. If the Printer posts this reason, the document-data has already passed any tests that would have led to the 'unsupported-document-format' job-state-reason.

4) ISSUE: Client (desktop or server) closes slow channel

Some IPP Printer implementations, such as forwarding servers, want to accept an IPP job, even though the down stream channel is being used at the moment by another job stream that the device supports. Rejecting the job would mean that an IPP job might never get in, since these other protocols queue the request.

However, some clients close the channel when it is flowed controlled off for too long a time?

Suggested clarification (same as Issues 5 and 20):

Clarify the IPP/1.1 Model and Semantics document that Clients (desktop or server) SHOULD NOT close the channel when flowed controlled off, unless the layer that initiated the submission does the close. Clients SHOULD do Get-Printer-Attributes and determine state of the device. Alert user if the printer is stopped. Let user decide whether to abort the job transmission or not.

IIG: Suggest that a client implementer avoid using synchronous writes, since they automatically close the channel. Use asynchronous writes instead, so that the lower layer doesn't time out the channel.

Also clarify the IPP/1.1 Model and Semantics document that the following actions are conforming for non-spooling IPP Printer objects: After accepting a create job operation, a non-spooling IPP Printer MAY either:

1. Reject any subsequent create job operations while it is busy transferring and/or processing an accepted job request and return the 'server-error-busy (0x0507).

2. Accept up to some implementation-defined subsequent create job operations and flow control them to prevent buffer overflow. When the implementation-defined number of jobs is exceeded, the IPP Printer MUST return the 'server-error-busy' status code and reject the create job request as in 1 above.

Client (desktop or server) SHOULD NOT close the channel when flow controlled off, unless the layer that initiated the submission does the close. Clients that are rejected with a 'server-error-busy' status code MAY retry periodically, try another IPP Printer, and/or subscribe for a 'ready-for-job' event when we have notification specified.

Clarify that a client may be either in a desktop under control of a user or in a server that accepts some protocol (IPP or other) and uses IPP to controls printers.

Suggested text for section 2.1 IPP Objects:

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

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

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

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

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

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

Suggested text for section 5.1 Client Conformance Requirements:

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

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

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

While a client is sending data to a printer, it SHOULD do its best to prevent a channel from being closed by a lower layer when the channel is blocked (i.e. flow-controlled off) for whatever reason, e.g. 'out of paper' or 'job ahead hasn't freed up enough memory'. However, the layer that launched the print submission (e.g. an end user) MAY close the channel in order to cancel the job. When a client closes a channel, a Printer MAY print all or part of the received portion of the document. See the "Encoding and Transport" document [IPP-PRO] for more details.

Suggested text for section 5.2 IPP Object Conformance Requirements:

This section specifies the conformance requirements for conforming implementations of IPP objects (see section 2). These requirements apply to an IPP object whether it is:

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

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

5) ISSUE: Client (desktop or server) closes stopped device

When a non-spooling printer is accepting data and putting it on media and runs into a problem, such as paper out or paper jam, what should it do?

Returning an error is not user friendly, if fixing the problem would allow the job to complete normally.

Suggested clarification (same as Issues 4 and 20):

Clarify the IPP/1.1 Model and Semantics document that IPP Printers MUST not return an error status code during a Print-Job operation when a device problem, such as jam or out of paper. Instead, the IPP Printer object flow controls the data off. Otherwise, only a partial job will be produced, when a whole job would be produced when the problem is attended to.

Clients (desktop or server) SHOULD NOT close the channel when flow controlled off, unless the layer that initiated the submission does the close. Clients SHOULD do Get-Printer-Attributes and determine state of the device. Alert user if the printer is stopped. Let user decide whether to abort the job transmission or not.

IIG: Add examples.

Suggested text for section 5.1 Client Conformance Requirements:

While a client is sending data to a printer, it SHOULD do its best to prevent a channel from being closed by a lower layer when the channel is blocked (i.e. flow-controlled off) for whatever reason, e.g. 'out of paper' or 'job ahead hasn't freed up enough memory'. However, the layer that launched the print submission (e.g. an end user) MAY close the channel in order to cancel the job. When a client closes a channel, a Printer MAY print all or part of the received portion of the document. See the "Encoding and Transport" document [IPP-PRO] for more details.

6) ISSUE: What error if wrong compressed data supplied?

Problem: IPP server supports 'deflate' and 'gzip'. If client sets "compression attribute" = 'deflate' but sends gziped data, what error does IPP server return to client? Cannot use the existing 'client-error-attributes-or-values-not-supported' (0x040B). But returning the operation attribute with the value that was sent ('deflate') would be incorrect, because 'deflate' is supported!

Suggested addition (similar addition for document-format in Issue 3; see related Issue 28):

Add a new error status code: 'client-error-compression-error' that the IPP object can return if the compression error is detected before the create job response is returned. Also add 'compression-error' as a "job-state-reason" value for use when the job is aborted because any kind of compression error is detected while decompressing the data after the create job response has been returned to the client.

The new 'client-error-compression-error' (0x0410) status code definition is:

The IPP object is refusing to service the request because the document data cannot be decompressed when using the algorithm specified by the "compression" operation attribute. This error is returned independent of the client-supplied "ipp-attribute-fidelity". The Printer object MUST return this status code, even if there are other attributes that are not supported as well, since this error is a bigger problem than with Job Template attributes.

The suggested new job state reason definitions are:

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

'compression-error': The job was aborted by the system because the Printer encountered an error in the document-data while decompressing it. If the Printer posts this reason, the document-data has already passed any tests that would have led to the 'document-access-error' or 'unsupported-compression' job-state-reasons.

7) ISSUE: Please implement Manufacturer make and model printer attribute and send the .INF file model name of the printer.

If you do this we will automatically install the correct driver (if we have it) (Microsoft)

Suggested clarification for the IIG:

At the front of the Implementer's Guide, indicate that implementation considerations that relate to particular operating system and NOS will be incorporated as they become known. Add recommendation to the IPP/1.1 Implementer's Guide that printer vendors are encouraged to configure the IPP Printer's "printer-make-and-model" attribute with the make and model name that matches the .INF file on Microsoft platforms. When so configured, the Microsoft driver install program will skip asking the user for the make and model of the printer being installed and use the value of the "printer-make-and-model" attribute.

""Do not attempt to clarify the "printer-make-and-model" attribute as to whether it includes a vendor name or not.

8) ISSUE: In IPP/1.0 Model and semantics 3.2.6.1, the definition for "limit", "which-jobs" and "my-jobs" is contradicting each other.

The problem is that the definition for "which-jobs" and "my-jobs" states that "all" jobs MUST be returned, while "limit" restricts the number of jobs to be returned. (Stefan Andersson Axis Communication AB)

Suggested clarification:

Clarify IPP/1.1 Model and Semantics "which-jobs" and "my-jobs" operation attributes to indicate that the number of jobs returned is limited by the "limit" attribute if supplied by the client.

Suggested text for section 3.2.6.2 Get-Jobs Response

In the first sentence add the phrase:

up to the number specified by the "limit" attribute

to give:

> The Printer object returns all of the Job objects up to the number specified by the "limit" attribute that match the criteria as defined by the attribute values supplied by the client in the request.

9) ISSUE: Customers become very unhappy when they go to the printer to pick up their job and a ream of PostScript source code is sitting in the output bin.

Cause: A PostScript datastream is accidentally sent to a PCL printer.

IPP Issue: IPP needs to clarify the standard in section 3.2.1.1 of the Model and Semantics document. Lines 1219-1221 defining the "document-format" operation attribute state that:

> If the client does not supply the [document format] attribute, the Printer object assumes that the document data is in the format defined by the Printer object's "document-format-default" attribute.

I would like to see the following clarification:

> If the client does not supply the [document format] attribute and the Printer object is not able to auto-sense the document format at print-job request time, the Printer object assumes that the document data is in the format defined by the Printer object's "document-format-default" attribute.

If the Printer object senses that the document format is PostScript, then job should be rejected if it is being sent to a PCL-only printer. The 'application/octet-stream' mechanism discussed in section 4.1.9 does not seem to be helpful in this case, because it appears to assume that the auto-sensing occurs at document processing time. Until the document is actually "ripped", the document format remains unknown. So it seems to me that lines 2453-2476 do not address the problem described above where the wrong document format is submitted. These lines, rather, seem to apply to the case of a printer that handles multiple document formats and assumes that the submitted document is in one of the supported formats.

Suggested clarification:

Add the suggested clarification that auto-sensing MAY be done at either job-submission time and/or job processing time to the IPP/1.1 Model and Semantics documents.

Suggested text for a new section 4.1.9.1 Application/octet-stream -- Auto-Sensing the document format:

During auto-sensing, a Printer may determine that the document-data has a format that the Printer doesn't recognize. If the Printer determines this problem before returning an operation response, it rejects the request and returns the 'client-error-document-format-not-supported' status code. If the Printer determines this problem after accepting the request and returning an operation response with one of the successful status codes, the Printer adds the 'unsupported-document-format' value to the job's "job-state-reasons" attribute.

10) ISSUE: How distinguish between submit vs processing auto-sense?

There are two different implementations of auto-sensing:

- at print submit time BEFORE the Print-Job or Send-Document responds
- at document processing (ripping) time AFTER the Print-Job or Send-Document has accepted the job and returned the response.

The description of 'application/octet-stream' doesn't clarify whether one, the other or both is meant. How can a client determine which is supported?

Suggested clarification in [ipp-mod] and [ipp-iig]:

Clarify IPP/1.1 Model and Semantics document that 'application/octet-stream' means either auto-sensing at job submission time and/or job processing time depending on implementation. Do NOT add a way for the client to determine whether auto-sensing happens at submit time or processing time.

Add to Implementer's Guide a discussion about the advantages of auto-sensing at job submit time, rather than waiting until job processing time, so that an IPP Printer can reject an unsupported document format instead of accepting the job and then aborting the job sometime later. Also discuss for print by reference that an IPP Printer may want to examine the file, at least the first few octets, in order to check that the document-format is supported. On the other hand, network delays may make such a strategy take too long. Alternatively, the client may want to supply the "document-format" explicitly when doing print-by-reference either using the file extension as a hint, or actually accessing the first few octets of the data an implementing an auto-sensing in the client.

Suggested text for section 4.1.9 mimeMediaType:

One special type is 'application/octet-stream'. If the Printer object supports this value, the Printer object MUST be capable of auto-sensing the format of the document data, either as part of the create operation and/or at document processing time.

11) ISSUE: Return what attributes with document-format-not-supported?

If a server receives a request with a document format which is not supported, it returns the client-error-document-format-not-supported (0x040A) status code. Is it also necessary to include document format in the unsupported attribute group?

We suggest adding text which says it NEED NOT be supplied in the unsupported group.

Suggested clarification (see also Issues 18 and 23):

Clarify IPP/1.1 Model and Semantics document that when returning the 'client-error-document-format-not-supported' in a create response or a Send-Document response, that IPP/1.1 NEED NOT return "document-format=xxx" in Unsupported Attribute Group since there is a special error status code.

Suggested clarification for section 13.1.4.11 client-error-document-format-not-supported

13.1.4.11 client-error-document-format-not-supported (0x040A)

The IPP object is refusing to service the request because the document data is in a format, as specified in the "document-format" operation attribute, that is not supported by the Printer object. This error is returned independent of the client-supplied "ipp-attribute-fidelity". The Printer object MUST return this status code, even if there are other Job Template attributes that are not supported as well, since this error is a bigger problem than with Job Template attributes. See section 0. Issue 11

12) ISSUE: length fields for the "UNSUPPORTED" tag

IPP/1.0: Model and Semantics, 16 Nov 1998, 3.2.1.2, Group 2 (unsupported attributes) -- states that in the case of an unsupported attribute name, the printer object should return a substituted out of band value of "unsupported". This impression is strengthened by the reference to section 4.1, where it gives the legal out of band values, none of which is an empty string.

This appears to conflict with Internet Printing Protocol/1.0: Encoding and Transport, 16 Nov 1998, section 3.10, where it states that the value length must be 0 and the value empty. (Claudio Cordova, Wade Mergenthal Xerox Corp.)

Suggested clarification (same as Issue 15):

Clarify the IPP/1.1 Model and Semantics document so that it does not appear to contradict the Encoding and Transport document. However, whether each of the "out-of-band" values are encoded as distinct attribute syntaxes with no value or as a single attribute syntax with a value that indicates which out-of-band value, is purely an encoding matter and cannot be indicated in the Model and Semantics document. Therefore, indicate in the IPP/1.1 Model and Semantics document that the reader is to refer to the IPP/1.1 Encoding and Transport document for the encoding of the out-of-band values.

Suggested text for section 3.1.7:

This value's syntax type is "out-of-band" and its encoding is defined by special rules for "out-of-band" values in the "Encoding and Transport" document [IPP-PRO]. Its value indicates no support for the attribute itself (see the beginning of section 4.1).

Suggested text for section 4.1:

In addition, the value of an attribute in a response (but not in a request) MAY be one of the "out-of-band" values whose special encoding rules are defined in the "Encoding and Transport" document [IPP-PRO].

13) ISSUE: What job-state value should be returned in the Create-Job response?

Pending, pending-held, or either depending on implementation?

The problem with 'pending' is that the job is not a "candidate to start processing" as the definition states. The 'pending-held' state seems more reasonable. Its definition is:

> 'pending-held': The job is not a candidate for processing for any number of reasons but will return to the 'pending' state as soon as the reasons are no longer present. The job's "job-state-reason" attribute MUST indicate why the job is no longer a candidate for processing.

Also there is a "job-state-reason" value 'job-incoming' which states:

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

""Suggested clarification:

Clarify the IPP/1.1 Model and Semantics document that an IPP Printer MAY put the job into the 'pending', 'pending-held', or 'processing' states after a Create-Job, depending on implementation as follows:

- 'pending' - if the job is a candidate for processing whether all of the document data is present or not. Add the 'waiting-for-data' "job-state-reasons" value to the job as an indication why this 'pending' job is not being processed OR
- 'pending-held' - if the job is not a candidate for processing until the last Send-Document or Send-URI operation has been performed with the "last-document" set to 'true' and the document data transferred. Here the implementation SHOULD set ""the 'job-incoming' value of the "job-state-reasons" attribute until the last data has arrived. The IPP Printer removes the 'job-incoming' value when the last data has arrived, and transitions the job from the 'pending-held' to the 'pending' job state OR
- 'processing' - if the IPP Printer is a non-spooling printer that does not implement the 'pending' state, i.e., it either accepts a job and processes it or rejects the job if it already processing a job. However, if a non-spooling printer does accept additional jobs while processing a job, then the additional jobs MUST NOT be put into the 'processing' state immediately. See Issue 20 resolution for non-spooling printers.

Suggested text addition to section 3.2.4 Create-Job operation:

After the Create-Job operation has completed, the value of the "job-state" attribute is similar to the "job-state" after a Print-Job, even though there is no document-data. A Printer MAY set the 'job-data-insufficient' value of the job's "job-state-reason" attribute to indicate that processing cannot begin until sufficient data has arrived and set the "job-state" to either 'pending' or 'pending-held'. A non-spooling printer that doesn't implement the 'pending' job state MAY even set the "job-state" to 'processing', even though there is not yet any data to process. See sections 4.3.7 and 4.3.8.

Suggested text addition to section 4.3.8 job-state-reasons:

Add the 'job-data-insufficient' value to be used with "job-state-reasons" with the following definition:

'job-data-insufficient': The Create-Job operation has been accepted by the Printer, but the Printer is expecting additional document data before it can move the job into the 'processing' state. If a Printer starts processing before it has received all data, the Printer removes the 'job-data-insufficient' reason, but the 'job-incoming' remains. If a Printer starts processing after it has received all data, the Printer removes the 'job-data-insufficient' reason and the 'job-incoming' at the same time.

Suggested clarification to IIG: AGREED - Explain the difference between the two job state reasons 'job-incoming' and 'job-data-insufficient', since both are likely to be meaningful for a spooling server.

Note: Change the Bake Off 2 bo38.test script so that the 'pending-held', the 'pending', or 'processing' job state is expected after a Create-Job operation.

14) ISSUE: Job-state for a forwarding server?

What job-state value should be returned in the Print-Job response for an IPP object that forwards the data over a one-way interface, such as a parallel port or LPD? pending, processing, completed, or unknown?

Unknown is the strict interpretation of section 4.3.7 "job-state", but it isn't very user friendly. The "job-state" SHOULD reflect the actual job state, but these implementations have no idea when the job actually starts or finishes.

How about a new "job-state-reasons" value: 'queued-in-device' (from PWG Job Monitoring MIB)?

Suggested addition:

Add to the IPP/1.1 Model and Semantics document the 'queued-in-device' value for use with the "job-state-reasons" attribute. REQUIRE that an IPP/1.1 implementation that forwards jobs, but does not have any means to query the state of the down stream job, MUST support the ""the new 'queued-in-device' value of the REQUIRED "job-state-reasons" attribute when returning the job in the 'completed' state. """

Suggested text for section 4.3.7 job-state:

Add the following qualification to the "job-state" description:

As with all other IPP attributes, if the implementation can not determine the correct value for this attribute, it SHOULD respond with the out-of-band value 'unknown' (see section 4.1) rather than try to guess at some possibly incorrect value and give the end user the wrong impression about the state of the Job object. For example, if the implementation is just a gateway into some printing system from which it can normally get status, but temporarily is unable, then the implementation should return the 'unknown' value. However, if the implementation is a gateway to a printing system that never provides detailed status about the print job, the implementation MAY set the IPP Job object's state to 'completed', provided that it also sets the 'queued-in-device' value in the job's "job-state-reasons" attribute (see section 4.3.8).

Suggested text for section 4.3.8 job-state-reasons:

'queued-in-device': The job has been forwarded to a device or print system that is unable to send back status. The Printer sets the job's "job-state " attribute to 'completed' and adds the 'queued-in-device' value to the job's "job-state-reasons" attribute to indicate that the Printer has no additional information about the job and never will have any better information.

15) ISSUE: 'unknown' and 'unsupported' Out of band values.

It is very unclear from the spec as to whether or not you should use the word 'unknown' (or unsupported in that case) as the value for attributes that are unknown.

You can read it that you set the length equal to zero and set the type to 'unknown'. You can also read it as saying you set the value to the string 'unknown'.

This is not helped by the Transport and Encoding spec saying – you must set the length to zero and then telling a client what to do with a non-zero length. (Microsoft)

Suggested clarification (same clarification as Issue 12):

Clarify the IPP/1.1 Model and Semantics document so that it does not appear to contradict the Encoding and Transport document. However, whether each of the "out-of-band" values are encoded as distinct attribute syntaxes with no value or as a single attribute syntax with a value that indicates which out-of-band value, is purely an encoding matter and cannot be indicated in the Model and Semantics document. Therefore, indicate in the IPP/1.1 Model and Semantics document that the reader is to refer to the IPP/1.1 Encoding and Transport document for the encoding of the out-of-band values.

Suggested text for section 3.1.7:

This value's syntax type is "out-of-band" and its encoding is defined by special rules for "out-of-band" values in the "Encoding and Transport" document [IPP-PRO]. Its value indicates no support for the attribute itself (see the beginning of section 4.1).

Suggested text for section 4.1:

In addition, the value of an attribute in a response (but not in a request) MAY be one of the "out-of-band" values whose special encoding rules are defined in the "Encoding and Transport" document [IPP-PRO].

16) ISSUE: Get-Printer-Attributes Polling

Some client polls printer periodically by Get-Printer-Attributes without specifying "requested-attributes". So printer has to reply all attributes. It consumes printer resource.

Suggested clarification in the IIG:

RECOMMEND in the IPP/1.1 Implementer's Guide that Clients should specify "requested-attributes", if it wants to get just the printer status.

17) ISSUE: Client display of absolute time for job attributes?

What are clients doing with printers that don't support absolute time? How can client display an absolute time that a job was submitted, started processing, and completed (which is what is useful for a user)?

Possible Solution

Get Uptime from printer ("printer-up-time" - time system has been up in seconds)

Get Job(s)

Calculate Display time = job tick time ("time-at-xxx" - in seconds that system has been up) – uptime ("printer-up-time") + local client absolute date and time.   The down side is that the client has to get the "printer-up-time" every time with a separate Get-Printer-Attributes operation.

Alternatively: Add OPTIONAL job attributes: "date-time-at-creation (dateTime)", "date-time-at-processing (dateTime)", and "date-time-at-completion (dateTime)"

(Microsoft)

Possible alternatives:

One or more of the following alternatives:

1. Allow the "time tick" job time attributes of jobs that persist across power-ups to be negative, so that they could represent the time of an event that happened before the most recent power up: "time-at-creation (integer(MIN:MAX))", "time-at-processing (integer(MIN:MAX))", and "time-at-completion ((MIN:MAX))"

2. Add to the IPP/1.1 Model and Semantics document OPTIONAL job description attributes: "date-time-at-creation (dateTime)", "date-time-at-processing (dateTime)", and "date-time-at-completion (dateTime)".

3. Instead of adding new dateTime job attributes, just add the dateTime attribute syntax as a second choice for the existing job attributes changing them to:

   "time-at-creation (integer | dateTime)", "time-at-processing (integer | dateTime)", and "time-at-completion (integer | dateTime)"

4. Same as 1, but make the job attributes be REQUIRED for IPP/1.1.

5. Same as 2, but make the job attributes be REQUIRED for IPP/1.1, but keep support of the dateTime OPTIONAL.

6. Same as 3, but make the time tick job attributes be REQUIRED for IPP/1.1, and REQUIRE a Printer implementation attempt to get the dateTime from somewhere (person or the network) at startup time. The implementation MUST use the integer form when the date cannot be obtained from a person or the network at startup time.

7. Same as 3, but make support of the dateTime REQUIRED for IPP/1.1.

8. Add three new "delta-time-at-xxx(integer)" where the value is the number of seconds in the past that the event occurred. In other words, the server does the subtract of:

   job tick time ("time-at-xxx" - in seconds that system has been up) – uptime ("printer-up-time") at query time, so that the client doesn't have to also query the Printer Description "printer-up-time" at all. Then the client just subtracts the value from the client's current local absolute date and time.

9. Return "printer-up-time" (in seconds) as an operation attribute in Get-Jobs and Get-Job-Attributes response.

10. Make the "printer-up-time" Printer Description attribute also be a Job Description attribute. Clients that request the "time-at-xxx" job attributes should also request the "printer-up-time" job attribute, so that they can avoid requesting it using a separate Get-Printer-Attributes request.

11. Add a REQUIRED "job-printer-up-time" Job Description attribute which is a copy of the IPP/1.0 REQUIRED "printer-up-time" Printer Description attribute.

Suggested resolution:

1. Change the range on the 3 "time-at-xxx" job time attributes from 0:MAX as it is in IPP/1.0 to MIN:MAX:

time-at-creation(integer(MIN:MAX))
time-at-processing(integer(MIN:MAX))
time-at-completed(integer(MIN:MAX))

A negative value indicates an event that happened that many seconds before the most recent power-up of the Printer; a 0 value means that the event occurred at some unspecified time before the printer was powered up most recently. Describe the 0 and negative values once in the time-at-xxx section.

2. Clarify the current section 4.4.26 printer-up-time(integer(1:MAX)) with respect to restarts. If the IPP/1.0 Printer resets the "printer-up-time" on power-up, it MUST reset the "time-at-xxx" Job time attributes for any persistent jobs back to 0 to indicate that the event took place sometime before the most recent power-up or to a negative value that represents the number of seconds before the most recent power-up that the event took place However, retain the IPP/1.0 implementation option to keep the "printer-up-time" counting higher on restarts; then the job's "time-at-xxx" MUST NOT be reset.

3. Problem: Make it easier for clients to get clock time for job events, make it easier for clients to correlate job events with notifications which need to use date and time (since there may not be intermediate servers to translate relative tick time to absolute date/time), allow the Printer to not have to adjust the time attribute values of all the persistent jobs on power-up, avoid the need for intermediate IPP servers to translate relative tick time as responses are cascaded back to the original client.

Solution: add three new OPTIONAL dateTime attribute syntax Job Description attributes:

date-time-at-creation(integer(MIN:MAX) | dateTime)
date-time-at-processing(integer(MIN:MAX) | dateTime)
date-time-at-completed(integer(MIN:MAX) | dateTime)

Thus the value returned is the Printer's "printer-current-time(dateTime)" when the event occurred. Now the client simply requests whichever of these attributes and deal with which ever attributes it gets back.

Clarify that the date and time does not have to be very accurate. The time does not have to be that precise in order to work in practice.

If an implementation cannot get the dateTime, then it MUST return the out-of-band 'no-value' value.

4. To solve the problem of the client having to make two trips to the printer when displaying jobs:

first to get the "time-at-xxx" job attributes with Get-Jobs or Get-Job-Attributes, and

second to get the "printer-up-time" with Get-Printer-Attributes,

we'll add a REQUIRED job attribute:

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

which is an alias for the Printer's "printer-up-time(integer(1:MAX))".

5. To help clients being able to depend on getting time tick, increase the conformance requirements: change the 3 "time-at-xxx(integer)" job time attributes from OPTIONAL to REQUIRED. This shouldn't be a burden, since the corresponding printer attribute: "printer-up-time" is already REQUIRED in IPP/1.0. Also the draft Printer MIB and MIB-II require that a device have a clock tick capability.

6. Clarify that if an implementation supports the OPTIONAL "printer-current-time(dateTime)" attribute by getting the time from some source such as the network or an operator, but was unable to, that it MUST return the out-of-band 'no-value' which means not configured (yet). See the beginning of section 4.1 in the Model.

7. Clarify that the time zone NEED NOT be that used by people in the vicinity of the Printer or device and that clients SHOULD convert dateTime attributes to the time zone of the client before display to the user.

8. Clarify that the "time-at-processing" is the first time the job begins processing after the create operation or the most recent Restart-Job operation.

IIG: Describe some of the many ways that implementations can get the date and time:

1. Any network printer can get time from NTP Time server. See RFC 1305. Also DHCP option 32 in RFC 2132 returns the IP address of the NTP server.
2. Get the date and time at startup from a human operator
3. Have an operator set the date and time using a web administrative interface
4. Get the date and time from incoming HTTP requests, though the problems of spoofing need to be considered. Perhaps comparing several HTTP requests could reduce the chances of spoofing.
5. Internal date time clock battery driven.
6. Query "http://tycho.usno.navy.mil/cgi-bin/timer.pl"

Suggested text for section 4.1.14 dateTime

4.1.14 'dateTime'

The 'dateTime' attribute syntax is a standard, fixed length, 11 octet representation of the "DateAndTime" syntax as defined in RFC 1903 [RFC1903]. RFC 1903 also identifies an 8 octet representation of a "DateAndTime" value, but IPP objects MUST use the 11 octet representation. A user interface will provide a mapping between protocol dateTime values and displayable user-friendly words or presentation values and phrases which are localized to the natural language and date format of the user, including time zone. Issue 17

Suggested text for the table in section 4.3:

time-at-creation	integer (MIN:MAX)	REQUIRED
time-at-processing	integer (MIN:MAX)	REQUIRED
time-at-completed	integer (MIN:MAX)	REQUIRED
job-printer-up-time	integer (1:MAX)	REQUIRED
date-time-at-creation	dateTime	OPTIONAL
date-time-at-processing	dateTime	OPTIONAL
date-time-at-completed	dateTime	OPTIONAL

Suggested text for a new section 4.3.12 Event Time Job Description Attributes:

Group the three "time-at-xxx" Job Description time attributes into a single section so that the common semantics can be said once:

4.3.12 Event Time Job Description Attributes

This section defines the Job Description attributes that indicate the time at which certain events occur for a job. If the job event has not yet occurred, then the IPP object MUST return the 'no-value' out-of-band value (see the beginning of Section 4.1). The "time-at-xxx(integer)" attributes represent time as an 'integer' representing the number of seconds since the device was powered up (informally called "time ticks"). The "date-time-at-xxx(dateTime)" attributes represent time as 'dateTime' representing date and time (including an offset from UTC).

In order to populate these attributes, the Printer object copies the value(s) of the following Printer Description attributes at the time the event occurs:

1. the value in the Printer's "printer-up-time" attribute for the "time-at-xxx(integer)" attributes

2. the value in the Printer's "printer-current-time" attribute for the "date-time-at-xxx(dateTime)" attributes.

If the Printer resets its "printer-up-time" attribute to 1 on power-up (see section 4.4.29) and has persistent jobs, then it MUST change all of the jobs' "time-at-xxx(integer)" (time tick) job attributes whose events have occurred either to:

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

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

Note: A Printer does not change the values of any "date-time-at-xxx(dateTime)" job attributes on power-up.

4.3.12.1 time-at-creation (integer(MIN:MAX))

This REQUIRED attribute indicates the time at which the Job object was created.

4.3.12.2 time-at-processing (integer(MIN:MAX))

This REQUIRED attribute indicates the time at which the Job object first began processing after the create operation or the most recent Restart-Job operation. The out-of-band 'no-value' value is returned if the job has not yet been in the 'processing' state (see the beginning of Section 4.1).

4.3.12.3 time-at-completed (integer(MIN:MAX))

This REQUIRED attribute indicates the time at which the Job object completed (or was cancelled or aborted). The out-of-band 'no-value' value is returned if the job has not yet completed, been canceled, or aborted (see the beginning of Section 4.1).

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

This REQUIRED Job Description attribute indicates the amount of time (in seconds) that the Printer implementation has been up and running. This attribute is an alias for the "printer-up-time" Printer Description attribute (see Section 4.4.27).

A client MAY request this attribute in a Get-Job-Attributes or Get-Jobs request and use the value returned in combination with other requested Event Time Job Description Attributes in order to display time attributes to a user. The difference between this attribute and the 'integer' value of a "time-at-xxx" attribute is the number of seconds ago that the "time-at-xxx" event occurred. A client can compute the wall-clock time at which the "time-at-xxx" event occurred by subtracting this difference from the client's wall-clock time.

4.3.12.5 date-time-at-creation (dateTime)

This attribute indicates the date and time at which the Job object was created.

4.3.12.6 date-time-at-processing (dateTime)

This attribute indicates the date and time at which the Job object first began processing after the create operation or the most recent Restart-Job operation.

4.3.12.7 date-time-at-completed (dateTime)

This attribute indicates the date and time at which the Job object completed (or was cancelled or aborted).

Suggested text for section 4.4.27 printer-up-time

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

This REQUIRED Printer attribute indicates the amount of time (in seconds) that this Printer instance has been up and running. The value is a monotonically increasing value starting from 1 when the Printer object is started-up (initialized, booted, etc.). This value is used to populate the Event Time Job Description Job attributes "time-at-creation", "time-at-processing", and "time-at-completed" (see Section 4.3.12).

If the Printer object goes down at some value 'n', and comes back up, the implementation MAY:

1. Know how long it has been down, and resume at some value greater than 'n', or
2. Restart from 1. In other words, if the device or devices that the Printer object is representing are restarted or power cycled, the Printer object MAY continue counting this value or MAY reset this value to 1 depending on implementation. However, if the Printer object software ceases running, and restarts without knowing the last value for "printer-up-time", the implementation MUST reset this value to 1. If this value is reset and the Printer has persistent jobs, the Printer MUST reset the "time-at-xxx(integer) Event Time Job Description attributes according to Section 4.3.12. Issue 17 An implementation MAY use both implementation alternatives, depending on warm versus cold start, respectively.

Suggested text for section 4.4.28 printer-current-time:

4.4.30 printer-current-time (dateTime)

This Printer attribute indicates the current date and time. This value is used to populate the Event Time Job Description attributes: "time-at-creation", "time-at-processing", and "time-at-completed" (see Section 4.3.12).

The date and time is obtained on a "best efforts basis" and does not have to be that precise in order to work in practice. A Printer implementation sets the value of this attribute by obtaining the date and time via some implementation-dependent means, such as getting the value from a network time server, initialization at time of manufacture, or setting by an administrator. See [IPP-IIG] for examples. If an implementation supports this attribute and the implementation knows that it has not yet been set , then the implementation MUST return the value of this attribute using the out-of-band 'no-value' meaning not configured. See the beginning of section 4.1.

The time zone of this attribute NEED NOT be the time zone used by people located near the Printer object or device. The client MUST NOT expect that the time zone of any received 'dateTime' value to be in the time zone of the client or in the time zone of the people located near the printer.

The client SHOULD display any dateTime attributes to the user in client local time by converting the 'dateTime' value returned by the server to the time zone of the client, rather than using the time zone returned by the Printer in attributes that use the 'dateTime' attribute syntax.

18) ISSUE: Return all Job Template errors on Print-Job fidelity=true

If ipp-attributes-fidelity=true, MUST all Job Template attributes that are not supported, be returned, or can just the first error be returned? Section 16.3 and 16.4 of the Model and Semantics document was moved to the Implementer's Guide when creating the November 1998 draft from the June 1998 draft. The following note was contained in section 16.4 that was moved:

Note: whether the request is accepted or rejected is determined by the value of the "ipp-attribute-fidelity" attribute in a subsequent step, so that all Job Template attribute supplied are examined and all unsupported attributes and/or values are copied to the Unsupported Attributes response group.

Suggested clarification (same clarification as Issue 27):

Clarify in the IPP/1.1 Model and Semantics document that all operation attributes and all Job Template attributes MUST be returned in the Unsupported Attributes group, unless there is a specific error status for the unsupported operation attribute, such as: server-error-version-not-supported, server-error-operation-not-supported, client-error-charset-not-supported, client-error-compression-not-supported, client-error-document-format-not-supported, and client-error-uri-scheme-not-supported".

Suggested text for section 3.1.6 Status Codes and a new section 3.1.7:

If the Printer performs an operation with no errors and it encounters no problems, it MUST return the status code 'successful-ok' in the response. See section 14.

If the client supplies unsupported values for the following parameters or Operation attributes, the Printer object MUST reject the operation, NEED NOT return the unsupported attribute value in the Unsupported Attributes group, and MUST return the indicated status code:

Parameter/Attribute	Status code
version-number	server-error-version-not-supported
operation-id	server-error-operation-not-supported

attributes-charset	client-error-charset-not-supported
compression	client-error-compression-not-supported
document-format	client-error-document-format-not-supported
document-uri	client-error-uri-scheme-not-supported, client-error-document-access-error

If the client supplies unsupported values for other attributes, or unsupported attributes, the Printer returns the status code defined in section 3.1.7 on Unsupported Attributes.

3.1.7 Unsupported Attributes

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

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

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

a) an Operation or Job Template attribute supplied in the request, and

b) unsupported by the printer. See below for details on the three categories "unsupported" attributes.

19) ISSUE: User Performing the Send-Document Operation

The Send-Document and Send-URI commands need the following clarification with regard to the user performing the operation. In the requesting-user-name section of Send-Document add:

> The user performing the Send-Document operation must be the same as for the Create- Job operation that created the job. The printer determines the user performing the operation from the requesting-user-name or the underlying authentication mechanism as described in Section 8.3 of the model document.

The wording in the Send-URI section would imply that the above change applies to Send-URI as well.

Suggested clarification:

Add the suggested clarification to the IPP/1.1 Model and Semantics document. Introduce the terms: "job owner" and "authenticated user". The new text for section 8.3 is:

8.3 URIs for each authentication mechanisms

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

The Printer object uses an authentication mechanism to determine the name of the user performing an operation. This user is called the "authenticated user". The credibility of authentication depends on the mechanism that the Printer uses to obtain the user's name. When the authentication mechanism is 'none', all authenticated users are "anonymous".

During job creation operations, the Printer initializes the value of the "job-originating-user-name" attribute (see section 4.3.6) to be the authenticated user. The authenticated user is this case is called the "job-owner".

If an implementation can be configured to support more than one authentication mechanism, then it MUST implement rules for determining equality of authenticated user names which have been authenticated via different authentication mechanisms. One possible policy is that identical names that are authenticated via different mechanism are different. For example, a user can cancel his job only if he uses the same authentication mechanism for both Cancel-Job and Print-Job. Another policy is that identical names that are authenticated via different mechanism are the same if the authentication mechanism for the later operation is not less strong than the authentication mechanism for the earlier job creation operation. For example, a user can cancel his job only if he uses the same or stronger authentication mechanism for Cancel-Job and Print-Job. With this second policy a job submitted via 'requesting-user-name' authentication could be cancelled via 'digest' authentication. With the first policy, the job could not be cancelled in this way.

A client is able to determine the authentication mechanism used to create a job. It is the ith value of the Printer's "uri-authentication-supported" attribute (see section 4.4.2), where i is the index of the element of the Printer's "uri-printer-supported" attribute (see section 4.4.1) equal to the job's "job-printer-uri" attribute (see section 4.3.3).

which replaces the following text:

8.3 The "requesting-user-name" (name(MAX)) Operation attribute

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

1) via the REQUIRED "requesting-user-name" operation attribute that a client SHOULD supply in all operations. The client MUST obtain the value for this attribute from an environmental or network login name for the user, rather than allowing the user to supply any value. If the client does not supply a value for "requesting-user-name", the printer MUST assume that the client is supplying some anonymous name, such as "anonymous".
2) via an authentication mechanism of the underlying transport which may be configured to give no authentication information.

There are six cases to consider:

a) the authentication mechanism gives no information, and the client doesn't specify "requesting-user-name".
b) the authentication mechanism gives no information, but the client specifies "requesting-user-name".

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

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

The user-name has two forms:

- one that is human readable: it is held in the REQUIRED "job-originating-user-name" Job Description attribute which is set during the job creation operations. It is used for presentation only, such as returning in queries or printing on start sheets
- one for authorization: it is held in an undefined (by IPP) Job object attribute which is set by the job creation operation. It is used to authorize other operations, such as Send-Document, Send-URI, Cancel-Job, to determine the user when the "my-jobs" attribute is specified with Get-Jobs, and to limit what attributes and values to return with Get-Job-Attributes and Get-Jobs.

The human readable user name:

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

The user name used for authorization:

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

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

20) ISSUE: Non-spooling printers accept/reject additional jobs

Some IPP Printer implementations reject a second Print-Job (or Create-Job) while they are processing a Print-Job. Other IPP Printer implementations, such as forwarding servers and non-spooling printers, accept some number of subsequent jobs, but flow control them off until the first job is finished.

Suggested clarification (same as Issues 4 and 5):

Also clarify the IPP/1.1 Model and Semantics document that the following actions are conforming for non-spooling IPP Printer objects: After accepting a create job operation, a non-spooling IPP Printer MAY either:

- Reject any subsequent create job operations while it is busy transferring and/or processing an accepted job request and return the 'server-error-busy (0x0507).

- Accept up to some implementation-defined subsequent create job operations and flow control them to prevent buffer overflow. When the implementation-defined number of jobs is exceeded, the IPP Printer MUST return the 'server-error-busy' status code and reject the create job request as in 1 above.

Client (desktop or server) SHOULD NOT close the channel when flow controlled off, unless the layer that initiated the submission does the close. Clients that are rejected with a 'server-error-busy' status code MAY retry periodically, try another IPP Printer, and/or subscribe for a 'ready-for-job' event when we have notification specified.

IIG: Suggest that a client implementer avoid using synchronous writes, since they automatically close the channel. Use asynchronous writes instead, so that the lower layer doesn't time out the channel.

Suggested text for section 3.1.9 Job Creation Operations:

At job submission time, a Printer object, especially a non-spooling Printer, MAY accept jobs that it does not have enough space for. In such a situation, a Printer object MAY stop reading data from a client for an indefinite period of time. A client MUST be prepared for a write operation to block for an indefinite period of time (see section 5.1 on client conformance).

When a Printer object has too little space for starting a new job, it MAY reject a new create request. In this case, a Printer object MUST return a response (in reply to the rejected request) with a status-code of 'server-error-busy' (See section 14.1.5.8) and it MAY close the connection before receiving all bytes of the operation. A Printer SHOULD indicate that it is temporarily unable to accept jobs by setting the 'spool-space-full' value in its "printer-state-reasons" attribute and removing the value when it can accept another job (see section 4.4.12).

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

Suggested text for clarification of 'spool-area-full' value of "printer-state-reasons":

'spool-area-full': The limit of persistent storage allocated for spooling has been reached. The Printer is temporarily unable to accept more jobs. The Printer will remove this value when it is able to accept more jobs. This value SHOULD be used by a non-spooling Printer that only accepts one or a small number jobs at a time or a spooling Printer that has filled the spool space.

Suggested text for section 5.1 Client Conformance Requirements:

While a client is sending data to a printer, it SHOULD do its best to prevent a channel from being closed by a lower layer when the channel is blocked (i.e. flow-controlled off) for whatever reason, e.g. 'out of paper' or 'job ahead hasn't freed up enough memory'. However, the layer that launched the print submission (e.g. an end user) MAY close the channel in order to cancel the job. When a client closes a channel, a Printer MAY print all or part of the received portion of the document. See the "Encoding and Transport" document [IPP-PRO] for more details.

21) ISSUE: Does 'none' "uri-security-supported" mean Basic/Digest?

Section 4.4.2 "uri-security-supported" 'none' values says:

'none': There are no secure communication channel protocols in use for the given URI.

Should be clarified that the REQUIRED Basic and Digest are intended for the 'none' value. (Hugo Parra)

Suggested clarification:

Instead, clarify that the "uri-security-supported" is only referring to the privacy part of security, not the authentication part, such as HTTP Basic and Digest authentication. Add a note to both the "uri-security-supported" attribute and Section 5.4 on Security Conformance Requirements in the IPP/1.1 Model and Semantics that authentication conformance requirements are specific to a transport, such as HTTP Basic and Digest, and are specified in the Encoding and Transport [ipp-pro] document.

Suggested text for (new) section 4.4.2 "uri-authentication-supported":

'basic': When a client performs an operation whose target is the associated URI, the Printer object challenges the client with HTTP basic authentication. The Printer object assumes that the authenticated user is the name received via the basic authentication mechanism. This authentication mechanism SHOULD be used with a secure channel, that is, the corresponding value of "uri-security-supported" SHOULD NOT be 'none'.

Suggested text for section 4.4.3 "uri-security-supported":

This attribute is orthogonal to the specification of a client authentication mechanism. Specifically, 'none' does not exclude client authentication. See section 4.4.2.

22) ISSUE: Status code on variable-length attributes that are 'too short'

IPP defines a status code 'client-error-request-value-too-long' for a variable-length attribute that exceeds the maximum length allowed by the attribute. However, it is not clear what status code to use in the opposite case, i.e. the supplied attribute value is shorter than the requirement. In the current spec, this problem will arise when a 0-length value is supplied in 'keyword' attributes. In this case, should the request be rejected with status code 'client-error-request-value-too-long' or 'client-error-bad-request' ?

Furthermore, if "ipp-attribute-fidelity" is 'false', should the request be rejected at all? (Jason Chien-Hung Chen)

Suggested clarification in the IIG:

No special status code is needed and no special action is needed by the IPP object. Since this is a keyword, its value needs to be compared with the supported values. Assuming that the printer doesn't have any values in its corresponding "xxx-supported" attribute that are keywords of zero length, the comparison will fail. Then the request will be accepted or rejected depending on the value of "ipp-attributes-fidelity" being 'false' or 'true', respectively. No change to the [ipp-mod]. Indicate this handling of too short keywords in the IIG. All other variable length attribute syntaxes have a minimum greater than 0.

23) ISSUE: There seems to be some misunderstanding about the unsupported-attributes group.

Some implementations return all the attributes that are in the spec that their implementation does not support in the Unsupported Attributes group on a get-attributes operation, independent of the attributes that were actually requested. The unsupported-attributes presumably contains all the attributes the implementation knows about but does not support. I do not believe this is the proper use of the unsupported-attributes group. Do we need a clarification in the document.

Suggested clarification (related to Issues 11 and 18):

Clarify IPP/1.1 Model and Semantics document that only attributes (operation, Job Template, ...) supplied in the request by the client that the IPP object does not support are returned in the Unsupported Attributes group, not all attributes that the implementation doesn't support.

Suggested text for section 3.1.3 Attributes:

The Unsupported Attribute group is defined for all operation responses for returning unsupported attributes that the client supplied in the request.

Suggested text for (new) section 3.1.7 Unsupported Attributes:

See Issue 18.

24) ISSUE What status does Get-Jobs return when no jobs?

Should Get-Jobs return 'successful-ok' when there are no jobs to be returned? The client can see that the Jobs group contains no jobs from the response. Returning an error may confuse the client. Some implementations returned 'client-error-not-found' error code.

Suggested clarification:

Clarify IPP/1.1 Model and Semantics document that the IPP Printer MUST return 'successful-ok' even when there are no jobs to return. The operation is successful and the client will see that there are no returned jobs.

Suggested text for section 3.2.6.2 Get-Jobs Response:

It is not an error for the Printer to return 0 jobs. If the response returns 0 jobs because there are no jobs matching the criteria, and the request would have returned 1 or more jobs with a status code of 'successful-ok' if there had been jobs matching the criteria, then the status code for 0 jobs MUST be 'successful-ok'.

25) ISSUE - MAY an IPP object return more Operation attributes?

Is it ok for an IPP object to return additional operation attributes in a response (as an extension to the standard)? If so, then the client MUST ignore or do something with them. (Hugo Parra)

Suggested clarification:

Clarify IPP/1.1 Model and Semantics document that the client MUST ignore or do something with additional operation attributes returned than are in the IPP/1.1 Model and Semantics document.

Suggested text for section 5.1 Client Conformance:

A response MAY contain attribute groups, attributes, and values that the client does not expect. Therefore, a client implementation MUST gracefully handle such responses and not refuse to inter-operate with a conforming Printer that is returning registered or private extensions, including attribute groups, attributes, attribute syntaxes, and attribute values that conform to Section 6. Clients may choose to ignore any parameters, attributes, or values that they do not understand.26) ISSUE: MAY an IPP object return additional groups?

It is ok for an IPP object to return additional groups of attributes in a response (as an extension to the standard)? For example, returning the "job-state" and "job-state-reasons" in a Hold-Job, Release-Job, and/or Cancel-Job operation. What about newly registered groups of attributes. If so, then the client MUST ignore or do something with them. (Hugo Parra)

Suggested clarification:

Clarify IPP/1.1 Model and Semantics document that the client MUST ignore or do something with additional attribute groups returned than are in the IPP/1.1 Model and Semantics document. Also clarify that these additional groups MAY occur in any position.

Suggested text for section 5.2.2 Operations:

Conforming IPP objects MAY return operation responses that contain attributes groups, attributes names, attribute syntaxes, and attribute values that are extensions to this standard. The additional attribute groups MAY occur in any order.

27) ISSUE: Return first or all unsupported attributes in Unsupported Group?

Section 16.3 and 16.4 of the Model and Semantics document was moved to the Implementer's Guide when creating the November 1998 draft from the June 1998 draft. The following note was contained in section 16.4 that was moved:

Note: whether the request is accepted or rejected is determined by the value of the "ipp-attribute-fidelity" attribute in a subsequent step, so that all Job Template attribute supplied are examined and all unsupported attributes and/or values are copied to the Unsupported Attributes response group.

Suggested clarification (same clarification as Issue 18):

Clarify in the IPP/1.1 Model and Semantics document that all operation attributes and all Job Template attributes MUST be returned in the Unsupported Attributes group, unless there is a specific error status for the unsupported operation attribute, such as: server-error-version-not-supported, server-error-operation-not-supported, client-error-charset-not-supported, client-error-compression-not-supported, client-error-document-format-not-supported, and client-error-uri-scheme-not-supported.

Suggested text:

See Issue 18.

28) ISSUE: What if compression is supplied but not supported?

The "compression" operation attribute is an OPTIONAL attribute for a Printer object to support in a create operation. However, if a client supplies the "compression" attribute, but the IPP object doesn't support the attribute at all, the Printer might attempt to print data it doesn't understand, because it is compressed. In order to prevent this error, the "compression" operation attribute should have been REQUIRED.

Possible Alternatives (related to Issues 3 and 6):

1. Clarify that an IPP object MUST reject a request that supplies a "compression" operation attribute, if the IPP object does not support the "compression" attribute at all. As with any such error, the IPP object copies the "compression" attribute to the Unsupported Attribute Group setting the value to the out-of-band 'unsupported' value and returns the "client-error-attributes-or-values-not-supported" status code. The IPP object MAY reject the request, even if the client supplies the 'none' value, since the IPP Printer does not have a corresponding "compression-supported" attribute.

2. Add a 'client-error-compression-not-supported' error status code. Require IPP Printer's to support this error code if they do not support the "compression" operation attribute.

3. Change IPP/1.1 Model and Semantics conformance requirement for the "compression" and "compression-supported" attributes from OPTIONAL to REQUIRED.

Suggested change:

Suggested IPP/1.1 Change (related to Issues 3 and 6): REQUIRE that IPP/1.1 implementations MUST support "compression" and "compression-supported" (with at least the 'none' value), even though it is OPTIONAL for IPP/1.0.

Add the 'client-error-document-format-error' for error detected at request time with a supported document format, such as PostScript Level 3 not supported by a PostScript level 2 printer. Describe the priority between 'client-error-document-format-not-supported', 'client-error-compression-not-supported', 'client-error-document-format-error', and 'client-error-compression-error' status codes.

Also add "compression-supported" to the Appendix E on directory schema as a RECOMMENDED attribute.

Add to IIG for IPP/1.0: IPP/1.0 SHOULD at least check for the "compression" attribute being present and reject the create request, if they don't support "compression". Not checking is a bug, since the data will be unintelligible.

Suggested text for "compression" operation attribute:

"compression" (type3 keyword)

The client OPTIONALLY supplies this attribute. The Printer object MUST support this attribute and the "compression-supported" attribute (see section 4.4.30). The client supplied "compression" operation attribute identifies the compression algorithm used on the document data. The following cases exist:

a) If the client omits this attribute, the Printer object MUST assume that the data is not compressed (i.e. the Printer follows the rules below as if the client supplied the "compression" attribute with a value of 'none').
b) If the client supplies this attribute, but the value is not supported by the Printer object, i.e., the value is not one of the values of the Printer object's "compression-supported" attribute, the Printer object MUST reject the request, and return the 'client-error-compression-not-supported' status code. See section 3.2.1.2 for returning unsupported attributes and values.
c) If the client supplies the attribute and the Printer object supports the attribute value, the Printer object uses the corresponding decompression algorithm on the document data.
d) If the decompression algorithm fails before the Printer returns an operation response, the Printer object MUST reject the request and return the 'client-error-compression-error' status code.
e) If the decompression algorithm fails after the Printer returns an operation response, the Printer object MUST abort the job and add the 'compression-error' value to the job's "job-state-reasons".
f) If the decompression algorithm succeeds, the document data MUST then have the format specified by the job's "document-format" attribute, if supplied (see "document-format" operation attribute definition below).

Suggested text for a new section 13.1.4.16 client-error-compression-not-supported

13.1.4.16 client-error-compression-not-supported (0x040F)

The IPP object is refusing to service the request because the document data, as specified in the "compression" operation attribute, is compressed in a way that is not supported by the Printer object. This error is returned independent of the client-supplied "ipp-attribute-fidelity". The Printer object MUST return this status code, even if there are other Job Template attributes that are not supported as well, since this error is a bigger problem than with Job Template attributes. See section 0.

29) ISSUE: Should "queued-job-count" be REQUIRED?

The "queued-job-count" Printer Description attribute is an OPTIONAL attribute for a Printer object to support. Since some clients may want a quick way to determine the load on an IPP Printer, querying the "Printer's "queued-job-count" should always be possible, but an implementation might not support it.

Suggested change:

Change IPP/1.1 Model and Semantics so that the "queued-job-count" changes from RECOMMENDED to REQUIRED.

30) ISSUE: Should "job-state-reasons" and "printer-state-reasons" be REQUIRED in IPP/1.1?

Considering that we tend to put more and more information into the currently OPTIONAL 'job-state-reason' and 'printer-state-reason' attributes, should we make them a MUST for the IPP/1.1 version? (Discussion in 990324 phone conference).

Suggested change:

Change IPP/1.1 document "job-state-reasons" and "printer-state-reasons" from OPTIONAL to REQUIRED for IPP/1.1. All references to "If the "job-state-reasons" attribute is supported, need to be removed.

Suggested changed to the "job-state-reasons" description in Print-Job response:

"job-state-reasons":

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

Suggested text for addition to section 4.3.8 "job-state-reasons":

These values MAY be used with any job state or states for which the reason makes sense. Some of these value defintions indicate conformance requirements; the rest are OPTIONAL.

The following state reasons have the indicated conformance requirements added to reflect conformance requirements already stated elsewhere in the document. All of the other values are OPTIONAL:

'none': There are no reasons for the job's current state. This state reason is semantically equivalent to "job-state-reasons" without any value and MUST be used when there is no other value, since the 1setOf attribute syntax requires at least one value.

'document-access-error': After accepting a Print-URI or Send-URI request, the Printer could not access one or more documents passed by reference. This reason is intended to cover any file access problem, including file does not exist and access denied because of an access control problem. Whether the Printer aborts the job and moves the job to the 'aborted' job state or prints all documents that are accessible and moves the job to the 'completed' job state and adds the 'completed-with-errors' value in the job's "job-state-reasons" attribute depends on implementation and/or site policy. This value SHOULD be supported if the Print-URI or Send-URI operations are supported. Issue 30 and Issue 35

'job-hold-until-specified': The value of the job's "job-hold-until" attribute was specified with a time period that is still in the future. The job MUST NOT be a candidate for processing until this reason is removed and there are no other reasons to hold the job. This value SHOULD be supported if the "job-hold-until" Job Template attribute is supported.

'job-canceled-by-user': The job was canceled by the owner of the job using the Cancel-Job request, i.e., by a user whose authenticated identity is the same as the value of the originating user that created the Job object, or by some other authorized end-user, such as a member of the job owner's security group. This value SHOULD be supported.

'job-canceled-by-operator': The job was canceled by the operator using the Cancel-Job request, i.e., by a user who has been authenticated as having operator privileges (whether local or remote). If the security policy is to allow anyone to cancel anyone's job, then this value may be used when the job is canceled by other than the owner of the job. For such a security policy, in effect, everyone is an operator as far as canceling jobs with IPP is concerned. This value SHOULD be supported if the implementation permits canceling by other than the owner of the job.

'job-canceled-at-device': The job was canceled by an unidentified local user, i.e., a user at a console at the device. This value SHOULD be supported if the implementation supports canceling jobs at the console.

'aborted-by-system': The job (1) is in the process of being aborted, (2) has been aborted by the system and placed in the 'aborted' state, or (3) has been aborted by the system and placed in the 'pending-held' state, so that a user or operator can manually try the job again. This value SHOULD be supported.

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

If the implementation requires some measurable time to cancel the job in the 'processing' or 'processing-stopped' job states, the IPP object MUST use this value to indicate that the Printer object is still performing some actions on the job while the job remains in the 'processing' or 'processing-stopped' state. After all the job's job description attributes have stopped incrementing, the Printer object moves the job from the 'processing' state to the 'canceled' or 'aborted' job states.

'job-completed-successfully': The job completed successfully. This value SHOULD be supported.

'job-completed-with-warnings': The job completed with warnings. This value SHOULD be supported if the implementation detects warnings.

'job-completed-with-errors': The job completed with errors (and possibly warnings too). This value SHOULD be supported if the implementation detects errors.

'job-restartable' - This job is retained (see section 4.3..7.1 and is currently able to be restarted using the Restart-Job operation (see section 3.3.7)). If 'job-restartable' is a value of the job's 'job-state-reasons' attribute, then the IPP object MUST accept a Restart-Job operation for that job. This value SHOULD be supported if the Restart-Job operation is supported.

Suggested text for addition to section 4.4.12 "printer-state-reasons":

Some of the these value definitions indicate conformance requirements; the rest are OPTIONAL.

'none': There are not reasons. This state reason is semantically equivalent to "printer-state-reasons" without any value and MUST be used, since the 1setOf attribute syntax requires at least one value.

'moving-to-paused': Someone has paused the Printer object using the Pause-Printer operation (see section 3.2.7 or other means, but the device(s) are taking an appreciable time to stop. Later, when all output has stopped, the "printer-state" becomes 'stopped', and the 'paused' value replaces the 'moving-to-paused' value in the "printer-state-reasons" attribute. This value MUST be supported, if the Pause-Printer operation is supported and the implementation takes significant time to pause a device in certain circumstances.

'paused': Someone has paused the Printer object using the Pause-Printer operation (see section 3.2.7 or other means and the Printer object's "printer-state" is 'stopped'. In this state, a Printer MUST NOT produce printed output, but it MUST perform other operations requested by a client. If a Printer had been printing a job when the Printer was paused, the Printer MUST resume printing that job when the Printer is no longer paused and leave no evidence in the printed output of such a pause. This value MUST be supported, if the Pause-Printer operation is supported.

'spool-area-full': The limit of persistent storage allocated for spooling has been reached. The Printer is temporarily unable to accept more jobs. The Printer will remove this value when it is able to accept more jobs. This value SHOULD be used by a non-spooling Printer that only accepts one or a small number jobs at a time or a spooling Printer that has filled the spool space. Issue 20 Issue 30 and Issue 31

31) ISSUE: How indicate a ripped job that is waiting for the marker?

Three alternatives being pursued: job stays in 'processing', job moves to 'pending', job moves to 'pending-held' job states. Any of the alternatives MAY use a new 'queued-for-marker' job state reason to indicate that the job has been ripped but is waiting for the marker in a high end system. The 'pending-held' state is used by systems where the Operator explicitly does a Release-Job to schedule the next job to be marked, while the 'pending' or 'processing' state is used by systems that choose the next job to mark automatically. The 'processing' state is typically used by systems that tend not to have much time between ripping and marking.

Suggested clarifications:

1. Clarify that a Printer may have more than one job in the processing state at the same time.

2. Clarify that a job can remain in the 'processing' state even when the Printer is 'stopped', if that job is being ripped; only the job that is being marked MUST be moved to the 'processing-stopped' state.

3. Simplify the descriptions of the three Printer states, while preserving compatibility with IPP/1.0, including fan-out, and incorporate the above clarifications as well.

Suggested addition:

All three job states may be used to represent jobs that have been interpreted and are waiting to be marked, depending on implementation.

Suggested text for additions to section 4.3.8 job-state-reasons:

'job-queued-for-marker': Job is in any of the 'pending-held', 'pending', or 'processing' states, but more specifically, the Printer has completed enough processing of the document to be able to start marking and the job is waiting for the marker. Systems that require human intervention to release jobs using the Release-Job operation, put the job into the 'pending-held' job state. Systems that automatically select a job to use the marker put the job into the 'pending' job state or keep the job in the 'processing' job state while waiting for the marker, depending on implementation. All implementations put the job into (or back into) the 'processing' state when marking does begin.

Simplified "printer-state" descriptions that preserves IPP/1.0 compatibility, including fan-out, while allowing multiple rips with a marker:

'3' 'idle': Indicates that new jobs can start processing without waiting.

'4' 'processing': Indicates that jobs are processing; new jobs will wait before processing.

'5' 'stopped': Indicates that no jobs can be processed and intervention is required.

Values of "printer-state-reasons", such as 'spool-area-full' and 'stopped-partly', MAY be used to provide further information.

Put the above deleted definitions for simple, multi-rip, and fan-out configurations into the IIG. This deleted text appeared in the May Internet-Drafts. Also discuss the difficulty of the simple IPP/1.1 Printer object abstraction to represent a set of devices that have widely differing capabilities, i.e., a color and a black and white printer.

Clarification of 'spool-area-full' "printer-state-reasons" value:

'spool-area-full': The limit of persistent storage allocated for spooling has been reached. The Printer is temporarily unable to accept more jobs. The Printer will remove this value when it is able to accept more jobs. This value MAY be used by a non-spooling Printer that only accepts one or a small number jobs at a time or a spooling Printer that has filled the spool space.

32) ISSUE: Is Digest REQUIRED for an IPP Client and an IPP Printer to support?

The Transport and Encoding document contains the following incorrect sentence:

> The IPP Model document defines an IPP implementation with "authentication" as one that implements the standard way for transporting IPP messages within HTTP 1.1.

since the IPP Model document doesn't mention HTTP 1.1, since that is a transport issue.

The Transport and Encoding document refers to RFC 2068 (HTTP/1.1) and RFC 2069 (Digest), but does not require that RFC 2069 be supported. Furthermore, RFC 2068 does not require that RFC 2069 be supported either.

Suggested change:

Change the Transport and Encoding document to require that clients and Printers MUST support HTTP 1.1.

Suggested change:

Suggested change to Encoding and Transport document for IPP/1.1 conformance:

> An IPP Printer SHOULD contain software that allows an administrator to configure the client authentication part of HTTP Digest (but not encryption of the body)
>
> IPP clients MUST implement the above in order to be able to interoperate with conforming Printers.

Clients and Printers MAY also support additional Client Authentication, such as:

1. HTTP Basic (not certificates) over a TLS secured channel (implementing TLS authentication is NOT REQUIRED).

2. HTTP Basic (not certificates) over an SSL3 secured channel.

A Printer implementation MAY allow an administrator to configure the Printer so that all, some, or none of the users are authenticated.

A Printer MUST NOT allow security to be compromised when accepting a '1.0' request. - from the Area Director e-mail.

Suggested text for Section 5.1 Client Conformance:

A client MUSTsupport Client Authentication as defined in the IPP/1.1 Encoding and Transport document [IPP-PRO]. A client SHOULD support Operation Privacy and Server Authentication as defined in the IPP/1.1 Encoding and Transport document [IPP-PRO]. See also section 8 of this document.

Suggested text for a new sub-section to Section 5.2 IPP Object Conformance:

5.2.7 Security

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

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

Security MUST NOT be compromised when a client supplies a lower "version-number" parameter in a request. For example, if an IPP/1.1 conforming Printer object accepts version '1.0' requests and is configured to enforce Digest Authentication, it MUST do the same for a version '1.0' request.

33) ISSUE: Include the IPP/1.0 conformance requirements in the IPP/1.1 document?

Suggested change:

No. The IPP/1.1 Model and Semantics document and the IPP/1.1 Encoding and Transport document will only cover IPP/1.1. They will NOT obsolete the experimental RFC that describes IPP/1.0. They will NOT describe IPP/1.0 at all.

The IPP/1.1 document will say that for interoperability with IPP/1.0 clients, that an IPP Printer SHOULD accept IPP/1.0 requests ("version-number" parameter = '1.0') and, if they accept the request, MUST respond with IPP/1.0 responses ("version-number" parameter = '1.0') . Furthermore, an IPP/1.1 conforming Printer or an IPP/1.0 conforming Printer MAY respond with any IPP/1.1 feature in such an IPP/1.0 response. If the IPP/1.1 Printer does not support version '1.0', i.e., does not support the conformance requirements of IPP/1.0, it SHOULD still accept the version-number '1.0' request. Likewise, the IPP/1.1 Printer SHOULD accept requests with the version-number '1.2' and '1.n' requests (even though it doesn't support all of the conformance requirements of that version).

Suggested text for the Appendices

The IPP/1.1 documents will contain an appendix that summarizes each difference from IPP/1.0 by section number and a brief description (see February 1999 I-Ds). The appendix will contain two separate lists: one is clarifications and OPTIONAL additions to IPP/1.1 and the other is changes in conformance requirements of existing IPP/1.0 features or new REQUIRED IPP/1.1 features.

Here are the items for the Appendix for IPP-PRO:

1. IPP/1.1 clients and Printers MUST support the IPP scheme; IPP/1.0 clients and Printers MUST support the http scheme.

2. IPP/1.1 clients MUST support the user authentication part of Digest; IPP/1.0 clients SHOULD support SSL3 which uses the https scheme and non-SSL3 access. Issue 32

3. IPP/1.1 Printers SHOULD contain support for the user authentication part of Digest; IPP/1.0 Printers SHOULD support SSL3 which uses the https scheme and non-SSL3 access. Issue 32

Here are the items for the second list in the Appendix for IPP-MOD:

The following changes in semantics and/or conformance have been incorporated into this document:

1. Section 3.1.8, 5.2.4, and 13.5.1.4 - Clients and IPP objects MUST support version 1.1 conformance requirements and SHOULD support version 1.0 conformance requirements. Also clarified that IPP Printers MUST accept '1.1' requests and SHOULD accept '1.x' requests. Issue 33 and Issue 36
2. Section 3.2.1.1 and section 4.4.32 - changed the "compression" operation and the "compression-supported" Printer Description attribute from OPTIONAL to REQUIRED. Issue 28
3. Sections 3.2.1.2 and 4.3.8 - changed "job-state-reasons" from RECOMMENDED to REQUIRED, so that "job-state-reasons" MUST be returned in create operation responses. Issue 30

4. Sections 3.2.4, 3,3,1, 4.4.16, and 16 - changed Create-Job/Send-Document so that they MAY be implemented while only supporting one document jobs. Added the "multiple-document-jobs-supported" boolean Printer Description attribute to indicate whether Create-Job/Send-Document support multiple document jobs or not. Added to the Directory schema. Issue 34
5. Section 4.1.9 - deleted 'text/plain; charset=iso-10646-ucs-2', since binary is not legal with the 'text' type.
6. Section 4.2.4 - indicated that the "multiple-document-handling" Job Template attribute MUST be supported with at least one value if the Printer supports multiple documents per job Issue 34
7. Section 4.3.7.1 - indicated that the 'job-restartable' job state reason SHOULD be supported if the Restart-Job operation is supported. Issue 30
8. Section 4.3.8 - changed "job-state-reasons" from RECOMMENDED to REQUIRED. Issue 30
9. Section 4.3.8 - clarified the conformance of the values of the "job-state-reasons" attribute by copying conformance requirements from other sections of the document so that it is clear from reading the definition of "job-state-reasons" which values MUST or SHOULD be supported. The 'none', 'unsupported-compression', and 'unsupported-document-format' values MUST be supported. The "job-hold-until-specified' SHOULD be specified if the "job-hold-until" Job Template is supported. The following values SHOULD be supported: 'job-canceled-by-user', 'aborted-by-system', and 'job-completed-successfully'. The 'job-canceled-by-operator' SHOULD be supported if the implementation permits canceling by other than the job owner. The 'job-canceled-at-device' SHOULD be supported if the device supports canceling jobs at the console. The 'job-completed-with-warnings' SHOULD be supported, if the implementation detects warnings. The 'job-completed-with-errors' SHOULD be supported if the implementation detects errors. The 'job-restartable' SHOULD be supported if the Restart-Job operation is supported. Issue 30
10. Section 4.3.14 - changed the "time-at-creation", "time-at-processing", and "time-at-completed" Event Time Job Description attributes from OPTIONAL to REQUIRED. Issue 17
11. Section 4.3.13.4 - added the REQUIRED "job-printer-up-time (integer(1:MAX))" Job Description attribute as an alias for "printer-up-time" to reduce number of operations to get job times. Issue 17
12. Section 4.4.2 - added the REQUIRED "uri-authentication-supported (1setOf type2 keyword)" Printer Description attribute to describe the Client Authentication used by each Printer URI. Issue 2
13. Section 4.4.12 - changed "printer-state-reasons" Printer Description attribute from OPTIONAL to REQUIRED. Issue 30
14. Section 4.4.12 - changed 'paused' value of "printer-state-reasons" to MUST if Pause-Printer operation is supported. Issue 30
15. Section 4.4.14 - added the REQUIRED "ipp-versions-supported (1setOf keyword)" Printer Description attribute, since IPP/1.1 Printers do not have to support version '1.0' conformance requirements. Issue 36
16. Section 4.4.16 - added the "multiple-document-jobs-supported (boolean)" Printer Description attribute so that a client can tell whether a Printer that supports Create-Job/Send-Document supports multiple document jobs or not. This attribute is REQUIRED if the Create-Job operation is supported. Issue 34
17. Section 4.4.24 - changed the "queued-job-count" Printer Description attribute from RECOMMENDED to REQUIRED. Issue 29
18. Section 4.4.32 - changed "compression-supported (1setOf type3 keyword)" Printer Description attribute from OPTIONAL to REQUIRED. Issue 28
19. Section 5.1 - changed the client security requirements from RECOMMENDED non-standards track SSL3 to MUST support Client Authentication as defined in the IPP/1.1 Encoding and Transport document [IPP-PRO]. A client SHOULD support Operation Privacy and Server Authentication as defined in the IPP/1.1 Encoding and Transport document [IPP-PRO]. Issue 32

20. Section 5.2.7 - changed the IPP object security requirements from OPTIONAL non-standards track SSL3 to SHOULD contain support for Client Authentication as defined in the IPP/1.1 Encoding and Transport document [IPP-PRO]. A Printer implementation MAY allow an administrator to configure the Printer so that all, some, or none of the users are authenticated. An IPP Printer implementation SHOULD contain support for Operation Privacy and Server Authentication as defined in the IPP/1.1 Encoding and Transport document [IPP-PRO]. A Printer implementation MAY allow an administrator to configure the degree of support for Operation Privacy and Server Authentication. Security MUST NOT be compromised when the client supplies a lower version-number in a request. Issue 32

For the IIG:

1. Discuss the advantage for client implementations to support both IPP/1.1 and IPP/1.0, so that they can interoperate with either Printer implementations.

2. Discuss the advantage for Printer implementations to support both IPP/1.1 and IPP/1.0, so that they can interoperate with either client implementations.

34) ISSUE: Ok to REQUIRE "multiple-document-handling if Create-Job is supported?

The IPP/1.0 Implementer's Guide contains the following issue:

> 2.16 Support of multiple document jobs
>
> IPP/1.0 is silent on which of the four effects an implementation would perform if it supports Create-Job, but does not support "multiple-document-handling".
>
> A fix to IPP/1.0 would be to require implementing all four values of "multiple-document-handling" if Create-Job is supported at all. Or at least 'single-document-new-sheet' and 'separate-documents-uncollated-copies'. In any case, an implementation that supports Create-Job SHOULD also support "multiple-document-handling". Support for all four values is RECOMMENDED, but at least the 'single-document-new-sheet' and 'separate-documents-uncollated-copies' values, along with the "multiple-document-handling-default" indicating the default behavior and "multiple-document-handling-supported" values. If an implementation spools the data, it should also support the 'separate-documents-collated-copies' value as well.

There is a need to allow Create-Job and Send-Document to be supported while making it OPTIONAL to support multiple documents per job. A client that wants to monitor a job while it is sending data can do so with Create-Job and Send-Document. A Printer that wants to support "long documents", namely, when the document data is indefinitely long (so long it can't be spooled) but does not want to support multiple documents.

Suggested solution:

Instead of requiring "multiple-document-handling" if Create-Job and Send-Document are supported as proposed in the original solution for Issue 34, lets:

1. Clarify that a conforming implementation NEED NOT support multiple documents when it supports the Create-Job and Send-Document operations. (There currently is no conformance sentence that requires support of multiple document jobs when Create-Job and Send-Document are supported, though that was certainly our intent which this clarification would countermand).

2. If the Printer does support the Create-Job and Send-Document operations, then it MUST support the (new) "multiple-document-jobs-supported (boolean)" Printer Description attribute. A 'true' value indicates that multiple documents are supported in a job.

3. Add "multiple-document-jobs-supported (boolean)" to the Directory Schema in Appendix E as OPTIONAL.

4. If the Printer does support multiple documents in a job, then it MUST support the "multiple-document-handling" Job Template attribute with at least one value and the associated "multiple-document-handling-default" and "multiple-document-handling-supported" Job Template Printer attributes.

5. Add a new status code: 'server-error-multiple-document-jobs-not-supported'

6. In the table in section 14.2 indicate that 'server-error-multiple-document-jobs-not-supported' can be used only with the Send-Document and Send-URI operations.

Suggested text for section 3.2.4 Create-Job:

If the Printer object supports this operation, then it MUST support the "multiple-document-jobs-supported" Printer Description attribute (see section 4.4.16) and indicate whether or not it supports multiple-document jobs.

If the Printer object supports this operation and supports multiple documents in a job, then it MUST support the "multiple-document-handling" Job Template job attribute with at least one value (see section 4.2.4) and the associated "multiple-document-handling-default" and "multiple-document-handling-supported" Job Template Printer attributes (see section 4.2).

Suggested text for section 3.3.1 Send-Document operation:

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

Suggested text for section 4.2.4 multiple-document-handling

After the first sentence which says:

> This attribute is relevant only if a job consists of two or more documents.

add:

> This attribute MUST be supported if the Printer supports multiple documents per job (see sections 3.2.4 and 3.3.1).

Suggested text for new section 4.4.28 multiple-document-jobs-supported

4.4.28 multiple-document-jobs-supported (boolean)

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

Suggested text for new section 14.1.5.10:

14.1.5.10 server-error-multiple-document-jobs-not-supported (0x0509)

The IPP object does not support multiple documents per job and a client attempted to supply document data with a second Send-Document or Send-URI operation.

35) ISSUE: What error code to return on Print-URI or Send-URI if document not accessible?

Section 3.2.2, "Print-URI Operation", it looks like it's an implementation decision whether to pull the data from the document-uri at job submission time or at job processing time. Say I decide to pull the data at job submission time. Say I get some kind of error doing so, like no-route-to-host, or HTTP 404. Shouldn't I return some kind of error status? Currently, it looks like I have to return successful-ok as long as the document-uri uses a scheme I support, regardless of whether or not I can actually get the document data. (Carl Kugler)

Suggested additions:

1. Add both a new 'client-error-document-access-error' status code and a 'document-access-error' value for "job-state-reasons", just like we have done for compression and document format errors for Issue 3, 6, and 28.

2. Add OPTIONAL "job-document-access-errors (1setOf text)" Job Description and "document-access-error (text(MAX))" Print-URI/Send-URI operation attribute. For protocol errors, such as HTTP or FTP errors, standard values would be the error code in parentheses, followed by the URL For example:

```
      (XXX) http://ftp.pwg.org/pub/pwg/ipp/new_MOD/ipp-model-v11-990510.pdf
      (XXX) ftp://ftp.pwg.org/pub/pwg/ipp/new_MOD/ipp-model-v11-990510.pdf
```

For example, the value: (404) http://... would indicate that the HTTP server could not find the resource. (Note: most Internet protocols use decimal error codes (unlike IPP), so the ASCII keyword representation is in decimal.)

3. Instead of adding "debug-info", add the "detailed-status-message (text(MAX))" operation attribute and "job-detailed-status-messages (1setOf text(MAX))" Job Description attribute which contains detailed information that is not localized by the client or Printer.

Add to section 3.1.6 Operation Status Codes and Messages:

3.1.6 Operation Response Status Code and Status Messages:

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

3.1.6.1 "status-code" (type2 enum)

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

The status code is intended for use by automata. A client implementation of IPP SHOULD convert status code values into any localized message that has semantic meaning to the end user.

The "status-code" value is a numeric value that has semantic meaning. The "status-code" syntax is similar to a "type2 enum" (see section 4.1 on "Attributes Syntaxes") except that values can range only from 0x0000 to 0x7FFF. Section 13 describes the status codes, assigns the numeric values, and suggests a corresponding status message for each status code for use by the client when the user's natural language is English.

If the Printer performs an operation with no errors and it encounters no problems, it MUST return the status code 'successful-ok' in the response. See section 13.

If the client supplies unsupported values for the following parameters or Operation attributes, the Printer object MUST reject the operation, NEED NOT return the unsupported attribute value in the Unsupported Attributes group, and MUST return the indicated status code:

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

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

3.1.6.2 "status-message" (text(255))

The OPTIONAL "status-message" operation attribute provides a short textual description of the status of the operation. The "status-message" attribute's syntax is "text(255)", so the maximum length is 255 octets (see section 4.1.1). The status message is intended for the human end user. If a response does include a "status-message" attribute, an IPP client NEED NOT examine or display the messages, however it SHOULD do so in some implementation specific manner. The "status-message" is especially useful for a later version of a Printer object to return as supplemental information for the human user to accompany a status code that an earlier version of a client might not understand.

If the Printer object supports the "status-message" operation attribute, the Printer object MUST be able to generate this message in any of the natural languages identified by the Printer object's "generated-natural-language-supported" attribute (see the "attributes-natural-language" operation attribute specified in section 3.1.4.1. Section 13 suggests the text for the status message returned by the Printer for use with the English natural language.

As described in section 3.1.4.1 for any returned 'text' attribute, if there is a choice for generating this message, the Printer object uses the natural language indicated by the value of the "attributes-natural-language" in the client request if supported, otherwise the Printer object uses the value in the Printer object's own "natural-language-configured" attribute.

If the Printer object supports the "status-message" operation attribute, it SHOULD use the REQUIRED 'utf-8' charset to return a status message for the following error status codes (see section 13): 'client-error-bad-request', 'client-error-charset-not-supported', 'server-error-internal-error', 'server-error-operation-not-supported', and 'server-error-version-not-supported'. In this case, it MUST set the value of the "attributes-charset" operation attribute to 'utf-8' in the error response.

3.1.6.3 "detailed-status-message" (text(MAX))

The OPTIONAL "detailed-status-message" operation attribute provides additional more detailed technical and implementation-specific information about the operation. The "detailed-status-message" attribute's syntax is "text(MAX)", so the maximum length is 1023 octets (see section 4.1.1). If the Printer objects supports the "detailed-status-message" operation attribute, neither the Printer nor the client localizes the message, since it is intended for use by the system administrator or other experienced technical persons. Clients MUST NOT attempt to parse the value of this attribute. See the "document-access-error" operation attribute (section 3.1.6.4) for additional errors that a program can process.

3.1.6.4 document-access-error(text(MAX))

This OPTIONAL operation attribute provides additional information about any document access errors encountered by the Printer before it returned a response to the Print-URI (section 3.2.2) or Send-URI (section 3.3.1) operation. For errors in the protocol identified by the URI scheme in the "document-uri" operation attribute, such as 'http:' or 'ftp:', the error code is returned in parentheses, followed by the URI. For example:

```
     (404) http://ftp.pwg.org/pub/pwg/ipp/new_MOD/ipp-model-v11-990510.pdf

```

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

Suggested text for section 3.2.2 Print-URI Operation:

Replace the sentences:

See The Implementer's Guide [IPP-IIG] for suggested additional checks. The Printer NEED NOT follow the reference and validate the contents of the reference.

with:

The IPP Printer MAY validate the accessibility of the document as part of the operation or subsequently. If the Printer determines an accessibility problem before returning an operation response, it rejects the request and returns the 'client-error-document-access-error' status code. The Printer MAY also return a specific document access error code using the "document-access-error" operation attribute (see section 3.1.6.4).

If the Printer determines this document accessibility problem after accepting the request and returning an operation response with one of the successful status codes, the Printer adds the 'document-access-error' value to the job's "job-state-reasons" attribute and MAY populate the job's "job-document-access-errors" Job Description attribute (see section 4.3.11). See The Implementer's Guide [IPP-IIG] for suggested additional checks.

Suggested text for section 4.3.8 job-state-reasons:

'document-access-error': After accepting a Print-URI or Send-URI request, the Printer could not access one or more documents passed by reference. This reason is intended to cover any file access problem, including file does not exist and access denied because of an access control problem. The Printer MAY also indicate the document access error using the "job-document-access-errors" Job Description attribute (see section 4.3.11). Whether the Printer aborts the job and moves the job to the 'aborted' job state or prints all documents that are accessible and moves the job to the 'completed' job state and adds the 'completed-with-errors' value in the job's "job-state-reasons" attribute depends on implementation and/or site policy. This value SHOULD be supported if the Print-URI or Send-URI operations are supported.

Add two new sections after section 4.3.9 job-state-message (text(MAX)):

4.3.10 job-detailed-status-messages (1setOf text(MAX))

This attribute specifies additional detailed and technical information about the job. Neither the Printer nor the client localizes the message(s), since they are intended for use by the system administrator or other experienced technical persons. Clients MUST NOT attempt to parse the value of this attribute. See "job-document-access-errors" (section 4.3.11) for additional errors that a program can process.

4.3.11 job-document-access-errors (1setOf text(MAX))

This attribute provides additional information about each document access error for this job encountered by the Printer after it returned a response to the Print-URI or Send-URI operation and subsequently attempted to access document(s) supplied in the Print-URI or Send-URI operation. For errors in the protocol that is identified by the URI scheme in the "document-uri" operation attribute, such as 'http:' or 'ftp:', the error code is returned in parentheses, followed by the URI. For example:

```
    (404) http://ftp.pwg.org/pub/pwg/ipp/new_MOD/ipp-model-v11-990510.pdf
```

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

Suggested text for section 13.1.4.19 Client Error Status Codes:

13.1.4.19 client-error-document-access-error (0x0412)

The IPP object is refusing to service the Print-URI or Send-URI request because Printer encountered an access error while attempting to validate the accessibility or access the document data specified in the "document-uri" operation attribute. The Printer MAY also return a specific document access error code using the "document-access-error" operation attribute (see section 3.1.6.4). This error is returned independent of the client-supplied "ipp-attribute-fidelity". The Printer object MUST return this status code, even if there are Job Template attributes that are not supported as well, since this error is a bigger problem than with Job Template attributes.

36) ISSUE: Don't require 1.0 support and add REQUIRED "ipp-version-supported" attribute

Suggested additions:

1. RECOMMEND, rather than REQUIRE, conforming IPP/1.1 clients and the IPP/1.1 Printers to support IPP/1.0, i.e., meet the conformance requirements for IPP/1.0 as specified in RFC 2566 and RFC 2565.

2. Therefore, add an "ipp-versions-supported" Printer Description attribute which indicates which conformance requirements the Printer implementation meets.

3. Indicate that a Printer implementation that supports version '1.0' MAY support any extension defined in the IPP/1.1 document, since the conformance requirement in RFC 2566 permit a conforming IPP/1.0 implementation to support extensions, as long as such extensions don't violate any conformance requirements in RFC 2566 and RFC 2565.

4. If the major version number matches, but the minor version number does not, the Printer SHOULD accept and attempt to process the request, or MAY reject the request and return the 'server-error-version-not-supported' status code. In all cases, the Printer MUST return the nearest version number that it supports. For example, suppose that an IPP/1.2 Printer supports versions '1.1' and '1.2'. The following responses are conforming:

Client supplies	Printer Accept Request?	Printer returns
1.0	yes (SHOULD)	1.1
	no (SHOULD NOT)	1.1
1.1	yes (MUST)	1.1
1.2	yes (MUST)	1.2
1.3	yes (SHOULD)	1.2
	no (SHOULD NOT)	1.2

Put this table in the IIG.

5. Also add the "ipp-version-numbers-supported" attribute as RECOMMENDED in the directory schema list in the Appendix.

6. Fix the rule for using minor version numbers so that we can still use '1.1' for this version, and not be forced to change the version number to '2.0'.

Suggested text for the new attribute:

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

This REQUIRED attribute identifies the IPP protocol version(s) that this Printer supports, including major and minor versions, i.e., the version numbers for which this Printer implementation meets the conformance requirements. For version number validation, the Printer matches the (two-octet binary) "version-number" parameter supplied by the client in each request (see sections 3.1.1 and 3.1.8) with the (US-ASCII) keyword values of this attribute.

The following standard keyword values are defined:

'1.0': Meets the conformance requirement of IPP version 1.0 as specified in RFC 2566 [RFC2566] and RFC 2565 [RFC2565] including any extensions registered according to Section 6 and any extension defined in this version or any future version of the IPP "Model and Semantics" document or the IPP "Encoding and Transport" document following the rules, if any, when the "version-number" parameter is '1.0'.

'1.1': Meets the conformance requirement of IPP version 1.1 as specified in this document and [IPP-PRO] including any extensions registered according to Section 6 and any extension defined in any future versions of the IPP "Model and Semantics" document or the IPP Encoding and Transport document following the rules, if any, when the "version-number" parameter is '1.1'.

Suggested modification to section 3.1.8 Versions:

3.1.8 Versions

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

If the IPP object does not support that major version number supplied by the client, i.e., the major version field of the "version-number" parameter does not match any of the values of the Printer's "ipp-versions-supported" (see section 4.4.14), the object MUST respond with a status code of 'server-error-version-not-supported' along with the closest version number that is supported (see section 13.1.5.4). If the major version number is supported, but the minor version number is not, the IPP object SHOULD accept and attempt to perform the request (or reject the request if the operation is not supported), else it rejects the request and returns the 'server-error-version-not-supported' status code. In all cases, the IPP object MUST return the "version-number" that it supports that is closest to the version number supplied by the client in the request.

There is no version negotiation per se. However, if after receiving a 'server-error-version-not-supported' status code from an IPP object, a client SHOULD try again with a different version number. A client MAY also determine the versions supported either from a directory that conforms to Appendix E (see section 16) or by querying the Printer object's "ipp-versions-supported" attribute (see section 4.4.14) to determine which versions are supported. Issue 36

An IPP object implementation MUST support version '1.1', i.e., meet the conformance requirements for IPP/1.1 as specified in this document and [IPP-PRO]. An IPP object implementation SHOULD support version '1.0', i.e., meet the conformance requirements for IPP/1.0 [RFC2566 and RFC2565]. A client MAY also determine the versions supported either from a directory that conforms to Appendix E (see section 16) or by querying the Printer object's "ipp-versions-supported" attribute (see section 4.4.14) to determine which versions are supported.

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

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

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

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

- grouping all extensions not included in a previous version into a new version
- adding new attribute values
- adding new object attributes
- adding OPTIONAL (for an IPP object to support) operation attributes (i.e., those attributes that an IPP object can ignore without confusing clients)
- adding OPTIONAL (for an IPP object to support) operation attribute groups (i.e., those attributes that an IPP object can ignore without confusing clients)
- adding new attribute syntaxes
- adding OPTIONAL operations
- changing Job Description attributes or Printer Description attributes from OPTIONAL to REQUIRED or vice versa.
- adding OPTIONAL attribute syntaxes to an existing attribute. Issue 33

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

Implementations that support a certain version NEED NOT support ALL previous versions. As each new version is defined (through the release of a new IPP specification document), that version will specify which previous versions MUST and which versions SHOULD be supported in compliant implementations. Issue 36

Suggested addition and change to section 5.2.4 [Conformance of] Versions:

Clients MUST meet the conformance requirements for clients specified in this document and [IPP-PRO] and SHOULD also support version 1.0, i.e., SHOULD meet the conformance requirements for clients as specified in [RFC2566] and [RFC2565].

IPP Printer and Job objects MUST meet the conformance requirements for IPP objects specified in this document and [IPP-PRO]. For interoperability with IPP/1.0 clients, IPP/1.1 objects SHOULD also meet the conformance requirements for IPP objects as specified in [RFC2566] and [RFC2565].

Clients MUST send requests containing a "version-number" parameter with a '1.1' value and SHOULD try supplying alternate version numbers if they receive a 'server-error-version-not-supported' error return in a response.

IPP objects MUST accept requests containing a "version-number" parameter with a '1.1' value (or reject the request if the operation is not supported). IPP objects SHOULD accept any request with the major version '1' (or reject the request if the operation is not supported). See section 3.1.8.

Suggested changes to section 13.1.5.4 server-error-version-not-supported (0x0503)

13.1.5.4 server-error-version-not-supported (0x0503)

The IPP object does not support, or refuses to support, the IPP protocol version that was supplied as the value of the "version-number" operation parameter in the request. The IPP object is indicating that it is unable or unwilling to complete the request using the same major and minor version number as supplied in the request other than with this error message. The error response SHOULD contain a "status-message" attribute (see section 3.1.6.2) describing why that version is not supported and what other versions are supported by that IPP object. See sections 3.1.6 and 3.1.8. Issue 11

The error response MUST identify in the "version-number" operation parameter the closest version number that the IPP object does support. For example, if a client supplies version '1.0' and an IPP/1.1 object supports version '1.0', then it MUST respond with version '1.0' in all responses to such a request. If the IPP/1.1 object does not support version '1.0', then it SHOULD accept the request and respond with version '1.1' or MAY reject the request and respond with this error code and version '1.1'. If a client supplies a version '1.2' the IPP/1.1 object SHOULD accept the request and return version '1.1' or MAY reject the request and respond with this error code and version '1.1'. See sections 3.1.8 and 4.4.14. Issue 36