Subj: IPP Bake Off 2 Issues
From: Peter Zehler, Tom Hastings, and Bob Herriot
File: Issues-raised-at-Bake-Off2.doc
Version: 1.4~~2~~
Date: 4/12~~3/22~~/1999

This version incorporates the discussion on the mailing list and three telecons held 3/24/99, 3/31/99, and 4/7/99 on resolving the IPP/1.1 issues raised at Bake Off 2.

NOTE: Since the Model and Semantics document and the Encoding and Transport documents are going to cover both IPP/1.0 and IPP/1.1, as agreed at the March IETF meeting, any issue that does not mention IPP/1.0 or IPP/1.1 explicitly means that the resolution applies to BOTH IPP/1.0 and IPP/1.1 in the same way. Only if IPP/1.0 and/or IPP/1.1 is mentioned explicitly is there to be a difference explicitly stated in the resulting IPP/1.1 standards track document that covers both IPP/1.0 (non-standards track) and IPP/1.1 (standards track).

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. Please feel free to add additional alternatives or disagree with our suggested clarifications or additions via e-mail so that the group may have the widest possible set of alternatives to choose from. All the additional material is indicated with revision marks from the issues list that Peter Zehler published March 19, 1999.

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 or alternatives, if the issue is still OPEN. 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.

Note: These are issues that are to be resolved in the IPP/1.1 documents before forwarding them to the IESG for publication as proposed standards. The IPP/1.0 documents have already been forwarded to the RFC Editor after approval by the IESG for publication as Informational RFCs, so these issues and their resolution will not affect the IPP/1.0 documents.

Status codes:

AGREED - agreement on the telecon 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 -", the Issue number, and brief description of the issue.

OPEN - still being discussed at future telecons and on the DL.

OPEN issues remaining: 2, 17, 30, 31, 32, and 33.

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: OPEN - alternatives being discussed: new operation, two URLs, its not a problem. Also relationship to SLP template.

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.

4) ISSUE: Client closes slow channel

Suggested clarification (same as Issues 5 and 20): AGREED that client MUST NOT close channel, unless user indicates or policy. RAISE on DL explicitly to verify AGREEMENT.

5) ISSUE: Client closes stopped device

Suggested clarification (same as Issues 4 and 20): AGREED that client MUST NOT close channel, unless user indicates or policy. RAISE on DL explicitly to verify AGREEMENT.

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.

Suggested clarification for the IIG: OPEN - ~~Need a general syntax that separates the vendor name from the model name. Propose to the DL.~~Recommend that the value contain the vendor name and the model in that order.

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.

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 MUST 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. Propose to DL explicitly, since not many implementations did return the attribute. In IPP/1.1 document say that IPP/1.0 MAY, but NEED NOT.

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. Bring out on the DL explicitly for confirmation.

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?

Possible alternatives: OPEN - carry on discussion on DL to add three new date/time job attributes or add dateTime attribute syntax to the existing job attributes:

ISSUE: Make the time job attributes REQUIRED for IPP/1.1? 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.

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

Suggested clarification (same clarification as Issue 27): AGREED - all unsupported 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.

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

Suggested clarification (same as Issues 4 and 5): AGREED that IPP object MAY accept implementation defined number of subsequent create operations, including NONE. RAISE on DL explicitly to verify AGREEMENT.

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.

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 - client MUST process or ignore additional attribute groups returned.

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

Suggested clarification (same clarification as Issue 18): AGREED - all unsupported 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~~Possible Alternatives~~ (related to Issues 3 and 6): ~~OPEN~~ CLOSED - propose to the DL explicitly that "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.

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.

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

Suggested change: ~~OPEN~~ CLOSED - propose to the DL explicitly that "queued-job-count" be REQUIRED for IPP/1.1, even though it is a SHOULD for IPP/1.0.

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

Suggested change: OPEN - 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? Raise on DL explicitly to see if there is agreement. (Discussion in 990324 phone conference).

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

Suggested addition: OPEN - 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 'interpreted-waiting-to-print' 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' state is used by systems that choose the next job to mark automatically. The 'processing' state is used by systems that tend not to have much time between ripping and marking.

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

Suggested change to Encoding and Transport document: OPEN - Ask the Area Director whether Digest MUST be supported by an IPP Printer or not.

33) ISSUE: Ok to include the IPP/1.0 conformance requirements in the IPP/1.1 document?

Suggested change: Most conformance requirements are the same for IPP/1.0 and IPP/1.1. For those make no special indication in the document. For those for which the conformance is REQUIRED for IPP/1.1, but OPTIONAL for IPP/1.0, state: "IPP/1.1 xxx MUST ...; OPTIONAL in IPP/1.0", where xxx is either clients or Printers.

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) OPEN - 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.

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", if supported. If the "job-state-reasons" attribute is supported, 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. The suggested text is:

’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 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 MUST NOT close the channel when flowed controlled off. 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.

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.

1. Client MUST NOT close the channel when flow controlled off. 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.

5) ISSUE: Client 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~~ix~~ (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 MUST not close the channel when flow controlled off. 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.

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 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 'unsupported-compression' job-state-reason.

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.

Recommend that the "printer-make-and-model" value contain the vendor name and the model in that order.

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.

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. ~~ISSUE: Still need to talk to proposer of this issue, since the "document-format-default" should be set to 'application/octet-stream' if the default is to auto-sense.~~

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 ~~s~~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.

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 MUST 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. In IPP/1.1 document say that IPP/1.0 MAY, but NEED NOT.~~that the "document-format" attribute and the supplied value NEED NOT be returned in the Unsupported Attributes group. If there are also some unsupported Job Template attributes supplied in the create request, the IPP Printer MAY, but NEED NOT, return them in the Unsupported Attributes Group when returning the 'client-error-document-format-not-supported', since the document-format error is a higher precedence error and the document is not going to be able to be processed at all on the Printer.~~

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.

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.

But "job-state-reasons" is OPTIONAL. Do we mandate it or recommend it if supporting Create-Job?

Suggested clarification:

Clarify the IPP/1.1 Model and Semantics document that an IPP Printer MAY put the job into the ’pending’, ~~or~~ ’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 support the "job-state-reasons" and use the ’job-incoming’ 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.

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 printing before it has received all data, the Printer removes the ’job-data-insufficient’ reason, but the ’job-incoming’ remains. If a Printer starts printing 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 ~~either~~ the 'pending-held', ~~or~~ 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. ~~RECOMMEND~~ 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 "job-state-reasons" attribute and the new 'queued-in-device' value when returning the job in the 'completed' state. IPP/1.0 implementations of forwarding servers NEED NOT support "job-state-reasons" with the 'queued-in-device' value.

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.

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) OPEN - 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 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:

Clarify that the "time-at-xxx" attributes can be negative if an IPP Printer is re-booted while jobs remain.

1. 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)".
2. Instead of adding new 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)"
3. Same as 1, but make the job attributes be REQUIRED for IPP/1.1.
4. Same as 2, but make support of the dateTime REQUIRED for IPP/1.1.
5. Return "printer-up-time" (in seconds) as an operation attribute in Get-Jobs and Get-Job-Attributes response.

6. 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.

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

If ipp-attributes-fidelity=true, MUST all 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, such as 'client-error-document-not-supported'.

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.

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 MUST NOT close the channel when flow controlled off. 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.

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.

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 specification.

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.

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.

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 specification.

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 specification.

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, such as 'client-error-document-not-supported'.

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 "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.

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.

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

Suggested change:

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? Raise on DL explicitly to see if there is agreement. (Discussion in 990324 phone conference).

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

Suggested addition:

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 'interpreted-waiting-to-print' 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’ state is used by systems that choose the next job to mark automatically. The ’processing’ state is used by systems that tend not to have much time between ripping and marking.

32) OPEN - 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.

Suggested change:

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

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:

IPP/1.1 clients and Printers MUST support Digest [RFC 2069]; OPTIONAL for IPP/1.0.

33) OPEN - ISSUE: Ok to include the IPP/1.0 conformance requirements in the IPP/1.1 document?

Suggested change:

Most conformance requirements are the same for IPP/1.0 and IPP/1.1. For those make no special indication in the document. For those for which the conformance is REQUIRED for IPP/1.1, but OPTIONAL for IPP/1.0, state: "IPP/1.1 xxx MUST ...; OPTIONAL in IPP/1.0", where xxx is either clients or Printers.

Here are the 12 items for which the conformance requirements differ between IPP/1.0 and IPP/1.1:

1. IPP/1.1 clients MUST be able to issue both IPP/1.1 and IPP/1.0 requests and accept both IPP/1.1 and IPP/1.0 responses.

2. IPP/1.1 Printers MUST accept and support both IPP/1.1 and IPP/1.0 requests and responses.

3. IPP/1.1 clients and Printers MUST support the IPP scheme; OPTIONAL for IPP/1.0.

4. IPP/1.1 clients SHOULD support TLS and non-TLS access; OPTIONAL for IPP/1.0. IPP/1.0 clients SHOULD support SSL3 and non-SSL3 access; OPTIONAL for IPP/1.1.

5. Printers MAY support SSL3 access, access without SSL3 or both. In addition, IPP/1.1 Printers SHOULD support TLS access, MAY support access without TLS, or MAY support both means of access; OPTIONAL for IPP/1.0.

6. Possible resolution of OPEN ISSUE 32: IPP/1.1 clients and Printers MUST support Digest; OPTIONAL for IPP/1.0.

7. IPP/1.1 Printers MUST return "document-format=xxx" in the Unsupported Attributes Group; OPTIONAL for IPP/1.0. See Issue 11.

8. IPP/1.1 Printers implemented as a forwarding server that can’t get status from the device or print system (such as LPD) that it forwards jobs to, MAY put the job into the ’completed’ state after forwarding the job. However, such implementations MUST support the "job-state-reasons" attribute with the ’queue-in-device’ value when it puts the job into the ’completed’ state, as an indication that the job is not necessarily completed; OPTIONAL for IPP/1.0 forwarding servers. See Issue 14.

9. Possible resolution of OPEN ISSUE 17: IPP/1.1 Printers MUST support dateTime for the new or existing Job attributes; OPTIONAL for IPP/1.0.

10. IPP/1.1 Printers MUST support "compression" and "compression-supported" attributes with at least the ’none’ value. IPP/1.0 Printers MUST at least check for the "compression" attribute being present and reject the create request, if they don’t support "compression" and "compression-supported". Not checking is a bug, since the data will be unintelligible.

    Note: On Larry Masinter’s advice, I changed the SHOULD to a MUST for IPP/1.0, ok?

11. IPP/1.1 Printers MUST support "queued-job-count"; RECOMMEND for IPP/1.0.

12. Possible resolution of OPEN ISSUE 30: IPP/1.1 Printers MUST support "job-state-reasons" and "printer-state-reasons"; OPTIONAL for IPP/1.0.