Peter,

Here are my comments shown as revisions.

There is one issue highlighted like this for the IPP telecon and mailing list

Tom

Q0) The textWithLanguage and nameWithLanguage Issue was resolved. (The description below applies to name as well as text.)

The agreed upon interpretation comes from the definition of 'textWithLanguage' itself. The 'textWithLanguage' definition contains:

"The 'textWithLanguage' attribute syntax is a compound attribute syntax consisting of two parts: a 'textWithoutLanguage' part plus an additional 'naturalLanguage' "

This verbiage together with the length definition of 'textWithoutLanguage of 1023 octets implies the following conclusion. The total length of an attribute value of type 'textWithLanguage' would be 1086. (This ignores the lengths types and attribute name portions of the encoding) The resulting two error cases, with respect to length, are the length of 'language' exceeds 63 octets and the length of 'text' exceeds 1023 octets

RESOLUTION 0)
No change will be made to the IPP/1.1 Model and Semantics. The following will be included in an update to the IPP/1.1 Implementers Guide

TH> I went with what the Minutes said and added the following clarification to the Model document:

```
This text clarification will be done to the Model document, and also explained in

the Implementer's
Guide.
```

"4.1.4 Maximum length for xxxWithLanguage and xxxWithoutLanguage
The 'textWithLanguage' and 'nameWithLanguage' are compound syntaxes that have two component. The first component is the 'language' component that can contain up to 63 octets. The second component is the 'text' or 'name' component. The maximum length of these are 1023 octets and 255 octets respectively. The definition of attributes with either syntax may further restrict the length. (e.g. printer-name (name(127)))
The length of the 'language' component has no effect on the allowable length of 'text' in 'textWithLanguage' or the length of 'name' in 'nameWithLanguage'

TH> This is very good and clear for the IIG (with typo fixed).

Here is what ~~the RFC~~[ipp-mod] has to say about the "Resume-Printer Operation",

"This operation allows a client to resume the Printer object scheduling jobs on all its devices. The Printer object MUST remove the 'paused' and 'moving-to-paused' values from the Printer object's "printer-state-reasons" attribute, if present. If there are no other reasons to keep a device paused (such as media-jam), the IPP Printer transitions itself to the 'processing' or 'idle' states, depending on whether there are jobs to be processed or not, respectively, and the device(s) resume processing jobs."
What this means is, according to the RFC description, the "Resume-printer" operation must change the 'printer-state-reason' to some thing other than 'Paused' and 'moving-to-paused' and it may or may not result in the Printer transitioning itself to the 'processing' or 'idle' states, depending on the actual physical device status (and, of~~f~~ course, depending on whether any jobs are queued-up for the printer or not !)

RESOLUTION 1a)
Edit the IPP/1.1 Model and Semantics document containing the above content to

“3.2.8 Resume-Printer Operation

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

TH> I goofed in editing the Model by not making this simple change indicated in the minutes to replace "transitions itself" with "is free to transition itself". I'll add that as a typo to fix with the RFC editor.

Q1) Is the responsibility of the "Resume-Printer" operation limited to changing the Printer objects 'printer-state-reasons' or does it have any bearing on the 'Printer-state' also ?

RESOLUTION 1)
"resume-printer" does NOT have a responsibility to change the printer state. Keep in mind that there may be other reasons for a printer's state besides someone issuing a 'pause-printer'. These other conditions may prevent the printer from transitioning to 'idle' or 'processing'. (Corrective action in resolution 1a)

Q2) If, in case, the IPP printer is unable to change its state due to some problem with the actual printer device (say, it is shut down or there is a media-jam as indicated in ~~the RFC~~[ipp-mod]), what should be the result of the "Resume-printer" operation?
Should it still change the 'printer-state-reasons' and return success or should it fail ?

RESOLUTION 2)

The 'resume-printer' operation must clear the 'paused' or 'moving-to-paused' 'printer-state-message'. The operation must return a 'successful-ok' status code (No corrective action)

TH> Gee. I think this would be a good clarification to add to the IIG. Why not add it as a question and answer in the IIG under the Resume-Printer section?

Q3) If it succeeds after changing the 'printer-state-reasons', what should be the value of 'Printer-state' and who should take care of the 'Printer-state' later on ?

RESOLUTION 3)

The 'printer-state 'will change to one of three states
'idle' - no additional jobs and no error conditions present
'processing' - job available and no error conditions present
current state (i.e. no change) an error condition is present (e.g. media jam)

In the third case the 'printer-state-reason' will be cleared by automata when it detects the error condition no longer exists. The 'printer-state' will move to 'idle' or 'processing' when conditions permit. (i.e. no more error conditions) (Corrective action in resolution 1a)

TH> This seems to be good info for the IIG in a section on Resume-Printer operation.

Q4) Assume a client has performed a Print-URI operation with a reference to a large document .The request is processed checking for the uri-scheme and the accessibility of the document, and the client is sent a response with a valid job-id. While the IPP server is downloading the document mid way ,the client does a get-job-attributes. We would return the job-state as pending-held , but what the job-state-reasons say. There is a job-state-reason 'job-incoming' , but is limited to create-job.

RESOLUTION 4) ***(Note: checking accessibility is optional)***
Use 'job-incoming'. I assume your implementation requires that the entire job stream be transferred to your printer prior to processing. If you could begin before transfer is complete then the 'job-state' would be 'processing' and the 'job-state-reasons' could also include 'job-interpreting', 'job-transforming' and/or 'job-printing'. Assuming your implementation can not process until the entire document data has been retrieved I agree with your 'job-state'.
Update IPP/1.1 Model and Semantics: 4.3.8 job-state-reasons (1setOf type2 keyword)
Change
"'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."
To
"'job-incoming': A job creation 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/retrieving document
data.

TH> Another good fix for the Model as a typo with the RFC editor. However, it applies to more than just the job creation operations (Create-Job, Print-Job, and Print-URI) and its only the latter part that applies to other operations. I suggest the following for [ipp-mod] instead, ok:

"'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 after accepting a Print-Job, Print-URI, Send-Document and/or Send-URI operation**."**

Q5) assume a user sends a Create-Job request and is successful. Without performing a send URI/Doc operation he issues a last-doc without any document data(so the IPP server would have a job that is created without any data) . what should be the response of the IPP server to this last-doc

. Will a client-error-not-possible (0x0404) status-code suffice.

RESOLUTION 5)

The protocol does not prohibit a job without a document. It is not a protocol error. The 'last-doc' request should be accepted. There is no requirement that an IPP Job contains a document. The response to the operation with the ‘last-doc’ set and no document must be ‘successful-ok’ assuming no other conditions exist. When the job is processed the resulting ‘job-state’ would be ‘completed’ and the ‘job-state-reasons’ would include ‘job-completed-successfully’ if no other conditions exist that would cause an error or warning (e.g. requesting a finishing option).

Add the following to the IPP/1.1 Implementers Guide:

“4.5 Empty Jobs

The IPP object model does not prohibit a job that contains no documents. Such a job may be created in a number of ways including a ‘create-job’ followed by an ‘add-document’ that contains no data and has the ‘last-document’ flag set.

An empty job is processed just as any other job. The operation that “closes” an empty job is not rejected because the job is empty. If no other conditions exist, other than the job is empty, the response to the operation will indicate success. After the job is scheduled and processed, the job state SHALL be ‘completed’

There will be some variation in the value(s) of the ‘job-state-reasons’ attribute. It is required that if no conditions, other than the job being empty, exist the ‘job-state-reasons’ SHALL include the ‘completed-successfully’. If other conditions existed, the ‘completed-with-warnings’ or ‘completed-with-errors’ values may be used.”

TH> Looks good for the IIG. However, I also clarified the [ipp-mod] by adding the following sentence to the Group 2 section of 3.3.1.1 Send-Document Request:

> It is not an error for a client to submit a job with no actual document data, i.e., only a single Create-Job and Send-Document request with a "last-document" operation attribute set to 'true' with no document data.

Q6) assume a printer is paused(as a result of which the printer status is set to 'stopped') and has jobs queued under it . A user issues a Purge-Jobs request and is successful . Now should the printer status be changed to 'idle' or does 'stopped' override 'idle'.

RESOLUTION 6)
The 'purge-jobs' does not affect the state of the printer. The state of the printer should remain as it was.

The IPP/1.1 Implementers Guide will be updated with a state transition diagram

TH> However, [ipp-mod] contains the following statement under Purge-Jobs:

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

I believe that we should clarify when the Printer doesn't move to 'idle', since it is correct for all conditions, except if the Printer is in the 'stopped' state with the 'paused' reason. If the Printer was in the 'stopped' state because of a 'media-jam', then Purge-Jobs does transition the Printer (eventually) to 'idle' (perhaps after the media jam is fixed by human intervention).

ISSUE 01: How about the following for [ipp-mod], while we work on a state transition table for the IIG:

Replace in Purge-Jobs operation:

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

with:

> The Printer object MUST accept this operation in any state and either (1) transition the Printer object to the 'idle' state or (2) keep it in the 'stopped' state or move it to the 'stopped' state, if the Printer had been paused using the Pause-Printer operation ("printer-state-reasons" = 'paused' or 'moving-to-paused', respectively).

Q7) Restart on a Create-Job request:
Assume I give a Create-Job request along with a set of 5 documents . All the documents get printed and the job state is moved to completed . I issue a Restart-Job request on the job. Now the issue is that, if I try to add new documents to the restarted job ,will the Ipp Server permit me to do so or return "client-error-not-possible " and again print those 5 jobs.

RESOLUTION 7)
A job can not move to the 'completed' state until all the documents have been processed. The 'last-document' flag indicates when the last document for a job is being sent from the client. This is the semantic equivalent of closing a job. No documents may be added once a job is closed. Section 3.3.7 of the IPP/~~v~~1.1 model states "The job is moved to the 'pending' job state and restarts the beginning on the same IPP Printer object with the same attribute values." 'number-of-documents' is a job attribute.

The IPP/1.1 Implementers Guide will be updated with a state transition diagram

TH> I also suggest adding the above paragraph to the IIG.

Q8) Restart on a Print-URI request that was completed :
If I issue a Restart-Job request on a job that is in state "completed", does the IPP Server need to down load the file again or proceed with a file that it had already downloaded.

RESOLUTION 8)

The IPP job does not contain any data, only a reference. The job data resides wherever the URL indicates. Although the specification does not prohibit caching of print data caching may introduce inconsistent behavior with other IPP implementations.

The following will be added to the IPP/1.1 Implementers Guide

“3.3.2 Restart-job

The ‘restart-job’ operation allows the reprocessing of a completed job. Some jobs store the document data on the printer. Jobs created using the Print-Job~~‘print-job’~~ operation are an example. It is required that the printer retains the job data after the job has moved to a ‘completed state’ in order for the ~~‘restart-job’~~Restart-Job operation to succeed. Some jobs contain only a reference to the job data. A job created using the ~~‘print-uri’~~Print-URI is an example of such a job. When the ~~‘restart-job’~~Restart-Job operation is issued the job is reprocessed. It is expected that the job data will be retrieved again to print the job. ~~The standard does not disallow caching the job data on the printer. Implementers should be aware that the contents of the print data might have changed from the last time the data was cached.~~ It is possible that a job fails while attempting to access the print data. When such a job is the target of a Restart-Job~~‘restart-job’~~ the Printer SHALL attempt to retrieve the job data again.”

TH> I thought we agreed that the Printer MUST re-fetch the data using the URI. If so, the above two sentences MUST be deleted as I have indicated.

I added the following to Restart-Job in the [ipp-mod], ok:

> If any of the documents in the job were passed by reference (Print-URI or Send-URI), the Printer MUST re-fetch the data, since the semantics of Restart-Job are to repeat all Job processing.

Q9) Restart on a Print-URI request that was aborted while downloading the URI :
If I issue a Restart-Job request on a job that is in state "aborted" (due to download failing), does the IPP Server need to down load the file again or return "client-error-not-possible "?

RESOLUTION 9)

It SHALL attempt to retrieve the job data again. (The 'print-uri' does necessarily mean download. A client may upload the file to the printer via FTP. The URL used in 'print-uri' could have a method of 'file://'). On a Restart-Job operation, the Printer MUST re-fetch the data using the URI no matter where it points.

TH> However, on a Restart-Job, the Printer MUST re-fetch the data using the 'file://' URI, in case a client had re-pushed the file data with FTP, correct? So the simplest fix is to use the word 're-fetch', not 'down-load', as I have done in the [ipp-mod] fix, ok? So add the additional sentence to the IIG that I added above.

See “RESOLUTION 8” for text that will be added to the IPP/1.1 Implementer’s Guide to address this issue.