PWG WORKING DRAFT
ipp-event-notification-proposal.doc   .pdf

T. Hastings
Xerox Corporation
S. Isaacson
Novell, Inc.
H. Lewis
IBM Corporation
May 12~~9~~, 1998

Event notifications for the IPP print protocol [and JMP]

Version 0.05~~4~~

~~There are several issues indicated in the document that we should cover at the upcoming meeting, as well as review the proposal. See color highlighting.~~

The appendix has the full specification for the ’collection’ attribute syntax, as agreed on our 5/6/98 telecon.

[Items in square brackets relate to the PWG Standard Job Monitoring MIB [jmp-~~-~~mib] trapping and will be removed when this document is made into an IPP Internet-Draft.]

Status of this Memo

This document is a PWG Working Draft. It is intended to become a first Internet-Draft when there is rough consensus that it is ready and then to proceed on the IETF standards track to be used with IPP/1.0. It is being developed under the charter for IPP/1.0 and meets the requirements in [req].

Abstract

In IPP/1.0, the user can determine what is happening to submitted jobs by using the Get--Attributes and Get-Jobs operations to poll for results. This document describes an OPTIONAL extension to the IPP/1.0 Model document for subscribing for event notifications using IPP, but which are delivered over some other protocol, either by the IPP Printer object or by any notification service that the IPP Printer object implementation may employ. See [req] for the notification requirements.

Two methods are provided for subscription for notification events: (1) as part of the job submission and (2) as a separate Subscribe-For-Event-Notifications operation. Both methods allow the requester to specify (1) about which event(s) to be notified, (2) which notification-recipient(s) are to receive the notification, (3) what content type is to be sent in the notification, and (4) which notification transport method is to be used. Both methods allow the requester to subscribe for job event groups, such as ’job-completion’, and/or printer events, such as ’printer-errors’.

The event notification subscription mechanism uses a new attribute syntax called a ’collection’. A ’collection’ value is a set of attributes. See the Appendix of this document for the complete specification of the ’collection’ attribute syntax.

1 Introduction

In IPP/1.0, the user can determine what is happening to submitted jobs by using the Get-Attributes and Get-Jobs operations to poll for results. This document describes an OPTIONAL extension to the IPP/1.0 Model document for subscribing for event notifications using IPP, but which are delivered over some other protocol, either by the IPP Printer object or by any notification service that the IPP Printer object implementation may employ. See the IPP Notification Requirements document [req] for further details. See also "General Event Notification Architecture Base [cohen] for terminology and framework.

This document contains the definition and use of event notifications (see terminology section) for two main purposes. First, when used to achieve printing over a wide area network, or the Internet, the end-user experience is similar to today's FAX paradigm, so we want to provide notification that the job has completed successfully (or not). This notification may traverse the Internet as an e-mail message or end up on someone's pager. Second, and more widely, when used as a standard LAN print submission protocol (i.e., LPR replacement), the end-user will have the desire and opportunity for a much more dynamic interaction with the printer and the print job. Here, notification should consist of a local area network messaging scheme that addresses unsolicited events related to the printer, the job's position in the server or printer queue, start of processing, printing progress and job completion, including forms of cancellation. This paper proposes MANDATORY IPP attributes to be used for both purposes, and OPTIONAL attributes and values that are appropriate only for one or the other.

[The notification events and content are also intended to apply to the PWG Job Monitoring MIB ~~(JMP)~~[jmp-mib]. See sections 5.1.2.2 and 6.]

1.1 Summary of the proposal for IPP Event Notification

This paper proposes the following:

1. One OPTIONAL "job-notify" Operation attribute for use with the Print-Job, Print-URI, and Create-Job operation. The "job-notify" Operation attribute has an attribute syntax of '1setOf collection' (see Appendix) so that the client can request different events for different notification recipients for the same job. Each collection value SHALL contain the "notify-recipients" and MAY contain any of the following remaining member attributes with the indicated syntax and support by the IPP object if it supports the "job-notify" Operation attribute at all:

Member attribute name	syntax	in request	support
"notify-event-groups"	1setOf type2 keyword	MAY	mandatory
"notify-recipients"	1setOf uri	SHALL	mandatory
"notify-content-type"	mimeMediaType	MAY	mandatory
"notify-charset"	charset	MAY	mandatory
"notify-natural-language"	naturalLanguage	MAY	optional
"notify-additional-attributes"	1setOf keyword	MAY	optional

2. Two new OPTIONAL Subscribe-For-Event-Notifications and Un~~S~~subscribe-For-Event-Notifications operations on the Printer object. These operations are intended for operator/administrators and servers for long term subscription for Printer object events that are independent of job submission. The servers may be involved with (1) job submission to IPP Printer objects and/or (2) collecting accounting data using the event notification mechanism.

   An IPP Printer SHALL support both of these operations, if it supports either one. If an IPP Printer supports these operations, it SHALL also support the "job-notify" attribute in the create operations.

3. One "job-notify" Job object Description attribute which is populated with the collection value(s) supplied by the "job-notify" Operation attribute in a create operation.

~~ISSUE 01: Would a better name be "job-notification-subscription" and the member attributes be named "notification-xxx"?~~

4. Six Job object Description attributes for monitoring job progress at the sheet completed and collated document copy level to align with the PWG Job Monitoring MIB.

~~4.~~5. One new "printer-notify" Printer object Description attribute which is populated with the collection value supplied by the "printer-notify" Operation attribute in the Subscribe-For-Event-Notifications operation. Both attribute use the same collection as the "job-notify" Operation attribute. The "printer-notify" Printer Description attribute also has an additional "notify-subscription-id" member attribute which is an integer id for the subscription for use with the Un~~S~~subscribe-For-Event-Notification operation.

6. Six "~~job-~~xxx-supported" Printer object Description attributes that correspond to the~~se~~ six member attributes in the collection values of the "job-notify" and "printer-notify" Operation attributes. ~~See the IPP Model for the semantics of xxx-supported Printer attributes.~~ ~~ISSUE 02: Would a better name be "printer-notification-subscription"?~~

2 Terminology

It is necessary to define a set of terms in order to be able to clearly express the requirements for notification services in an IPP System. These terms are from the requirements document [req]. Cohen [cohen] has similar terminology, with some differences. ~~ISSUE 03: Which terminology should we use?~~

~~ISSUE 04: Some of these terms are not used in the specification. Should we delete them?~~

2.1 Job Submitting End User

A human end user who submits a print job to an IPP Printer. This person may or may not be within the same security domain as the Printer. This person may or may not be geographically near the printer.

2.2 Job Submitting Application

An application (for example a batch application), acting on behalf of an end user, which submits a print job to an IPP Printer. The application may or may not be within the same security domain as the Printer. This application may or may not be geographically near the printer.

2.3 Security Domain

For the purposes of this discussion, the set of network components which can communicate without going through a proxy or firewall. A security domain may be geographically very large, for example - anyplace within IBM.COM.

2.4 IPP Client

The software component on the client system which implements the IPP protocol which can be either a Job Submitting End User or a Job Submitting Application.

2.5 Job Recipient

A human who is the ultimate consumer of the print job. In many cases this will be the same person as the Job Submitting End User, but this need not always be the case. For example, if I use IPP to print a document on a printer in a business partner’s office, I am the Job Submitting End User, while the person I intend the document for in my business partner’s office is the Job Recipient. Since one of the goals of IPP is to be able to print near the ultimate recipient of the printed output, we would normally expect the Job Recipient to be in the same security domain as, and geographically near the Printer. However, this may not always be the case. For example, I submit a print job across the Internet to a Kinko’s print shop. I am both the Submitting end User and the Job Recipient, but I am neither near nor in the same security domain as the Printer.

2.6 Job Recipient Proxy

A person acting on behalf of the Job Recipient. In particular, the Job Recipient Proxy physically picks up the printed document from the Printer, if the Job Recipient cannot perform that function. The Proxy is **by definition** geographically near and in the same security domain as the printer. For example, I submit a print job from home to be printed on a printer at work. I'd like my secretary to pick up the print job and put it on my desk. In this case, I am acting as both Job Submitting End User and Job Recipient. My secretary is acting as a Job Recipient Proxy.~~ An issue that needs to be considered in the notification architecture is the impact of a third party receiving many unwanted notifications.~~

2.7 Notification Recipient Agent

A program which receives events on behalf of the notification recipient. The agent may take some action on behalf of the recipient, forward the notification to the recipient via some alternative means (for example, page the recipient), or queue the notification for later retrieval by the recipient.

2.8 Notification Recipient

Any of: Job Submitting End User, Job Submitting Application, Job Recipient, or Job Recipient Proxy or Notification Recipient Agent.

2.9 Notification Events

There are Job events and Printer events. These events are characterized as being report, warning, or error, depending on the severity of the event. A report event is purely informational, such as 'job-completed'. A warning event is not serious and processing continues, but a more serious event might come next. An error event is serious and either the job is aborted, or the printer stops until human intervention occurs.

A Job event is some interesting change in the Job object, such as: (1) a change in the Job object's "job-state" attribute, (2) the stacking of another sheet, reflected in the incrementing of the job's "job-media-sheets-completed" attribute or (3) some of the changes in the value of the job's "job-state-reasons" attribute. Not all changes in a job's "job-state" attribute are separate events. For example, the event 'job-received' is the transition from the 'unknown' state to either the 'pending' or 'pending-held' state. Not all changes in a job's other attributes are events.

A Printer event is some interesting change in the Printer object, such as: (1) a change in the Printer object's "printer-state" attribute from 'processing' to 'stopped', (2) a change in the Printer object's "printer-is-accepting-jobs" attribute, or 3~~(2~~) ~~a change ~~certain changes in the Printer object's "printer-state-reasons" attribute. A Printer event corresponds one-to-one with the addition or removal of a row in the Printer MIB alert table, for those implementations that also implement the Printer MIB [RFC-1759]~~[prtmib]~~ or a change in the Printer object's "printer-is-accepting-jobs" attribute.

2.10 Notification Subscription

End users may "subscribe" for notifications of Job events and/or Printer events when they submit a job. Operators, servers, and stand-alone applications may "subscribe for notifications of Job events and/or Printer events directly with the IPP Printer. These events include any of those described in the preceding section.

2.10.1 Job Submission Subscription

A Job Submission Subscription is a subscription submitted with a job for job events for that Job and/or Printer events while the job is "active" on that IPP Printer.

2.10.2 Printer Subscription

A Printer Subscription is a subscription submitted to a Printer using the Subscribe-For-Event-Notification operation for job events for any Job on that IPP Printer and/or Printer events for that IPP Printer.

2.11 Event Notification Content Attributes

When a Job or Printer event notification is delivered to the notification-recipient, it contains attributes whose values reflect the state of that Job or Printer at the time of the event, respectively. Examples of Job content attributes include:

"number-of-intervening jobs"
"job-impressions-completed"
"job-state-reasons"

Examples of Printer object content attributes include:

"printer-state-reasons"

"device-name"

"alert-code"

Note: when a Job event is sent, no Printer attributes, except the "printer-uri", are sent. When a Printer event is sent, no Job attributes are sent.

2.12 Immediate Notification

Notifications sent to the notification recipient or the notification recipient’s agent in such a way that the notification arrives immediately , within the limits of common addressing, routing, network congestion and quality of service.

2.13 Queued Notification

Notifications which are not necessarily sent immediately, but are queued for delivery by some intermediate network application, or for later retrieval. Email with store and forward is an example of queued notification.

2.14 Notification with Reliable Delivery

Notifications which are delivered by a reliable, sequenced delivery of packets or character stream, with acknowledgment and retry, such that delivery of the notification is guaranteed within some reasonable time limits. For example, if the notification recipient has logged off and gone home for the day, an immediate notification cannot be guaranteed to be delivered, even when sent over a reliable transport, because there is nothing there to catch it. Guaranteed delivery requires both queued notification and a reliable transport. If delivery of the notification requires process to process communications, each session is managed in a reliable manner, assuring fully ordered, end-to-end delivery.

2.15 Notification with Unreliable Delivery

Notifications are delivered via the fundamental transport address and routing framework, but no acknowledgment or retry is required. Process to process communications, if involved, are unconstrained.

2.16 Quality of Service

Some notification delivery methods may allow users to select quality of service parameters. These will depend upon the specific delivery method chosen, and may include parameters such as priority, security, number of retries, and the like.

2.17 Human Consumable Notification

Notifications which are intended to be consumed by human end users **only**. They contain no machine readable encodings of the event. Email would be an example of a Human consumable notification.

2.18 Machine Consumable Notification

Notifications which are intended for consumption by a program **only**, such as an IPP Client. Machine Consumable notifications may not contain human readable information.

2.19 Mixed Notification

A mixed notification may contain both human consumable and machine consumable information. Sending ’multi-part/alternative’ MIME media type is mixed notification, since both ’text/plain’ and a machine consumable content are sent.

3 Model for Job and Printer Event Notification

~~The following pictures~~ Figure 1 from the IPP/1.0 Model and Semantics [ipp-model] ~~are~~ is enhanced to show Job Submission Subscription (1) by end-user B to send notifications to notification recipients B and C and (2) using the new IPP Subscribe-For-Event-Notifications operation~~s~~ ~~event notifications~~ by operator client A to be sent to notification recipient A, by server D to be sent to server D, and by accounting application E to be sent to notification recipient E. ~~to (multiple) end-user notification-recipients and a system operator.~~

```
Legend:

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

      any   indicates any network protocol or direct
            connect, including IPP

       O A +----------+
      /|\  | client/  |----IPP Subscribe-For-Notification-+
      / \  | notif.   |                                   |
      oper-| recip.   |<---job and printer--------+       |
      ator +----------+    event notification     |       |
                                                  +       |
                                                   \      v
       O B +----------+                             \###########
      /|\  | client/  |----IPP job submission------->#  IPP    #
      / \  | notif.   |                              # Printer #
      end- | recip.   |<---job and printer event----o# Object  #
      user +----------+    notification            / ###########
                                                  /
       O C +----------+                          +
      /|\  | notifi-  |                          |
      / \  | cation   |<---job and printer-------+
      end- | recipient|    event notification
      user +----------+

                      +----IPP Subscribe-For-Notification----+
                      |                                      |
                      |                                      v
      IPP     D +----------+                         ###########
      jobs----->| server/  |----IPP job submission-->#  IPP    #
                | notif.   |                         # Printer #
      other---->| recipient|<---job and printer------# Object  #
      jobs      +----------+    event notification   ###########
                                                          |  ^
              E +----------+    job and printer           |  |
                | server/  |<--event notification---------+  |
                | notif.   |                                 |
                | recipient|--IPP Subscribe-For-Notification-+
                +----------+
                 accounting
```

Figure 1 - Model for Job and Printer Notification

Figure 2 shows the ~~An~~ implementation option ~~is~~ for the IPP Printer object to forward the subscription requests received in the job submission (from B) and with Subscribe-For-Event-Notification operations (from A, D, and E) to a notification service transparently to the requester. The IPP object then passes event notifications to this notification service to distribute the event notifications to the notification recipients (A, B, C, D, E, and F).

```
 O A +----------+
/|\  | client/  |<------------------------------------------+
/ \  | notif.   |                                           |
oper-| recip.   |----IPP Subscribe-For-Notification-+       |
ator +----------+                                   |       |
                                                    |       |
                                                    v       |
 O B +----------+                              ###########  |
/|\  | client/  |----IPP job submission------->#  IPP     #  |
/ \  | notif.   |                              # Printer #  |
end- | recip.   |<---job and printer--+        # Object  #  |
user +----------+    event notification\       ###########  |
                                        \           |       |
                                         \      subscribe   |
                                          \     & events   |
                                           \        |       |
 O C +----------+                           \  +----v---------+
/|\  | notifi-  |                            \ | notification |
/ \  | cation   |<---job and printer----------o| service      |
end- | recipient|    event notification        +--------------+
user +----------+


   D +----------+    job and printer event notification
     | server/  |<------------------------------------------+
     | notif.   |                                           |
     | recip.   |----IPP Subscribe-For-Notification-+       |
     +----------+                                   |       |
      accounting                                    |       |
other                                               v       |
jobs +----------+                              ###########  |
---->| server/  |----IPP job submission------->#  IPP     #  |
   E | notif.   |                              # Printer #  |
---->| recip.   |<---job and printer--+        # Object  #  |
IPP  +----------+    event notification\       ###########  |
jobs                                    \           |       |
                                         \      subscribe   |
                                          \      & events   |
                                           \        |       |
 O F +----------+                           \  +----v---------+
/|\  | notifi-  |                            \ | notification |
/ \  | cation   |<---job and printer----------o| service      |
end- | recipient|    event notification        +--------------+
user +----------+
```

Figure 2 - Model with Transparent Notification Service

4 Subscription for event notification

This section describes the mechanisms for subscribing for event notification.

4.1 Subscription as part of job submission

Subscription for event notifications is accomplished via IPP for end-user and server-to-device notifications related to the jobs being submitted. This proposal includes specifics for these types of subscriptions. Here the subscription information is submitted with the job and an implementation SHALL store the information with the Job object so that it may be queried with the Get-Job-Attributes operation.

As an implementation option, an implementation MAY employ an event notification service to keep the event notification subscription information and to actually deliver the event notifications. In this case, the IPP object passes each event as it occurs to the event notification service for event notification delivery to the notification recipients for which the Printer object had previously forwarded event notification subscriptions.

When the IPP Printer removes the job from the system, the subscription is automatically removed with such an implementation. If the IPP Printer object implementation uses a notification server, then the IPP object will have to ~~un-s~~unsubscribe with that notification server when the job completes.

4.2 Subscription independent of job submission

Subscription by servers that control IPP Printers and by 3rd party accounting or job monitoring applications, which are independent of job submissions, is accomplished by using the Subscribe-For-Event-Notification operation. In these cases, the subscription is in force, until the server or application performs an ~~Un-S~~Unsubscribe-For-Event-Notifications operation.

4.3 Semantics of Subscriptions

This sub-section summarizes the semantics of event notification subscriptions.

~~ISSUE 06: Ok if the semantics is duplicated here in the spec?~~

~~1.Job Events are changes in a Job object. Printer Events are changes in the Printer object.~~

~~2.~~1. Any subscription can contain either Job Events or Printer Events or both.

~~3.~~2. Subscriptions can be sent to the IPP Printer object either by being included in a create operation when the job is submitted (called "Job Submission Subscriptions") or by being sent in a separate subscription using the Subscribe~~pt~~-For-Event-Notifications operation (called "Printer Subscriptions).

~~4.~~3. For "Job Submission Subscriptions", the subscription is only valid while the job is "~~on the scene~~active". The job is "active" while it is in the ’pending’, ’processing’, and ’processing-stopped’ states. The job ceases to be active when it enters the ’pending-held’ state or ~~on the scene from the time the IPP Job object is created and enters either the ’pending’ or ’held’ states~~ until the time it is "done" and enters any of the 'completed', 'canceled', or 'aborted' states. When the job is released from the 'pending-held' state, it becomes active again.

~~5.~~4. For "Printer Subscriptions", the subscription is valid until it is explicitly ~~un-s~~unsubscribed with an ~~Un-S~~Unsubscribe-For-Event-Notifications operation.

~~6.~~5. Job Events in a "Job Submission Subscription" ONLY apply to "this job" (the Job object created because of the job create operation).

~~7.~~6. Job Events in a "Printer Subscription" apply to ALL jobs contained in the IPP Printer object.

~~8.~~7. Subscriptions indicate the delivery method and destination for each set of events being subscribed to. For example, an application may submit a job with a "Job Submission Subscription" indicating that some events should be sent back to it (using some new HTTP based event delivery mechanism using it own address), some events should be sent to a 3rd party accounting/monitoring application (using the same HTTP based event delivery mechanism but with the address of the 3rd party app, not its own address), and finally that some events should be sent to a 3rd party human being (using email and the email address of that human being).

Implemented another way, the 3rd party accounting/management app could subscribe to all job events using a persistent (until ~~un-s~~unsubscribed) "Printer Subscription" indicating its own address as the address for delivery of events.

~~9.~~8. Once a job is submitted, no~~Any~~ subscription (neither a "Job Submission Subscription" nor a "Printer Subscription" is allowed for subsequently subscribing Job Events to this~~a~~ specific (named or otherwise identified) Job only.

5 New Operation attribute for the create operations

This section specifies the single "job-notify" Operation attribute that supplies one or more Job Submission ~~Notification~~ Subscriptions as part of a job create operation.

5.1 job-notify (1setOf collection (1023))

The client OPTIONALLY supplies this Operation attribute as a *collection* attribute as part of the Validate-Job, Print-Job, Print-URI, and Create-Job operations. The Printer object OPTIONALLY supports this Operation attribute as part of the Validate-Job, Print-Job, Print-URI, and Create-Job operations. If the Printer object supports this attribute for any of these create operations, it MUST support it for all of these create operations that it supports.

The "job-notify" Operation attribute specifies the Job Submission ~~Notification~~ Subscription that starts when the job is created and ends when the job completes (enters the 'completed', 'aborted', or 'canceled' job states). The subscription may request Job Events and/or Printer Events. The Job Events SHALL apply only to changes in this job (the one being created), while the Printer Events apply to all job. (Note: The Job Events requested with the Subscribe-For-Event-Notifications operation SHALL apply to all jobs, just as for Printer Events).

5.1.1 Notification collection value

The value of this attribute is one or more collection values. Each collection value SHALL contain a "notify-recipients" member attribute and MAY contain any of the remaining following *member* attributes with the indicated syntax:

Member attribute name	syntax	in request	support
"notify-event-groups"	1setOf type2 keyword	MAY	mandatory
"notify-recipients"	1setOf uri	SHALL	mandatory
"notify-content-type"	mimeMediaType	MAY	mandatory
"notify-charset"	charset	MAY	mandatory
"notify-natural-language"	naturalLanguage	MAY	optional
"notify-additional-attributes"	1setOf keyword	MAY	optional

The "support" column indicates the support required by the IPP object if it supports the "job-notify" Operation attribute at all.

If the client supplies this Operation attribute, but the number of octets in any collection value exceeds 1023 octets, the Printer object SHALL reject the request and return the 'client-error-bad-request' status code, since the syntax is not correct.

If the client supplies this Operation attribute, but does not supply the "notify-recipients" member attribute as one of the attributes in (each) collection value, the Printer object SHALL reject the request and return the 'client-error-bad-request' status code, since the syntax is not correct.

If the client supplies this Operation attribute (like the "job-k-octets", "job-impressions", and "job-media-sheets" Operation attributes, see [ipp-model]), but the Printer object does not support the "job-notify" Operation attribute, the Printer object SHALL ignore the "job-notify" attribute and copy it to the Unsupported Attribute group with the out-of-band value of 'not-supported'.

If the client supplies the "job-notify" Operation attribute and the Printer object supports the "job-notify" Operation attribute, the collection value(s) of the attribute are used to populate the job object's "job-notify" Job Description attribute (see section 8) according to the following conditions:

If the values of the member attributes are within the range of the corresponding Printer object's "xxx-supported" attributes (see section 8), the Printer object SHALL use the collection value(s) to populate the job object's "job-notify" Job Description attribute.

If some of the member attributes are not supported, i.e., the corresponding Printer object's xxx-supported" attributes (see section 8) do not exist, the Printer object SHALL copy such member attributes to the Unsupported Attributes response group with the out-of-band value of 'not-supported', copy the remaining (supported) member attributes to the job object's "job-notify" Job Description attribute, accept the request, and return the 'successful-ok-ignored-or-substituted-attributes' status code.

If some of the member attribute values are outside the range of the corresponding Printer object's "xxx-supported" attributes (see section 8), the Printer object SHALL copy such member attributes and their values to the Unsupported Attributes response group, substitute or ignore the supplied values, copy the remaining (supported) member attribute values to the job object's "job-notify" Job Description attribute, accept the request, and return the 'successful-ok-ignored-or-substituted-attributes' status code.

The following attributes are defined for use in one or more collection values of the "job-notify" Operation attribute in the create operation:

5.1.2 notify-event-groups (1setOf type2 keyword)

The client OPTIONALLY supplies this attribute as a member of the "job-notify" Operation attribute. The Printer object SHALL support this attribute if it supports the "job-notify" Operation attribute. This attribute specifies one or more Job event groups and/or Printer event groups for which the IPP client desires some sort of notification to be sent to one or more notification recipients that the client supplies in the same "job-notify" collection value in the create request for this job.

Each event is assigned a keyword value (see section 5.1.2.2). Each of the events is assigned to one or more of the standard event groups. Each standard group is also assigned a keyword (see section 5.1.2.1), in order to simplify (1) client subscription for the events supplied by the client and (2) event filtering by the notification mechanism.

~~ISSUE 07: Should a requester be able to supply either event group names and/or specific event keywords, or is it ok to require only event group names?~~

5.1.2.1 Notification Groups

This section defines the event groups that a client may subscribe for in the create operation. These event group keywords (not the actual event keywords themselves) are passed as attribute values in the "notify-event-groups" Operation attribute in the create request. There are Job event groups and Printer event groups. An IPP object SHALL support the following~~all~~ event groups: ’none’, ’all-job-events’, job-completion’, ’all-printer-events’, and ’printer-errors’. Support of all of the events in a group is not required.

~~ISSUE 08: Ok if all groups are required for conformance?~~

Standard event group values are:

’none’: (MANDATORY) no notifications of any events. This value is useful to prevent notifications when the client has default notification attributes configured.

’all-job-events’: (MANDATORY) any job events.

~~ISSUE 09: Ok if I split ’all’ into two, now that we have both kinds?~~

’job-~~delivery~~’state-changes’: any of the following report events which~~, in general, pertain to the progress of delivering the job to the Printer~~ are state changes while the job is ’active’:

’job-received’, ’job-held’, ’job-released’, ’job-started-processing’

’job-progress’: any of the following report events which, in general, pertain to the progress of ~~pending or~~ actually interpreting, marking, finishing or otherwise processing the job by the Printer object:

~~’job-held’, ’job-released’,~~ ’sheet-completed’, ’collated-copy-completed’

’job-warnings’: any of the following warning events which do not abort the job and do not require human intervention, such as the interpreter encountering a request for a missing font, but for which it is able to perform font substitution:

’job-warning’

Any device warnings, such as ’toner-low’, SHALL be ’printer-warnings’, NOT ’job-warnings’.

’job-errors’: any of the following error events which stop that job from further processing:

’job-aborted’

Any device errors, such as ’toner-out’, SHALL be ’printer-errors’, not ’job-errors’

NOTE: the ’job-aborted’ error event is in both the ’job-errors’ group and the ’job-completion’ group, because it is both an error event and also indicates that the job has finished.

’job-completion’: (MANDATORY) any of the following events which, in general, pertain to ways that a job can end:

’job-completed’, ’job-aborted’, ’job-canceled’

’all-printer-events’: (MANDATORY) any printer events.

’printer-reports’: any Printer object or device report event that are informational, as opposed to warnings or errors. Printer MIB events that fall in this report group included the alertRemovalOfBinaryChangeEntry(1801) alert that indicates that a binary change event entry row has been removed from the Alert Table and any event with the prtAlertSeverityLevel value set to noInterventionRequired(7) [draft-prtmib]. See section 9.3. Standard events in this group are:

’printer-report’, ’printer-accepting-jobs’, ’printer-not-accepting-jobs’

’printer-warnings’: any Printer object or device event that are warnings, i.e., non-critical alert where the Printer object’s "printer-state" attribute remains in the ’processing’ state and the device(s) continue to operate. However, if there is not human intervention soon, the device will stop. Standard events in this group are:

’printer-warning’

Examples of the ’printer-warning’ event include: inputMediaSupplyLow(807)~~paper-low~~ and markerTonerAlmostEmpty(1104)~~toner-low~~ prtAlertCode. See section 9.3. Warning events may be either binary or unary [see draft-prt-mib]. A binary event is one in which a second event terminates the warning. Examples include: paper low and toner low. A unary event is one in which there is not a second event that terminates the warning. If an IPP Printer is controlling more than one device and one of the devices stops, then the IPP Printer SHALL generate either a warning event or an error event, depending on implementation and cite policy.

~~ISSUE 10: What if a Printer object controls several devices and one of them stops. The "printer-state" remains in ’processing’, but it should be a Printer error, since some device stopped.~~

’printer-errors’: (MANDATORY) any Printer object or device event that is an error~~s~~, i.e., critical alert where the Printer object’s "printer-state" attribute changes to ’stopped’ or (at least one of) the devices stop (even though other devices that the Printer object controls, continue to operate). Standard events in this group are:

’printer-error’

Examples of the ’printer-error’ event include: jammed(8) and markerTonerEmpty(1101) prtAlertCode. See section 9.3.

Note: The ’job-warnings’, ’printer-warnings’, and ’printer-errors’ event groups each have a corresponding ’job-warning’, ’printer-warning’, and ’printer-error’ event (singular), respectively.

Implementers MAY add additional events to a group. Therefore, notification recipients SHOULD check the event that is sent in the notification content (see section 6) to make sure that it is an event that is wanted. Implementors SHOULD NOT add new groups, lest interoperability will be lessened.

In a create request, if the client supplies ’none’ along with any other combination of values, it is the same as if only that other set of values had been supplied (i.e., the ’none’ value has no affect). If the client supplies ’all’ along with any other combination of values, it is the same as if only ’all’ had been supplied (i.e., the ’all’ value subsumes all other values).

Note: the group ’job-progress’ is intended for those who wish to receive more frequent, "real-time" progress notifications on a page and copy boundary basis. This is why ’job-started-printing’ is in the ~~’delivery’~~ ’job-state-change’ group, rather than the ’progress’ group, for example. An application which was interested in less granular milestones of print job progress would likely subscribe for 'job-completion' and 'printer-errors' event groups (only).

5.1.2.2 Notification Events

This section defines the notification events. Each event is a member of one or more event groups. Each event is categorized as being a report event, warning event, or error event. See section 2.9. When an event occurs, the event keyword, not the event group, is included in the notification content (see section 9.1.2).

The standard event values are:

Job object events:

'job-received': when the Printer object accepts the job (i.e., when the job is created entering the 'pending' or 'pending-held' [JMP 'pendingHeld' states] [JMP: issued by the agent when the agent creates a row in the MIB for that job.]

'job-started-processing': the Printer starts processing the Job (i.e., when the job leaves the 'pending' state and enters the 'processing' state).

'sheet-completed': when each sheet in the job is completed (i.e., stacked in the output bin).

'collated-copy-completed': when each document copy in the job is completed (i.e., last sheet of a collated copy is stacked in an output bin)

'job-held': when the job enters the 'pending-held' (JMP pendingHeld) state (using some protocol operation not defined in IPP/1.0, but perhaps in another protocol or added as an extension), or the system or device holds the job because of some requirement that cannot be met and other jobs could be processed, if there are any.

'job-released': when the job leaves the 'pending-held' (JMP pendingHeld) state entering the 'pending' or 'processing' states due to the user, operator, or system releasing the held job (using some protocol operation not defined in IPP/1.0, but perhaps in another protocol or added as an extension).

'job-warning': when the job encounters a warning. See the definition of the 'job-warnings' event group.

~~'job-error': when the job encounters a problem (i.e., when the job leaves the 'processing' state and enters the 'processing-stopped' state)~~

'job-completed': when the job completes processing (with or without errors or warnings) and enters the 'completed' state.

'job-aborted': when the job was aborted by the system while in the 'processing' or 'processing-stopped' state, due to some encountered problem that cannot be remedied by human intervention.

'job-canceled': when the job was canceled by the user or operator using the Cancel-Job operation while the job was in any state.

Printer object events:

'printer-report': when the Printer issues a non-warning and non-error.

'printer-accepting-jobs': when the Printer is powered up and the Printer object's "printer-is-accepting-jobs" attribute is 'true' or when the Printer object's "printer-is-accepting-jobs" is changed from 'false' to 'true'.

’printer-not-accepting-jobs’: when the Printer is powered up and the Printer object’s "printer-is-accepting-jobs" attribute is ’false’ or when the Printer object’s "printer-is-accepting-jobs" is changed from ’false’ to ’true’.
’printer-warning’: when the Printer issues a non-critical event and continues in the ’processing’ state.
’printer-error’: when the Printer issues a critical event and enters the ’stopped’ state.

5.1.3 notify-recipients (1setOf uri)

The client ~~OPTIONALLY~~ SHALL supply~~ies~~ this attribute as a member of the "job-notify" Operation attribute. The Printer object SHALL support this attribute if it supports the "job-notify" Operation attribute and SHALL support the ’ipp-tcpip-socket~~mailto~~’ scheme at least~~.~~. If the client does not supply the "notify-recipients" member attribute as one of the attributes in (each) collection value, the Printer object SHALL reject the request and return the ’client-error-bad-request’ status code, since the syntax is not correct.

~~ISSUE 11: Is it too hard to require an embedded device to include sending e-mail?~~

This attribute describes both where (the address) and how (the mechanism for delivery) events are to be delivered. The Printer object SHALL use this attribute as the set of addresses and methods for sending notifications when one of the events occurs that the client supplied in the "notify-event-groups" member attribute in the same "job-notify" collection value in the create request for this job.

The Printer object MAY achieve the subscription and event notification delivery either (1) itself or (2) by using some (unspecified) notification service that supports the requested mechanism of notifying the notification recipients. Either implementation choice SHALL be transparent to clients and notification-recipients.

Each scheme value has at least one MANDATORY "notify-content-type" (see section 5.1.4) that a Printer object MUST support if it supports the indicated scheme and possibly additional OPTIONAL "notify-content-type". Standard uriScheme values are:

’mailto’: a ~~text~~ message via email to the specified email address. The IPP Printer SHALL support the ’application/ipp’ "notify-content-type" and MAY support the ’text/plain’ if it supports this scheme.
~~’http’: an HTML formatted message via an HTTP POST method to the specified URI~~
’ftp’: a ~~text~~ message via an FTP ’append’ command to the specified remote file. The IPP Printer SHALL support the ’application/ipp’ "notify-content-type" and MAY support the ’text/plain’ if it supports this scheme.

The following values are not yet standardized or registered. Some of them represent work in progress. They will be registered following the procedures [url-reg]. See also [cohen] for HTTP URL schemes for notification.

Note: the ’ipp-tcpip-socket’ method is MANDATORY, so we will progress its standardization and registration in parallel with this document.
~~ISSUE 12: Which schemes do we want to progress?~~

’ipp-tcp-ip-socket’: (MANDATORY) an IPP notification via a TCP/IP socket that is opened by the Printer object on the IP address specified in the URI (using IP address dot notation) using the port on that host specified using the /port=nnn keyword. The IPP Printer SHALL support the ’application/ipp’ "notify-content-type" and MAY support the ’text/plain’ if it supports this scheme. For example:

ipp-tcp-ip-socket:13.240.120.138/port=6000

would cause the Printer object to open the TCP/IP port 6000 at IP address 13.240.120.138.

~~ISSUE 13: Ok that I removed this note, since the printer-uri is being returned in all event notifications?~~

’snmpv1’: a notification as an SNMPv1 trap to the host specified as the address in the URI. The IPP Printer SHALL support the ’snmpv1-trap’ "notify-content-type" if it supports this scheme.

’snmpv2’: a notification as an SNMPv2 inform to the host specified as the address in the URI. The IPP Printer SHALL support the ’snmpv2-inform’ "notify-content-type" if it supports this scheme.

’snmpv3’: a notification as an SNMPv3 inform to the host specified as the address in the URI. The IPP Printer SHALL support the ’snmpv3-inform’ "notify-content-type" if it supports this scheme.

’http’: a message via a new HTTP method to the specified URI [see cohen]. The IPP Printer SHALL support the ’application/ipp’ "notify-content-type" and MAY support the ’text/plain’ if it supports this scheme.

’sense’: a notification as a SENSE UDP data gram that is opened by the Printer object on the IP address specified in the URI (using IP address dot notation) using the port on that host specified using the /port=nnn keyword. See the ’ipp-tcp-ip-socket’ example. The IPP Printer SHALL support the ’application/ipp’ "notify-content-type" if it supports this scheme.

’page’: a pager phone number to call as specified by the /phone-number parameter in the URL. The IPP Printer SHALL support the ’text/plain’ "notify-content-type" if it supports this scheme.

The Printer object SHALL validate that the schemes supplied in the "notify-recipients" is supported by comparing with the Printer object’s "notify-schemes-supported".

5.1.4 notify-content-type (mimeMediaType)

The client OPTIONALLY supplies this attribute as a member of the "job-notify" Operation attribute. The Printer object SHALL support this attribute if it supports the "job-notify" Operation attribute~~ and SHALL support the ’multi-part/alternative’, ’application/ipp’, and the ’text/plain’ values for all event groups~~.

~~ISSUE 14: Ok to require supporting all three values? Ok for all event groups?~~

This attribute specifies the type of content that is sent in the notification. Thus the client can control whether the event notification content is human readable, machine readable, or both.

If the MIME media type registration permits a charset parameter, the~~a~~n such a specification SHALL be used (instead of the "notify-charset" member attribute) in order to indicate the charset to be used in the notification content.

Standard values are:

’multi-part/alternative’ - contains both human consumable notification content using the ’text/plain’ MIME media type and machine consumable notification content using the ’application/ipp’ MIME media type with the Get-Job-Attributes response encoding of the attributes listed in Table 2 or the Get-Printer-Attributes response encoding of the attributed listed in Table 3. This value SHALL be supported and is the default, if the client does not supply the "notify-content-type" member attribute.

~~ISSUE 15: Should we make this attribute 1setOf so that the additional values could specify which alternatives are to be used with ’multi-part/alternative’?~~

’application/ipp’ - the machine consumable notification content using the ’application/ipp’ MIME media type [ipp-model] with the Get-Job-Attributes response encoding of the attributes listed in Table 2 or the Get-Printer-Attributes response encoding of the attributed listed in Table 3.

’text/plain’ - the human consumable notification content. If the charset is other than US-ASCII, the /charset parameter SHALL be included in the value of this attribute and in the event notification content. See the ’mimeMediaType’ attribute syntax in section 4.1.11 in [ipp-model].~~RFC 2046 indicates that the absence of the charset parameter SHALL mean US-ASCII rather than simply unspecified [RFC2046]. Examples:~~

~~’text/plain’: A plain text document in US-ASCII [US-ASCII]~~

~~’text/plain; charset=US-ASCII’: A plain text document in US-ASCII.~~

~~’text/plain; charset=ISO-8859-1’: A plain text document in ISO 8859-1 (Latin 1) [ISO8859-1].~~

~~’text/plain; charset=utf-8’: A plain text document in ISO 10646 represented as UTF-8 [RFC 2044]~~

~~’text/plain, charset=iso-10646-ucs-2’: A plain text document in ISO 10646 represented in two octets (UCS-2) [ISO10646-1]~~

’snmpv1-trap’ - SNMPv1 traps.

’snmpv2-inform’ - SNMPv2 informs.

’snmpv3-inform’ - SNMPv3 informs.

5.1.5 notify-charset (charset)

The client OPTIONALLY supplies this attribute as a member of the "job-notify" Operation attribute. The Printer object SHALL support this attribute if it supports the "job-notify" Operation attribute.

This attribute specifies the charset to be used in the human readable part of the notification content that is sent to the notification recipients that the client supplied in this same collection value. This attribute SHALL NOT be used when the "notify-content-type" attribute value specifies the charset parameter in its MIME media type value.

If the "notify-charset" attribute is not supplied, the charset supplied in the "attributes-charset" Operation attribute SHALL be used, if the charset value is supported by the Printer, else the Printer object shall use the Printer's "charset-configured" value.

5.1.6 notify-natural-language (naturalLanguage)

The client OPTIONALLY supplies this attribute as a member of the "job-notify" Operation attribute. The Printer object OPTIONALLY supports this attribute if it supports the "job-notify" Operation attribute.

This attribute specifies the natural language for the IPP object to use in the human readable part of the notification content is sent to the notification recipients that the client supplied in this same collection value. If this attribute is not supported or the supplied value is not supported, the IPP Printer SHALL return the attribute in the Unsupported Attributes Group but still accept the operation, as with all create operations. If this attribute is not supplied or the attribute or value is not supported by the Printer object, the natural language supplied in the "attributes-natural-language" create operation attribute SHALL be used, if that natural language value is supported by the Printer, else the Printer object SHALL use the Printer's "natural-language-configured" value. See the Print-Job operation in [ipp-model].

5.1.7 notify-additional-attributes (1setOf keyword)

The client OPTIONALLY supplies this attribute as a member of the "job-notify" Operation attribute. The Printer object OPTIONALLY supports this attribute if it supports the "job-notify" Operation attribute.

This attribute specifies the additional attributes that the requester wishes to be included in the notification content, in addition to the fixed set that depends on the event as shown in the table in section 6. If this attribute is not supported or not supplied by the client, the Printer object SHALL supply the fixed set of attributes indicated in section 6 depending on the event being requested.

6 Operations to Subscribe and ~~Un-s~~Unsubscribe for notifications

There are two new OPTIONAL operations to allow a client or server to subscribe for Printer object events without submitting a job. An IPP Printer SHALL support both of these operations, if it supports either one. If an IPP Printer supports these operations, it SHALL also support the "job-notify" attribute in the create operations as described in section 5.

These new operations are intended for use by servers that control printers, by clients used by operators/administrators that manage printers, and by applications that collect accounting data.

6.1 Subscribe-For-Event-Notifications Operation

This OPTIONAL operation allows a client to subscribe with the Printer object to be notified when identified events happen to the device(s) that the Printer object is representing without requiring that the client submit jobs. In the request, the client supplies the set of Job event group names and/or Printer event group names in which the notification-recipient(s) are interested. In the response, the Printer object returns a list of the current subscriptions, including the new one requested by this operation.

This operation is intended for use by system operators and administrators that have a long term interest in the events without submitting jobs. It is also intended to be used by servers that control IPP Printers. Finally, it is also intended to be used by accounting applications that need to be notified when jobs complete.

The possible names of Job and Printer event groups are the same as for use in the "job-notify" Operation attribute in create requests. See section 5.1.2.1. An IPP object SHALL support ~~all~~ the same event groups for use in this operation as it supports for use in the create operations. Support of all of the events in a group is not required. See section 5.1.2.2.

~~ISSUE 16: Ok if all groups are required for conformance?~~

6.1.1 Subscribe-For-Event-Notifications Request

The following sets of attributes are part of the Subscribe-For-Event-Notifications Request:

Group 1: Operation Attributes

Target:
The "printer-uri" operation attribute which is the target for this operation as described in section 3.1.3.

Natural Language and Character Set:
The "attributes-charset" and "attributes-natural-language" attributes as described in section 4.3.23 and 4.3.24.

Requesting User Name:

The "requesting-user-name" attribute SHOULD be supplied by the client as described in section 8.3.

"printer-notify" (collection(1023)) :

The client SHALL supply a "printer-notify" Operation attribute that MUST specify the notification-recipient(s), and MAY specify additional information about the subscription. The Printer object SHALL support this Operation attribute (if it supports this OPTIONAL operation). The value of this attribute is one collection value. The collection value SHALL contain a "notify-recipients" member attribute and MAY contain any of the other *member* attributes defined for use with the "job-notify" Operation attribute in create operations (see section 5.1). If the client omits this attribute, the Printer SHALL reject the operation and return the ’client-error-bad-request’ status code.

Note: only one collection value is permitted, so that each collection value will have its own "notify-subscription-id".

The Printer object SHALL validate that this client is permitted to subscribe for Printer notifications. The means for configuring the permissions is outside the scope of this specification. If a requester is not permitted to subscribe for Printer notifications, the IPP Printer SHALL reject the request and return the ’client-error-authenticated’ or ’client’-error-not-authorized’ status code.

If the same subscription (same client and same collection value~~s~~) has already been made as indicated in one of the collection values of the Printer object’s "printer-notify" Description attribute, the IPP Printer SHALL reject the request and return the ’client-error-~~not-possible~~already-subscribed’ status code.

~~ISSUE 17: Or should we add a new status code that is more specific, such as ’client-error-already-subscribed’.~~

If the IPP Printer object forwards the subscription to a notification service which rejects the subscription for whatever reason, the IPP Printer SHALL return the ’client-error-subscription-rejected’ along with the "subscription-error-status" Operation attribute in the response. The IPP Printer SHOULD return a "status-message" Operation attribute that contains any text message that the notification service makes available that explains the error as well.

If the IPP Printer object accepts the request, it SHALL add the subscription collection value to the Printer object’s "printer-notify" attribute. The Printer object SHALL add a "notify-subscription-id" member attribute with a unique integer id and SHALL return the "notify-subscription-id" Operation attribute in the response. Later, the requester uses t~~T~~his id ~~is used~~ to ~~un-s~~unsubscribe using the ~~Un-S~~Unsubscribe-For-Event-Notifications operations.

6.1.2 Subscribe-For-Event-Notifications Response

The Printer object returns the following sets of attributes as part of Subscribe-For-Event-Notifications Response:

Group 1: Operation Attributes

Status Code and Message:
The response includes the MANDATORY status code and an OPTIONAL "status-message" (text) operation attribute as described in section 3.1.5.

Natural Language and Character Set:
The "attributes-charset" and "attributes-natural-language" attributes as described in section 3.1.4.2.

"notify-subscription-id" (integer(1:MAX)):
The Printer object MUST return the unique integer id for the accepted subscription to be used subsequently to ~~un-s~~unsubscribe using the ~~Un-S~~Unsubscribe-For-Event-Notifications operation. ~~This value SHOULD NOT be re-used too soon after subscription in order to avoid confusion in subsequent Un-Scribe-For-Event-Notification operations.~~

"subscription-error-status" (integer(-MAX:MAX)) | keyword):
If the Printer object uses a third party notification service and that notification service rejected the subscription, the Printer object SHALL return the error status code from a subscription service that indicates the reason for the rejection.

Group 2: Unsupported Attributes

This is a set of Operation (member) attributes supplied by the client (in the request) that are not supported by the Printer object or that conflict with one another (see sections 15.3 and 15.4).

Group 3: Printer Object Attributes

The updated "printer-notify" attribute that contains the requested subscription supplied in this operation request, along with any that have been previously subscribed by any client.

6.2 Unsubscribe-For-Event-Notifications Operation

This OPTIONAL operation allows a client to ~~un-s~~unsubscribe with the Printer object for event notifications that had been subscribed to previously using the Subscribe-For-Event-Notification operation. In the request, the client supplies the "notify-subscription-id" Operation attribute that the Printer object created and returned in the Subscribe-For-Event-Notifications operation. In the response, the Printer object returns a list of the current subscriptions which SHALL NOT include the one removed by this operation.

This operation is intended for use by system operators and administrators that have a long term interest in the events without submitting jobs. It is also intended to be used by servers that control IPP Printers. Finally, it is also intended to be used by accounting applications that need to be notified when jobs complete.

6.2.1 Unsubscribe-For-Event-Notifications Request

The following sets of attributes are part of the ~~Un-S~~Unsubscribe-For-Event-Notifications Request:

Group 1: Operation Attributes

Target:
The "printer-uri" operation attribute which is the target for this operation as described in section 3.1.3.

Natural Language and Character Set:
The "attributes-charset" and "attributes-natural-language" attributes as described in section 3.1.4.1.

Requesting User Name:
The "requesting-user-name" attribute SHOULD be supplied by the client as described in section 8.3.

"notify-subscription-id" (integer(1:MAX)) :
The client SHALL supply a "notify-subscription-id" Operation attribute that specifies a subscription id assigned by the Printer object in a previous Subscribe-For-Event-Notifications. The Printer object MUST support this Operation attribute (if it supports this OPTIONAL operation). If the client omits this attribute, the Printer SHALL reject the operation and return the ’client-error-bad-request’ status code. The Printer object SHOULD NOT re-use this id value too soon after unsubscription in order to avoid confusion in subsequent Subscribe-For-Event-Notification operations.

The Printer object SHALL validate that this client is permitted to ~~un-s~~unsubscribe notifications in general and this notification subscription in particular. The means for configuring the permissions is outside the scope of this specification.

If a requester is not permitted to ~~un-s~~unsubscribe for notifications in general or for the requested subscription, the IPP Printer SHALL reject the request and return the ’client-error-authenticated’ or ’client’-error-not-authorized’ status code. The means for keeping track of which clients requested each subscription is not specified by this document and is implementation dependent. For example, an implementation might add an additional "client-id" member attribute to each subscription value of the Printer object’s "printer-notify" Description attribute, that is not returned to non-privileged users.

If the value of the "notify-subscription-id" is not found, the IPP Printer SHALL reject the request and return the ’client-error-not-found’ status code.

If the IPP Printer object accepts the request, it SHALL remove the requested event notification subscription from the Printer object's "printer-notify" attribute. Clients SHOULD remove subscriptions that are no longer wanted using this operation.

6.2.2 Unsubscribe-For-Event-Notifications Response

The Printer object returns the following sets of attributes as part of the ~~Un-S~~Unsubscribe-For-Event-Notifications Response:

Group 1: Operation Attributes

Status Code and Message:
The response includes the MANDATORY status code and an OPTIONAL "status-message" (text) operation attribute as described in section .

Natural Language and Character Set:
The "attributes-charset" and "attributes-natural-language" attributes as described in section 3.1.4.2.

Group 2: Unsupported Attributes

This is a set of Operation (member) attributes supplied by the client (in the request) that are not supported by the Printer object or that conflict with one another (see sections 15.3 and 15.4).

Group 3: Printer Object Attributes

The updated "printer-notify" attribute that no longer contains the event notification subscription that was requested to be removed.

7 Job Object Description attributes for ~~Job~~ Notification

This section specifies the Job object Description attributes for notification.

7.1 "job-notify" (1setOf collection(1023))

This attribute specifies one or more Job Submission Subscriptions (collections of events, notification-recipients, and other member attributes) that the client supplied in the "job-notify" Operation attribute of the create request for this job. The Printer object SHALL support this Job Description attribute if it supports the "job-notify" Operation attribute in create operations.

The IPP Printer object SHALL populate the value(s) of this attribute with the collection value(s) supplied by the "job-notify" Operation attribute in the create operation that created this job. See the description of the "job-notify" Operation attribute for the complete specification of the semantics of this Job Description attribute.

7.2 Job Attributes for Monitoring Job Progress

There are a number of objects and attributes for monitoring the progress of a job. These objects and attributes count the number of K octets, impressions, sheets, and pages requested or completed. For impressions and sheets, "completed" SHALL mean stacked, unless the implementation is unable to detect when each sheet is stacked, in which case stacked is approximated when processing of each sheet completes. There are objects and attributes for the overall job and for the current copy of the document currently being stacked. For the latter, the rate at which the various objects and attributes count depends on the sheet and document collation of the job.

Job Collation included sheet collation and document collation. Sheet collation is defined to be the ordering of sheets within a document copy. Document collation is defined to be ordering of document copies within a multi-document job. There are three types of job collation (see terminology definitions in Section **Error! Reference source not found.**):

1. uncollatedSheets(3) - No collation of the sheets within each document copy, i.e., each sheet of a document that is to produce multiple copies is replicated before the next sheet in the document is processed and stacked. If the device has an output bin collator, the uncollatedSheets(3) value may actually produce collated sheets as far as the user is concerned (in the output bins). However, when the job collation is the ’uncollatedSheets(3)’ value, job progress is indistinguishable to a monitoring application between a device that has an output bin collator and one that does not.

2. collatedDocuments(4) - Collation of the sheets within each document copy is performed within the printing device by making multiple passes over either the source or an intermediate representation of the document. In addition, when there are multiple documents per job, the i’th copy of each document is stacked before the j’th copy of each document, i.e., the documents are collated within each job copy. For example, if a job is submitted with documents, A and B, the job is made available to the end user as: A, B, A, B, …. The 'collatedDocuments(4)' value corresponds to the IPP [ipp-model] 'separate-documents-collated-copies' value of the "multiple-document-handling" attribute.

   If jobCopiesRequested or documentCopiesRequested = 1, then jobCollationType is defined as 4.

3. uncollatedDocuments(5) - Collation of the sheets within each document copy is performed within the printing device by making multiple passes over either the source or an intermediate representation of the document. In addition, when there are multiple documents per job, all copies of the first document in the job are stacked before the any copied of the next document in the job, i.e., the documents are uncollated within the job. For example, if a job is submitted with documents, A and B, the job is mad available to the end user as: A, A, …, B, B, …. The 'uncollatedDocuments(5)' value corresponds to the IPP [ipp-model] 'separate-documents-uncollated-copies' value of the "multiple-document-handling" attribute.

Consider the following four variables that are used to monitor the progress of a job's impressions:

1. jmJobImpressionsCompleted - counts the total number of impressions stacked for the job
2. impressionsCompletedCurrentCopy - counts the number of impressions stacked for the current document copy
3. sheetCompletedCopyNumber - identifies the number of the copy for the current document being stacked where the first copy is 1.
4. sheetCompletedDocumentNumber - identifies the current document within the job that is being stacked where the first document in a job is 1. NOTE: this attribute SHOULD NOT be implemented for implementations that only support one document per job.

For each of the three types of job collation, a job with three copies of two documents (1, 2), where each document consists of 3 impressions, the four variables have the following values as each sheet is stacked for one-sided printing:

"job-collation-type" = uncollated-sheets(3)

"job-impressions-completed"	"impressions-completed-current-copy"	"sheet-completed-copy-number"	"sheet-completed-document-number"
0	0	0	0
1	1	1	1
2	1	2	1
3	1	3	1
4	2	1	1
5	2	2	1
6	2	3	1
7	3	1	1
8	3	2	1
9	3	3	1
10	1	1	2
11	1	2	2
12	1	3	2
13	2	1	2
14	2	2	2
15	2	3	2
16	3	1	2
17	3	2	2
18	3	3	2

"job-collation-type" = collated-documents(4)

"job-impressions-completed"	"impressions-completed-current-copy"	"sheet-completed-copy-number"	"sheet-completed-document-number"
0	0	0	0
1	1	1	1
2	2	1	1
3	3	1	1
4	1	1	2
5	2	1	2
6	3	1	2
7	1	2	1
8	2	2	1
9	3	2	1
10	1	2	2
11	2	2	2
12	3	2	2
13	1	3	1
14	2	3	1
15	3	3	1
16	1	3	2
17	2	3	2
18	3	3	2

"job-collation-type" = uncollated-documents(5)

"job-impressions-completed"	"impressions-completed-current-copy"	"sheet-completed-copy-number"	"sheet-completed-document-number"
0	0	0	0
1	1	1	1
2	2	1	1
3	3	1	1
4	1	2	1
5	2	2	1
6	3	2	1
7	1	3	1
8	2	3	1
9	3	3	1
10	1	1	2
11	2	1	2
12	3	1	2
13	1	2	2
14	2	2	2
15	3	2	2
16	1	3	2
17	2	3	2
18	3	3	2

7.2.1 "output-bin" (1setOf text(63))

The name or number (represented as ASCII digits) of the output bin to which all or part of the job is placed in.

7.2.2 "sheet-completed-copy-number" (integer(-2:MAX))

The number of the copy being stacked for the current document. This number starts at 0, is set to 1 when the first sheet of the first copy for each document is being stacked and is equal to n where n is the nth sheet stacked in the current document copy. See section 7.2.

7.2.3 "sheet-completed-document-number" (integer(-2:MAX))

The ordinal number of the document in the job that is currently being stacked. This number starts at 0, increments to 1 when the first sheet of the first document in the job is being stacked, and is equal to n where n is the nth document in the job, starting with 1.

Implementations that only support one document jobs SHOULD NOT implement this attribute.

7.2.4 "job-collation-type" (enum)

The type of job collation. See section 7.2,

Standard values are:

```
other(1),
unknown(2),
uncollatedSheets(3),           -- sheets within each document copy
                               -- are not collated: 1 1 ..., 2 2 ...,
collatedDocuments(4),          -- internal collated sheets,
                               -- documents: A, B, A, B, ...
uncollatedDocuments(5)         -- internal collated sheets,
                               -- documents: A, A, ..., B, B, ...
```

7.2.5 "impressions-interpreted" (integer(-2:MAX))

The number of impressions interpreted for the job so far.

7.2.6 "impressions-completed-current-copy" (integer(-2:MAX))

The number of impressions completed by the device for the current copy of the current document so far. For printing, the impressions completed includes interpreting, marking, and stacking the output. For other types of job services, the number of impressions completed includes the number of impressions processed.

This value SHALL be reset to 0 for each document in the job and for each document copy.

8 Printer Object Description attributes for Notification

This section specifies the Printer object Description attributes for ~~Job and Printer~~ notification~~Notifications~~.

8.1 "printer-notify" (1setOf collection(1023))

This attribute specifies the outstanding Printer Notification Subscriptions (collections of events, notification-recipients, and other member attributes) for this Printer object, i.e., the subscriptions that clients have supplied in the "printer-notify" Operation attribute of the Subscribe-For-Event-Notifications requests, that have not been removed by subsequent Unsubscribe-For-Event-Notifications requests. The Printer object SHALL support this Printer Description attribute if it supports the Subscribe-For-Event-Notifications operation.

The IPP Printer object SHALL populate the value(s) of this attribute with the collection value supplied by the "printer-notify" Operation attribute in each Subscribe-For-Event-Notification operation. See the description of the "job-notify" Operation attribute for the complete specification of the semantics of this Printer Description attribute.

8.2 Notification Support Printer Description attributes

~~If the Printer object supports the "job-notify" Operation attribute for the Print-Job, Print-URI, and Create-Job operations, then the Printer object SHALL support the following~~

~~supported Printer object Description attributes in the second column in Table 1 that correspond to the "job-notify" member attributes supported.~~

~~If the Printer object supports the Subscribe-For-Event-Notifications operations, then the Printer object SHALL support the following Printer object Description attributes in the third column in Table 1 that correspond to the "printer-notify" member attributes supported.~~

~~Note: These Printer attributes are specified as separate Printer object attributes, rather than as member attributes of a Printer object's collection attribute, since any combination of values may be used for any of the attributes.~~

Table 1 - Notification Support Printer Description Attributes

Collection member attributes in "job-notify" and "printer-notify" Operation attributes	Printer object Notification supported Attributes
notify-event-groups (1setOf type2 keyword)	notify-event-groups-supported (1setOf type2 keyword)
notify-recipients (1setOf uri)	notify-schemes-supported (1setOf uriScheme)
notify-content-type (mimeMediaType)	notify-content-type-supported (1setOf mimeMediaType)
notify-charset (1setOf charset)	notify-charset-supported (1setOf charset)
notify-natural-language (naturalLanguage)	notify-natural-language-supported (1setOf naturalLanguage)
notify-additional-attributes (1setOf keyword)	notify-additional-attributes-supported (1setOf keyword~~s~~)

~~8.1~~8.2.1 Validation of Job Submission ~~Notification~~ Subscriptions ~~Support Printer Description attributes~~

The ~~Job~~ Notification Support Printer object Description attributes (column 2 in Table 1) specify the supported values for the corresponding member attributes of the "job-notify" Operation collection attribute used in the job create operations. The value of the Printer object’s "~~job-~~notify-recipients-supported" attribute is a ’uriScheme’. The Printer object SHALL use the values of this attribute to validate the scheme supplied by the client in the "notify-recipients" member attribute.

For example, if a Printer object supports:

1) ’mailto:’ method for the ’job-completion’ event groups using English, French, U.S. English, and German and supporting additional attributes: "job-uri", "job-name", "job-originating-user-name", "number-of-documents", "job-state", "sides", "finishing"
2) ’sense’ and ’ipp-tcp~~-~~ip-socket’ methods for the ’job-state-changes~~delivery~~’, ’job-progress’, and ’job-completion’ event groups in English only

a system administrator could configure the following Printer Description attributes":

"~~job-~~notify-schemes-supported" = ’mailto’, ’sense’, ’ipp-tcp~~-~~ip-socket’

"~~job-~~notify-event-groups-supported" = ’none’, ’all-job-events’, ’all-printer-events’, ’job-status-change~~delivery~~’, ’job-progress’, ’job-completion’, ’printer-errors’

"notify-content-supported" = ’multi-part/alternative’, ’application/ipp’, ’text/plain’

"~~job-~~notify-natural-language-supported" = ’en’, ’fr’, ’en-us’, ’de’

"~~job-~~notify-additional-attributes-supported" = ’job-uri’, ’job-name’,
’job-originating-user-name’, ’number-of-documents’,
’job-state’, ’sides’, ’finishing’

~~ISSUE 18: Should an administrator be able to configure so that the groups supported is less than all of them. All of them are required for conformance?~~

Note: the fact that not all events are supported for the mailto scheme, or that not all languages are supported for the ’sense’ and ’ipp-tcp~~-~~ip-socket’ methods is not represented, since the collection mechanism is not used to represent the supported attributes. If the client supplies a combination that is not supported, the Printer object SHALL accept the create request (independent of the value of the "ipp-attribute-fidelity" attribute supplied by the client), make suitable substitutions, and return the attributes that are ignored or substituted in the create operation response.

~~ISSUE 19: Are we still ok with not making these "xxx-supported" attributes member attributes of one collection "notifications-supported" Printer Description attribute? Or maybe two collections: "job-notifications-supported" and "printer-notifications-supported" Printer Description attributes?~~

~~8.2~~8.2.2 Validation of Printer ~~Notification~~ Subscriptions ~~Support Printer Description attributes~~

The ~~Printer~~ Notification Support Printer object Description attributes (column 2~~3~~ in Table 1) specify the supported values for the corresponding member attributes of the "printer-notify" Operation collection attribute used in the Subscribe-For-Event-Notifications operation. The value of the Printer object's "~~printer-~~notify-recipients-supported" attribute is a 'uriScheme'. The Printer object SHALL use the values of this attribute to validate the scheme supplied by the client in the "notify-recipients" member attribute. See section 8.2.1 for an example~~, except change all "job-xxx" attributes to "printer-xxx" attributes~~.

9 Notification Content definitions

Just as applications need a defined (extendable) set of notifications, they also need a fixed structure and reliable notification content. The notification content depends on the event. Job events in a Job Submission Subscription via a create operation ONLY apply to the job created. Job events in a Printer Subscription apply to ALL jobs.

An IPP Printer object MAY also implement the "notify-additional-attributes" Operation member attribute in order to allow a client to request additional attributes over and above the fixed set shown in Table 2.

[Some delivery methods, such as SNMP, do not support the requester requesting additional attributes; the notification recipient will have to explicitly use a Get-Job-Attributes or Get-Printer-Attributes operation to get additional attributes about the job or device.]

~~[IPP does not have some of the job progress attributes that the PWG Job Monitoring MIB has. These are indicated with "-" in the IPP attribute column.]~~

~~ISSUE 20: Should we add the job progress attributes to IPP that the PWG Job Monitoring MIB returns in an SNMP trap so that accounting programs can get the same attributes with IPP?~~

9.1 Notification Content attributes

The following sub-sections specify those content attributes that are not Job or Printer attributes:

~~9.1~~9.1.1 "time-at-event" (integer~~-~~(0:MAX)

This notification content attribute indicates the point in time at which the event occurred. In order to populate this attribute, the Printer object uses the value in its "printer-up-time" attribute at the time the job or printer event occurred. This notification content attribute SHALL be part of all notification contents for all events.

NOTE: The "time-at-event" and "printer-up-time" are in units of seconds, not one hundreds of a second (like prtAlertTime and sysUpTime). Thus the attribute name is "time-at-event", rather than "prt-att-18-9-*r*" (where "*r*" is the row in the alert table of this alert), since the value has different semantics.

~~9.2~~9.1.2 "event" (keyword)

This notification content attribute indicates the event (not the event group) that occurred. This notification content attribute SHALL be part of all notification contents for all events, so that a notification recipient can determine which event occurred, even though implementors add their own events and/or other MIBs may use their MIB-specific alert codes in the "alert-code" notification content attribute. For example, for any Printer errors, the value of the "event" notification content attribute SHALL be the ’printer-error’ keyword.

Standard values are defined in section 5.1.2.2.

~~ISSUE 21: Ok, that the "event" attribute always occurs in the notification content, even when there is also the prtAlertCode from the Printer MIB, so that we can add other MIB alerts in the future, too?~~

~~9.2.1~~9.2 *Job event notification content*

Table 2 shows the notification content attributes that SHALL be included in any notification content for a Job event.

Table 2 - Mandatory attributes for notification content depending on the Job event

IPP attribute (content)	**JMP VarBind object/attribute (content)**	**Job Event (not Event Group)**			
		job-received	job-started-processing, job-held, job-released	job-warning, job-error	sheet-completed, collated-copy-completed, job-completed, job-aborted, job-canceled
Common to Job and Printer events:					
printer-uri	hrDeviceIndex	Yes	Yes	Yes	Yes
time-at-event	jmAlertTime (new)	Yes	Yes	Yes	Yes
event	event	Yes	Yes	Yes	Yes
Specific to Job events:					
job-id	jmJobIndex	Yes	Yes	Yes	Yes
number-of-intervening-jobs	jmNumberOfInterveningJobs	Yes	Yes	Yes	-
job-k-octets	jmJobKOctetsPerCopyRequested	-	Yes	Yes	Yes
job-k-octets-processed	jmJobKOctetsProcessed	-	-	Yes	Yes
job-impressions	jmJobImpressionsPerCopyRequested	-	Yes*	Yes*	Yes*
impressions-interpreted-	impressionsInterpreted(112)	-	-	Yes	Yes
job-impressions-completed	jmJobImpressionsCompleted	-	-	Yes	Yes
copies	jobCopiesRequested(90)	-	-	Yes	Yes
impressions-completed-current-copy-	impressionsCompletedCurrentCopy(113)	-	-	Yes	Yes
sheet-completed-copy-	sheetCompletedCopyNumber(95)	-	-	Yes	Yes

IPP attribute (content)	**JMP VarBind object/attribute (content)**	**Job Event (not Event Group)**			
		job-received	job-started-processing, job-held, job-released	job-warning, job-error	sheet-completed, collated-copy-completed, job-completed, job-aborted, job-canceled
number-					
sheet-completed-document-number-	sheetCompletedDocumentNumber(96)	-	-	Yes	Yes
job-collation-type-	jobCollationType(97)	-	-	Yes	Yes
output-bin-	outputBin(54)	-	-	-	Yes**
job-state	jmJobState	-	-	Yes	-
job-state-reasons	jmJobStateReasons1	Yes	Yes	Yes	Yes

’-’ indicates that the attribute SHALL NOT be included in the notification content.

* The IPP Printer object will treat jmJobImpressionsPerCopyRequested in the following manner. If explicitly *passed in on submission*, this will be the value used. If there is no value passed in on submission, then the *implicit value, derived from the final number of impressionsInterpreted for the first copy will be used.*

** **outputBin** may be multi-valued

Note: the ’job-state-changes~~delivery~~’ group has different patterns of attributes sent in the notification content, so that the IPP Printer object would have to subscribe with the SNMP agent using several different SNMP trap OIDs because the VarBind lists must be different.

NOTE: The following objects and attributes have not been included in the fixed set of attributes that SHALL be returned for the indicated reasons (they MAY be requested in implementations that support the "ipp-notify-additional-attributes" attribute):

~~1)"job-state" (JMP jmJobState) - the event indicates the job’s new state.~~

~~ISSUE 22: But "job-state" does appear in the table for certain events?~~
~~ISSUE 23: What about "job-state-reasons"?~~

~~2)~~1) "job-owner" (JMP jobOwner) - the notification recipient should know who the owner is. Also the owner is a string, so it can be long. The total size of the content must fit in the maximum size of a PDU for any transport, which is about 500 octets or so (for IPX).

~~3)~~2) For an IPP device, the jmJobSubmissionID is "job-uri", at least the last 47 octets of it.

~~9.2.2~~9.3 *Printer event notification content*

Table 3 shows the notification content attributes that SHALL be included in any notification content for a Job event. The following sub-sections specify those attributes that are neither Printer attributes not Printer MIB alert objects:

~~9.2.2.1~~9.3.1 "device-name" (name)

This Printer attribute specifies the device name of the device generating the event. This attribute is needed for those IPP Printer objects that support more than one device (so-called fan-out). See [ipp-model]. This attribute is being added as a Printer attribute as well (see [mib-access]).

The other Printer attributes that are contained in a notification-content are the attributes that would be returned in a Get-Printer-Attributes Response, when the "which-device" Operation attributes were supplied with the value equal to that of the "device-name" attribute. For example, the "printer-state" attribute is returned as if the device identified by "device-name" were the only device that the IPP Printer controlled. In other words, the Printer attributes returned in a notification are specialized to the device that generated the event (see [mib-access] for more explanation of this specialization).

~~9.2.2.2~~9.3.2 "which-alert-row" (keyword)

This notification content attribute identifies the row in the Printer MIB alert table. The value is a keyword of the form: "prt-row-18-*r*" where "*r*" is the decimal digits representing the alert row number in the prtAlertTable that was added to generate this alert. The value is a keyword that the client MAY supply directly in a Get-Printer-Attributes operation to get the entire alert group row that causes this alert.

Table 3 - Mandatory attributes for notification content depending on the Printer event

IPP attribute (content)	Printer MIB VarBind object (content)	Printer Event (not Event Group)	
		printer-report, printer-warning, printer-error	printer-accepting-jobs, printer-not-accepting-jobs
Common to Job and Printer events:			
printer-uri (uri)	hrDeviceIndex	Yes	Yes
time-at-event (integer(0:MAX))	prtAlertTime	Yes	Yes
event (enum)	n/a-	Yes	Yes
Specific to Printer events:			
device-name	n/a-	Yes	n/a
which-alert-row (keyword)	prtAlertIndex	Yes	n/a
prt-att-18-2-*r* (enum)	prtAlertSeverityLevel	Yes	n/a
prt-att-18-3-*r* (enum)	prtAlertTrainingLevel	Yes	n/a
prt-att-18-4-*r* (enum)	prtAlertGroup	Yes	n/a
prt-att-18-5-*r* (integer(1:MAX))	prtAlertGroupIndex	Yes	n/a
prt-att-18-6-*r* (integer(-MAX:MAX))	prtAlertLocation	Yes	n/a
prt-att-18-7-*r* (enum)	prtAlertCode	Yes	n/a
prt-att-18-8-*r* (text(255))	prtAlertDescription	Yes	n/a
printer-state (type1 enum)	n/a-	Yes	Yes
printer-state-reasons (1setOf type2 keyword)	n/a-	Yes	Yes

'n/a-' indicates that the table entry is not applicable and ~~attribute~~ SHALL NOT be included in the notification content.

~~ISSUE 24: Ok that I changed the data types that go with prtAlertGroup and prtAlertGroupIndex from keyword back to the ones in the Printer MIB (except time), so that we could use the values returned from the Printer MIB directly.~~

10 Examples

This section provides some complete examples of Job Submission Subscription and Printer Subscription and the resulting event notification.

Notation: A multi-valued attribute is indicated as:

"attribute-name" = ’value1’, ’value2’

The member attributes of a single collection value are indicated inside a pair of {}.

10.1 Example 1: two subscriptions submitted with the job

User Smith submits a print job to printer ’http://cp10.es.ajax.com/wiley’. He subscribes for event notification as part of job submission by supplying the "job-notify" Operation attribute in the Print-Job operation with two collection values: (1) one collection value uses the sockets method so he can fix the problem on the printer and (2) the other collection value uses e-mail to indicate job completion to Jones and White.

```
        "job-notify" =    {     "notify-event-groups" = ’printer-errors’
                                "notify-recipients" =
                    ’ipp-tcp-ip-socket:13.240.120.138/port=6000’ },
                          {     "notify-event-groups" = ’job-completion’
                                "notify-recipients" = ’mailto:Jones@ajax.com’,
                                                  ’mailto:White@ajax.com’ }
```

When the input tray 3 runs out of media, the following notification content is sent to Smith using sockets:

content attribute	Printer MIB source	example value
printer-uri (uri)	hrDeviceIndex	’http://cp10.es.ajax.com/wiley’
time-at-event (integer(0:MAX))	prtAlertTime	no. of seconds since power up that ran out of media
event (enum)	-	’printer-error~~problem~~’
device-name	-	’wiley’
which-alert-row (keyword)	prtAlertIndex	’prt-row-18-235’
prt-att-18-2-*r* (enum)	prtAlertSeverityLevel	criticalBinaryChangeEvent(3)
prt-att-18-3-*r* (enum)	prtAlertTrainingLevel	untrained(3)
prt-att-18-4-*r* (enum)	prtAlertGroup	8
prt-att-18-5-*r* (integer(1:MAX)	prtAlertGroupIndex	3
prt-att-18-6-*r* (integer(-MAX:MAX))	prtAlertLocation	1
prt-att-18-7-*r* (enum)	prtAlertCode	subunitEmpty(13)
prt-att-18-8-*r* (text(255))	prtAlertDescription	"Input tray 3 empty"
printer-state (type1 enum)	-	stopped(5)
printer-state-reasons (1setOf type2 keyword)	-	’media-empty’

When the job completes, the following notification content is sent to both Jones and White using e-mail:

content attribute	Job MIB source	example value
printer-uri	hrDeviceIndex	’http://cp10.es.ajax.com’

content attribute	Job MIB source	example value
time-at-event	jmAlertTime (new)	no. of seconds since power up that job completed
event	event	’job-completed’
job-id	jmJobIndex	1000
number-of-intervening-jobs	jmNumberOfIntervening Jobs	-
job-k-octets	jmJobKOctetsPerCopyRequested	23000
job-k-octets-processed	jmJobKOctetsProcessed	23000
job-impressions	jmJobImpressionsPerCopyRequested	10
impressions-interpreted	impressionsInterpreted(112)	10
job-impressions-completed	jmJobImpressionsCompleted	20
copies	jobCopiesRequested(90)	2
impressions-competed-current-copy	impressionsCompletedCurrentCopy(113)	10
sheet-completed-copy-number	sheetCompletedCopyNumber(95)	2
sheet-completed-document-number	sheetCompletedDocumentNumber(96)	1
job-collation-type	jobCollationType(97)	collatedDocuments(4)
output-bin	outputBin(54)	2, 3
job-state	jmJobState	’completed’
job-state-reasons	jmJobStateReasons1	’job-completed-successfully’

10.2 Example 2: Add a Printer monitoring application

This example is the same as example 1, but with the addition of a printer monitoring application being run by the system operator. This monitoring application subscribes for event notification to the IPP Printer and does not submit a job. The application is not displaying the job queue and is only interested in problems with the Printer.

The application subscribes to IPP Printer ’http://cp10.es.ajax.com/wiley’ for event notification using the Subscribe~~pt~~-For-Event-Notifications operation. The events are to be delivered to the application using the sockets method. The "printer-notify" Operation attribute in the Subscribe~~pt~~-For-Event-Notifications operation as follows:

```
        "printer-notify" =     { "notify-event-groups" = ’printer-errors’
                                 "notify-recipients" =
                              ’ipp-tcp-ip-socket:13.240.120.138/port=6000’ }
```

When the input tray 3 runs out of media, the same notification as in example 1 is sent to Smith using sockets. In addition, the identical notification content is sent to the monitoring application.

10.3 Example 3: Add a Job queue monitoring application

This example is the same as example 2, but with the addition of a job queue monitoring application being run by the system operator. This monitoring application subscribes for event notification to the IPP Printer and does not submit a job. The application is displaying the job queue and is not interested in problems with the Printer, but is interested in job state changes and the progress of each job at the sheet and collated document copy level.

The application subscribes to IPP Printer ’http://cp10.es.ajax.com/wiley’ for event notification using the Subscribe~~pt~~-For-Event-Notifications operation. The events are to be delivered to the application using the ’ipp-tcpip-socket’ method. The "printer-notify" Operation attribute in the Subscribe~~pt~~-For-Event-Notification operation as follows:

```
      "printer-notify" =    { "notify-event-groups" = ’job-~~delivery’~~state-changes,
                              ’job-progress’, ’job-completion”
                              "notify-recipients" =
                          ’ipp-tcp-ip-socket:13.240.120.138/port=6000’ }
```

When the input tray 3 runs out of media, the same notification as in example 1 is sent to Smith using sockets. In addition, the identical notification content is sent to the monitoring application as in example 2, but the queue monitoring application does not receive an event notification. Suppose that there were 3 jobs ahead of Smith’s job.

~~ISSUE: I think that such a queue monitoring program wants job state changes, but NOT ’sheet-completed’, and ’collated-copy-completed’. Perhaps, we should move ’job-held’ and ’job-released’ events to the ’job-delivery’ group? Maybe rename the ’job-delivery’ group to something like: ’job-state-change’ group?~~

Instead, the queue monitoring application receives notifications of the following form, when ~~the~~ each job is received, when ~~the~~ each job starts processing, when each job stacks a sheet or collated document copy, and when ~~the~~ each job completes:

content attribute	Job MIB source	example value
printer-uri	hrDeviceIndex	’http://cp10.es.ajax.com’
time-at-event	jmAlertTime (new)	no. of seconds since power up that job was received, started processing, or completed
event	jmE~~e~~vent (new)	’job-received’, ’job-held’, ’job-released’, ’job-started-processing’, ~~or~~ ’sheet-

content attribute	Job MIB source	example value
		completed’, ’collated-copy-completed’, ’job-completed’, ’job-aborted’, or ’job-canceled’
job-id	jmJobIndex	1000
number-of-intervening-jobs	jmNumberOfInterveningJobs	3, 0, or -
job-k-octets	jmJobKOctetsPerCopyRequested	23000
job-k-octets-processed	jmJobKOctetsProcessed	23000
job-impressions	jmJobImpressionsPerCopyRequested	10
-	impressionsInterpreted	10
job-impressions-completed	jmJobImpressionsCompleted	20
copies	jobCopiesRequested	2
-	impressionsCompletedCurrentCopy	10
-	sheetCompletedCopyNumber	2
-	sheetCompletedDocumentNumber	1
-	jobCollationType	collatedDocuments(4)
-	outputBin	2, 3
job-state	jmJobState	’pending’, ’processing’, or ’completed’
job-state-reasons	jmJobStateReasons1	nothing, nothing, or ’job-completed-successfully’

~~11Encoding~~

~~The new ’collection’ attribute syntax will use the 0x34 tag value that has been reserved in the IPP/1.0: Protocol Specification for this purpose.~~

11 References

[cohen]

J. Cohen, et al, General Event Notification Architecture Base. April 23, 1998, work-in-progress, <draft-cohen-gena-p-base-00.txt>.

[draft-prtmib]

R. Turner, Printer MIB, work-in-progress, submitted to IESG, <draft-ietf-printmib-mib-info-03.txt>, October 1997. Will supersede RFC 1759.

[ipp-model]
Isaacson, S, deBry, R, Hastings, T, Herriot, R, Powell, P, “Internet Printing Protocol/1.0: Model and Semantics”.

[ipp-pro]
Herriot, R., Butler, S., Moore, P., Tuner, R., " Internet Printing Protocol/1.0: Protocol Specifications", draft-ietf-ipp-pro-05.txt, January 9, 1998.

[ISO10646-1]
ISO/IEC 10646-1:1993, "Information technology -- Universal Multiple-Octet Coded Character Set (UCS) - Part 1: Architecture and Basic Multilingual Plane, JTC1/SC2."

[ISO8859-1]
ISO/IEC 8859-1:1987, "Information technology -- 8-bit One-Byte Coded Character Set - Part 1: Latin Alphabet Nr 1", 1987, JTC1/SC2.

[jmp-mib]
R. Bergman, T. Hastings, S. Isaacson, H. Lewis, PWG Standard Job Monitoring MIB - V1, February 3, 1998, work-in-progress, submitted to the IESG to be published as an informational RFC, <draft-ietf-printmib-job-monitor-07.txt>

[mib-access]
S. Isaacson, T. Hastings, R. Herriot, K. Schoff, IPP Device and MIB access, Version 0.03, May 5, 1998, work-in-progress, <ipp-mib-access-980505.pdf>.

[req]
R. deBry, Requirements for IPP Notifications, March 11, 1998, work-in-progress, <draft-ietf-ipp-not-01.txt>.

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

[RFC-2044]
F. Yergeau, "UTF-8, a transformation format of Unicode and ISO 10646", RFC 2044, October 1996.

[RFC-2046]
Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types. N. Freed & N. Borenstein. November 1996. (Obsoletes RFC1521, RFC1522, RFC1590), RFC 2046.

[url-reg]
R. Petke, Registration Procedures for URL Scheme Names, April 30, 1998, work-in-progress, <draft-ietf-urlreg-procedures-01.txt>

[US-ASCII]
Coded Character Set - 7-bit American Standard Code for Information Interchange (ASCII), ANSI X3.4-1986. This standard is the specification of the US-ASCII charset.

12 Copyright Notice

None,

13 Author's Address

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

Phone: 310-333-6413
Fax: 310-333-5514
e-mail: hastings@cp10.es.xerox.com

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

Phone: 801-861-7366
Fax: 801-861-2517
e-mail: sisaacson@novell.com

Harry Lewis
IBM Corporation
6300 Diagonal Hwy
Boulder, CO 80301

Phone: (303) 924-5337
Fax: (303) 924-4662
Email: harryl@us.ibm.com

14 Appendix - Specification for the IPP collection attribute syntax

This appendix is the complete specification for the new 'collection' attribute syntax that the notification specification uses. Other future extensions, both registered and private, will make use of this new attribute syntax.

This mechanism had originally been named 'dictionary', but we agreed to change it since the member attributes are not ordered, typically.

~~There are two issues highlighted in yellow.~~

14.1 Problem Statement

There is no good way to add attributes that contain several fields, whether the fields are mandatory or optional. Instead of each new attribute that needs more than one field (struct), requiring an ad hoc attribute syntax, such as we have done for the 'resolution' attribute syntax for use in the "printer-resolution" attribute, it would be desirable to have a simple, general mechanism for representing multi-field values. It would also be desirable to allow fields to be omitted, when the attribute specification allows that. This mechanism would be useful for both new attributes that we might register as extensions to be used with the IPP standard, or that implementers might implement as private extensions.

14.2 Summary of the attribute syntax alternative

A number of alternatives were considered. See the last section for a list and the reasons for their rejection.

The proposal is to add a new attribute syntax, called 'collection'. Any attribute of type 'collection' shall have a value that is a set of unordered attributes, where each attribute MAY be single-valued or multi-valued as specified for the collection attribute. Since the attribute value has a length, like any other attribute value, IPP objects not supporting the attribute can easily skip over the entire attribute value, i.e., skip over the entire set of attributes that make up the collection value.

14.3 Requirements for and properties of the suggested collection mechanism

The collection mechanism for use with IPP needs to have the following semantic properties:

1. The collection mechanism provides a way to supply and query a set of attributes as a logical unit. Then each 'field' that is present in the collection would be self-identifying by its attribute name.

2. The attributes in a collection are unordered. Therefore, an IPP object MUST be able to accept attributes in a collection in any order.

3. The semantics of a collection attribute specifies which attributes in a collection instance are MANDATORY for the IPP object to support and which are OPTIONAL for the IPP object to support when the IPP object supports that collection attribute.

4. The semantics of a collection attribute specifies which attributes in a collection instance are required for the requester to supply and which the requester may omit.

5. A collection attribute could be single valued, i.e., with one collection value consisting of a set of attributes, or could be multi-valued, i.e., with multiple collection values, each consisting of a set of attributes.

6. An attribute in a collection value can be single valued or multi-valued as well according to the specification of the collection attribute.

7. As with all attribute values, if an IPP object does not support a collection attribute, it must be easy for the IPP object to ignore each collection attribute value.

8. The syntax of each collection value is the same as a group of attributes in a request or response, so each attribute in a collection value instance has its keyword name, its attribute syntax code, and its value.

9. An implementer MAY support additional registered or private attributes in a collection. In other words, a collection is extensible, just like an attribute group in an operation or response.

10. Since support of all possible combinations of values for all attributes in a collection value may not be supported by some implementations, there should be a way for the IPP object to indicate which combinations of values are supported. For example, 300x300, 600x300, and 600x600, but not 300x600 dpi.

11. Finally, an attribute in a collection value can be itself a collection, so that nesting could be allowed, if the specification of a collection attribute allowed a collection attribute to be contained in its collection.

14.4 Examples of collection usage

This section describes four collection Job Template examples: "printer-resolution", "job-notify", "job-start-page-contents", and "postal-mail-disposition" attributes. The "printer-resolution" attribute only contains single-valued attributes, while the "printer-resolution-supported" and "job-notify" attribute contains multi-valued collection attributes, i.e., contain more than one collection as a value of an attribute.

14.4.1Example a: "printer-resolution" Job Template attribute

For example, the new "printer-resolution" attribute was defined using a very ad hoc 'resolution' attribute syntax. Had we had the collection attribute syntax, we might have chosen to use it here, though we wouldn't have had to either. If we did use the 'collection' attribute syntax for the "resolution", the attribute value would contain the following attributes: "resolution", "cross-feed-resolution", and "resolution-units". We could have also specified that the "cross-feed-resolution" attribute is OPTIONAL and when omitted, the cross-feed resolution is the same as the "resolution" attribute, since most resolutions are the same in both directions. We could have also specified that the "resolution-units" attribute is OPTIONAL and when omitted, the resolution units are dots per inch.

~~ISSUE 25: Should we also allow the member attributes of a collection to be supplied by themselves when the client does not want to group them or is that just an unnecessary alternative form?~~

The specification for the "printer-resolution" collection attribute is that its collection value is made up of the following attributes:

Attribute name	syntax	in request
"resolution"	integer	required
"cross-feed-resolution"	integer	optional
"resolution-units"	enum	optional

For a simplified collection attribute notation, lets use:

"*collection attribute*" = { *set of attributes and values* }

where a set of {} is used to group a single collection value.

For example, a client supplying a resolution of 600 x 300 would be indicated in examples using the following notation:

"printer-resolution" = { "resolution" = ’600’, "cross-feed-resolution" = ’300’ }

14.4.1.1 "printer-resolution-default" example

The Printer object could represent the "printer-resolution-default" default values as a single collection value. For example, a system administrator (or the printer vendor) could specify the default as:

"printer-resolution-default" = { "resolution" = ’300’ }

14.4.1.2 "printer-resolution-supported" example and validation of collections

The Printer object could indicate the combinations of resolutions that are supported by three sets of collection values which represent 300x300, 600x300, and 600x600 dpi, respectively (300x600, say, is not supported). Such a configured situation could be represented in examples as:

```
"printer-resolution-supported" = {
                        { "resolution" = ’300’ },
                        { "resolution" = ’600’, "cross-feed-resolution" = ’300’ },
                        { "resolution" = ’600’ } }
```

14.4.2Example b: "job-notify" Operation attribute

In order to meet the IPP notification requirements, the requester must be able to supply one or more notification profile values, where each profile value consists of a set of "job-notify-events", one "job-notify-method", multiple "job-notify-recipients", one "job-notify-natural-language", one "job-notify-charset", and possibly multiple "job-notify-additional-requested-attributes". There might be a similar multi-valued "printer-notify" Printer object collection attribute that is set by means outside of the IPP/1.0 protocol, but is independent of jobs, so that they would specify notification to operators. Both the "job-notify" and the "printer-notify" collection attributes are MULTI-VALUED and contain attributes that themselves are MULTI-VALUED.

The "job-notify" Operation collection attribute would have collection values with the following syntax:

Attribute name	syntax	in request
"notify-event-groups"	1setOf enum	optional
"notify-recipients"	1setOf uri	required

A Print-Job request could supply the collection attribute values in order to send immediate ~~’job-aborted’ and ’job-canceled’~~ ’job-error’ events to Smith (himself) and e-mail ’job-completion’ to Jones and White. A notation for this example could be to use a set of {} to indicate each collection value

```
        "job-notify" = ~~{~~   {     "notify-event-groups" = ’job-errors’
                                     "notify-recipients" =
                                     ’ipp-tcpip-socket:13.240.120.138/port=6000~~Smith~~’ },
                               {     "notify-event-groups" = ’job-completion’
                                     "notify-recipients" = ’mailto:Jones’, ’mailto:White’ }~~}~~
```

14.4.3Example c: Start page fields supplied by the end-user

As a third example of a collection, an attribute could represent the fields that the submitter wishes to be printed on the job-start page. The name of the attribute might be: "job-start-page-contents". The collection value might include: "job-name", "user-name", "job-comment", "account-name", "job-disposition", "job-delivery", etc. where the values of the attributes in the collection are printed after each attribute name on the job-start-page.

Attribute name	syntax	in request
"job-name"	name	required
"user-name"	name	required
"job-comment"	text	optional
"account-name"	name	optional
"job-disposition"	keyword	optional
"job-delivery"	1setOf keyword	optional

14.4.4Example d: Postal mailing address

As a final example of a collection, an attribute could represent a postal mailing address for the output. The name of the attribute might be "postal-mail-disposition" and it would be multi-valued, i.e., 1setOf collection. The collection attribute might have the following specification and support requirements if the "postal-mail-disposition" collection attribute is supported at all:

Attribute name	syntax	in request	IPP object support
"addressee-name"	text	required	MANDATORY
"company-name"	text	optional	OPTIONAL

"internal-mail-stop"	text	optional	OPTIONAL
"apartment-number	text	optional	MANDATORY
"street-address"	text	required	MANDATORY
"city-or-town	text	required	MANDATORY
"state"	text	required	MANDATORY
"postal-zone	text	required	MANDATORY
"country"	text	optional	OPTIONAL
"phone-numbers	1setOf text	optional	OPTIONAL

14.5 Detailed description 'collection' attribute syntax

Register the following attribute syntax, written in the style of section 4.1 Attribute Syntaxes of the IPP Model specification:

4.1.n 'collection'

A set of unordered attributes, where each attribute MAY be single-valued or multi-valued as specified for the collection attribute. The maximum length of a collection value is specified enclosed in parentheses in the sub-section header of the specification of the attribute.

As in the attribute sets that are passed in an operation groups~~s~~, an IPP object SHALL accept the attributes in a collection value in any order. The specification of an attribute whose attribute syntax is 'collection' MAY specify one or more attributes that SHALL be first in each collection value, in order to simplify processing, just as in the Operation attributes. If an attribute that is specified to be first is not in its required position, the IPP object SHALL reject the operation and return the 'client-error - bad syntax' error status code.

~~and n~~No attribute SHALL occur more than once in a collection value. As in operation requests~~However~~, if the same attribute does occur more than once in a collection by error, the IPP object SHALL reject the operation and SHALL return the 'client-error - bad syntax' error status code.

The specification of the attribute that uses the 'collection' attribute syntax SHALL specify:

1. as with any attribute, whether the attribute is single-valued (attribute syntax = 'collection') or multi-valued (attribute-syntax = '1setOf collection').

2. For each attribute in the collection value, whether the IPP object MUST implement the attribute (MANDATORY) or MAY implement the attribute (OPTIONAL).

3. for each attribute in the collection value, whether the attribute's presence is required or optional.

4. for each attribute permitted in the collection value, the completed specification of that attribute shall be included or inferred by reference to the specification of that attribute elsewhere, including its keyword name, its attribute syntax, including '1setOf, if it is multi-valued, and the semantics of the values.

5. for each attribute defined in the collection, whether that attribute may also be used separately by itself. For example, in the "job-notify" example, could the "job-notify-events" and "job-notify-recipients" attributes occur by themselves in a create operation, say, when the client is only specifying a single collection or must they always occur within a collection value.

6. for each attribute defined in the collection, whether that attribute MAY occur anywhere in the collection value (the default case) or MUST be first or after some other attribute that MUST be first (must be explicitly specified).

A collection may contain another collection, i.e., may include an attribute whose attribute syntax is, itself, a 'collection', if the specification of the (outer) collection attribute allows.

Additional attributes may be registered for use in a collection attribute.

Implementers may support additional private attributes in a collection value.

~~ISSUE 26: What should the maximum size of a collection value be? If it is much bigger than the current maximum of 1023 octets, it may not be safely ignored by existing parsers. Is 2047 octets sufficiently big, without being a problem to existing parsers?~~

14.6 Encoding

This section shows the encoding for the alternative of representing a collection as a new attribute syntax. The new 'collection' attribute syntax will use the 0x34 tag value that has been reserved in the IPP/1.0: Protocol Specification [ipp-pro] for this purpose.

The following example is written in the style of the IPP/1.0 "Encoding and Transport" (nee "Protocol") document [ipp-pro].

Octets	**Symbolic Value**	**Protocol field**	comments
0x34	collection type	value-tag	"job-notify" attribute
0x000a		name-length	
job-notify	job-notify	name	
0x0064~~2~~		value-length	100~~98~~ octets in 1st dict value
0x45	uri type	value-tag	"~~job-~~notify-recipients" attribute
0x0011		name-length	
notify-recipients	notify-recipients	name	
0x0019~~20~~		value-length	
ipp-tcp-ip-socket:port=700	ipp-tcp-ip-socket:port=700	value	
0x44	keyword type	value-tag	"~~job-~~notify-event-groups" attribute
0x0013		name-length	
notify-event-groups	notify-event-groups	name	

Octets	**Symbolic Value**	**Protocol field**	comments
0x0b		value-length	
job-errors	job-errors group	value	
0x44	keyword type	value-tag	start of 2nd job-notify-event-groups value
0x0000		name-length	0 length means next multiple value
0x000e		value-length	
job-completion	job-completion	value	
0x34	collection-type	value-tag	start of 2nd collection value
0x0000		name-length	0 length mean next multiple value
0xnnnn	0xnnnn	value-length	nnnn octets in 2nd dict value
0x45	uri type	value-tag	"~~job-~~notify-recipients" attribute
0x0015		name-length	
job-notify-recipients	job-notify-recipients	name	
0x000c		value-length	
mailto:smith	mailto:smith	value	
...			nnnn octets of the next dict value

~~1.7~~14.7 *Rejected alternatives for a collection mechanism*

This section lists the alternatives we considered for adding a new attribute syntax to represent a collection value.

1. No maximum length for the new attribute syntax: ’collection’. If an IPP object supports collection it has to read a piece at a time. If it doesn’t it has to be able to ignore an arbitrarily long data value. See the encoding example in the next section.

Reason for rejection: Not completely compatible with current parsers that have a fixed butter size for entities of around 1023 octets, the current IPP data type maximum.

2. Have a 2047 octet max length, continueCollection as a second attribute syntax and endCollection so that dictionaries can nest.

Reason for rejection: More complexity.

3. Have a 2047 octet max length but allow repeated instances of an attribute to append additional collection values.

Reason for rejection: Not the current procedure for duplicate attributes; the IPP Object is to return an error.

4. Add a new group tag to represent a collection value somehow. Groups do NOT have lengths and existing parsers are supposed to ignore group tags they don't understand.

Reason for rejection: Not completely compatible with existing parsers.

5. Add an out-of-band value that indicates that this attribute was the beginning of a collection and add an attribute that marked the end of the collection value.

Reason for rejection: Not completely compatible with existing parsers. Existing parser would try to interpret the contents of the collection as regular attributes.

6. Extend the attribute naming mechanism to include a collection name and a collection index for use with multi-valued dictionaries. Use the colon (":") to separate component names. Thus if foo is a set of dictionaries, then "foo:1:x" is the name that accesses field x of the 2nd collection of attribute foo (indexing is 0 based). Leaving off the syntax after either colon, is interpreted as a wild card meaning all values with the prefix up to the colon.

Reason for rejection: Changing the naming more of a change than is necessary with the current 1setOf 1setOf proposal, which does not change the naming and does not add an attribute syntax.

7. Add a numeric instance number to the end of parallel attributes, i.e., "~~job-~~notify-method-supported-1".

Reason for rejection: Not needed to be able to address a particular instance of a parallel attribute value.

8. Use the semantics of parallel multi-valued attributes that we have in IPP/1.0, such as we already have for the "printer-uri-supported" and "uri-security-supported" Printer attributes, in order to achieve the effect of multi-valued dictionaries containing single values attributes. In order to represent the effect of a collection which contains attributes that are multi-valued, we only need to introduce the model semantics of: 1setOf 1setOf X as an attribute syntax.

Reason for rejection: Implementation with DPA parallel attributes has shown that it is too difficult for clients and servers to deal with parallel values. Its much better if the values in a collection value are all bound together. Also what if the number of values isn't the same?

9. Calling the new data type a 'dictionary'. Instead, we chose 'collection', since the name dictionary implies some sort of sorting or ordering.