Basic IPP Notifications using HTTP

Combining "Server Directed" Client Pull and Server Push with MIME Multipart/Mixed

Harry Lewis – IBM Printing Systems

Scope

Print Job notifications that leverages

- The IPP HTTP connection
- Tracking to completion
- The possibility of reliable page progress
- The possibility of indirection in the monitoring process
- Server directed polling throughout the spooling phase
- A persistent connection for detailed, asynchronous notification during printing

Abstract

The initial IPP standard and specifications convey print job and device status only via polling. If the IPP client submits a print job and would like to determine the status of that job or the device to which the job was sent, the client must access the job or printer URL. Typically, the client has little information on which to base it's polling interval and is unaware of any redirection that may occur in the job submission path.

The IPP working group of the PWG and IETF is developing schemes for notification by various means and with selectable content. These may include e-mail, raw TCP, Instant Messaging, Pager, SNMP and UDP forms. Both "per job" and "per printer" subscriptions are under consideration. A client may subscribe on behalf of a 2nd or 3rd party recipient as well as designate desired events for notification (job-end and jam but not page

progress, for example). There are also ways of managing and administering subscriptions. Certainly, one approach to satisfying the notification requirements for IPP is to extend the architecture to fully address a broad range of possibilities. The current IPP notification work expresses all these goals.

This paper addresses a specific goal for IPP - a simple, reliable, base form of "IPP notification" that leverages the HTTP connection established during printing. This goal is embodied in the "Qualdocs" (FAX over IPP) requirement for real-time delivery assurance in the form of end-of-job notification. The proposal further provides the possibility of allowing page-progress notification across the Internet.

This paper outlines a "per-job" (only) notification scheme which is tied to the upstream process rather than to any particular end-user or client. Notifications may relate to both job and device events and payload will be relatively fixed. The notification scheme is a sequence of "Server Directed" Polling with the option for a period of connection based Asynchronous Notification that exists during actual page rendering and completion. Both polling and asynchronous notifications rely on the HTTP response from the IPP server to direct the upstream entity. "Server Directed Polling" directs the upstream client about **when** (seconds to delay) and **where** (URL) to poll next. Connection based asynchronous notification is achieved when the IPP printer holds open the HTTP connection for the duration of the print job and formats the response using MIME Multipart/Mixed to relay page and copy progress, device events and overall completion status, as it occurs, to the upstream entity.

Comparison

This proposal is intended to accommodate a basic form of IPP notification. The thesis of this proposal is to recommend the more challenging notification requirements be satisfied outside the context of IPP. *For example, e-mail notification of an alternate recipient may be handled by client software rather than placing this burden on the IPP printer. A network notification server may provide a wide variety of alternative methods such as pager or instant messaging*. Here is an attempt to compare this proposal with others currently on the scene.

- Current proposals allow extreme flexibility in terms of transport, subscription type and content. This may actually reduce acceptance and interoperability.

- Some current proposals require an HTTP server at the client in order to receive notifications. This proposal makes use of the existing IPP infrastructure.
- There is no need to "administer" (i.e. Query subscriptions and their attributes) this basic form of notification.
- There are no special security requirements for the proposed notification scheme. The security context of the submission is inherent in the notification scheme.
- This proposal makes use of MIME Multipart/Mixed which may not be supported by some off the shelf web servers and clients. Other IPP requirements (such as HTTP1.1 with chunking and persistent connections) have already introduced similar constraints which have resulted in the adaptation of specialized IPP HTTP servers and clients.

Table of Contents

Server Directed Polling

The IPP model allows the end user to query the status of the Printer by using the Get-Printer-Attributes operation and follow print job progress with Get-Jobs, and Get-Job-Attributes operations. Polling for status can be "expensive" in terms of the overhead required to open and close connections. Polling intervals (default and administratively set) are arbitrary, at best. Polling for job notification should be guided by something more intelligence than a fixed polling rate.

Each IPP response contains not only a "pass/fail" status code but may also contain operation attributes, object attributes, and/or status messages generated during the operation. "Server Directed Polling", as proposed here, introduces two new attributes for the Get-Job-Attributes operation.

1. "job-polling-delay" (integer(0:MAX))

The job-polling-delay attribute, if present in the Get-Job-Attributes response, provides a "hint" to the upstream entity regarding it's polling interval. The value is the recommend time to wait (in seconds) before requesting the next Get-Job-Attributes operation. The heuristics associated with determining the value for this delay is implementation dependent. The IPP client may ignore this attribute.

Values for job-polling-delay are in seconds. A value of 0 means the client should issue the next Get-Job-Attributes operation as soon as possible.

2. "next-polling-url" (uri)

The next-polling-url attribute, if present in the Get-Job-Attributes response, recommends the target URL for the next Get-Job-Attributes operation pertaining to this print job. This attribute acknowledges latency and indirection in the print submission pipeline, and the possibility of cascaded IPP client/servers. The attribute supplies a URL where the next Get-Job-Attributes operation for this job should be directed. An example of indirection might be a notification service that uses a streamlined notification protocol with the device but handles a variety of outgoing notification schemes to a large number of recipients.

Asynchronous Notification

Notification of print job status and device events, as they occur, can be achieved without polling by maintaining persistence on the HTTP connection established during submission of the IPP print job. Using this connection for asynchronous notification will be more efficient than polling because it does not repeatedly make and break the connection. While efficient in terms of transaction overhead, this notification scheme is resource intensive. IPP printers may support a limited set of HTTP connections. Also, if the job is put on hold or resides in a long print queue, the connection will be open for a long period of time. Therefore, this form of notification is appropriate when the job is at or near the actual processing stage where pages are rendered and stacked.

The persistent connection will accommodate reliable, asynchronous notifications by formatting the response as MIME multipart/mixed. RFC2046 ((MIME) Part Two: Media Types) section 5.1.3. (Mixed Subtype) describes the "mixed" subtype of "multipart" as "intended for use when the body parts are independent and need to be bundled in a particular order". In our case, the body parts are in response printing via IPP and the order is dictated by the actual occurrence of events.

Details pertaining to the operation of HTTP1.1, persistent connections and MIME Multipart/Mixed are (currently) supplied by the following references.

1. HTTP1.1 ftp://ftp.isi.edu/in-notes/rfc2616.txt
2. MIME ftp://ftp.isi.edu/in-notes/rfc2046.txt

Notification Payload

The notification payload is intended to be relatively fixed depending on the range or proximity to completeness of the print job. Device events may be contained in the payload whenever they occur and no matter which job is currently printing. Device events are defined by the Printer MIB alert table definition.
ftp://ftp.pwg.org/pub/pwg/pmp/drafts/

Notification payload as it relates to the print job will carry basic job identification and characterization information as appropriate for the job's current position in the print queue. For example, there is no requirement to provide page progress information while the print job is in the queue. Likewise, there is no reason to indicate queue position while the job is actively printing. Information that will enable the proper construction of a job progress "gas gauge" can be provided whenever it is accurately known by the printer and may be updated if new information is determined.

(*Obviously, this section needs more definition. I decided it was best to get the overall proposal out for review with the details of this section mostly TBD*.)

Summary

Combine two types of HTTP transaction, each appropriate to specific parts of the print operation. One type of HTTP print job notification is called "Server Directed Client Polling". This type of notification will track the job as it traverses the print "spool" process. The client "polls" for status but at a rate and location indicated in the HTTP response header. The other type of notification is "Server Push" which makes use of the MIME type "Multipart/Mixed" and a persistent connection to allow the server to send unsolicited notifications a piece at a time until the job is complete.