Subj: Comparison of Notification Delivery Method alternatives
From: Tom Hastings
Date: 10/27/99
File: IPP-notifiication-delivery-method-comparison.doc

This paper is an attempt to capture the list of various proposals for notification delivery methods (schemes) that have appeared in contributions and/or were mentioned in the brainstorming telecon on Wednesday, 10/20/99. The intent is to use this as starting point at the IPP WG meeting, 10/27/99. One of the principles of brainstorming is that we list all suggested alternatives. If someone wants to change one of the proposals, we add the changed proposal as a new proposal. After we complete the list we go through them all listing the PROs and CONs of each. Often when such a process is finished, the winner emerges clearly.

1. **Server-directed Client Polling** - The client requests notification in the Job Creation request as in [ipp-not] using "notify-events" and "notify-recipient" (scheme with no recipient address). The server returns a new operation attribute in the Job Creation response and the Get-Job-Attributes response to the client telling the client when next to poll using Get-Job-Attributes on a new channel and at which URL. May use the Subscription object. See Harry Lewis's proposal: **Simple_IPP_Notification.pdf**.

   PRO:
   a. If you can get to the Printer, e.g., through the firewall, then you can get notification.
   b. Don't have to keep open a HTTP connection for every pending job.

   CON:
   a. Only works for Job Creation operations.
   b. Printer can't reliably tell when an event will occur, i.e., when job will start or when job will complete.
   c. Won't work for unanticipated events like jams.
   d. More network traffic polling than with other methods.

2. **Server push using MIME-multi-part responses** - The client requests notification as in Job Creation request as in [ipp-not] using "notify-events" and "notify-recipient". The server returns multiple responses to the Job Creation request, each as separate multiple MIME-multi-part responses on the same open channel, one for each requested event as it occurs. In other words, one Job Creation request has multiple responses over time. See Harry Lewis's proposal: **Simple_IPP_Notification.pdf**.

   PRO:
   a. If you can get to the Printer, e.g., through the firewall, then you can get notification.
   b. Only needs one connection for a print job operation; don't have to open a

new channel.

CON:
a. Need to keep an HTTP connection open for every pending job that wants events.
b. Only works for Job Creation operations for Per-Job subscriptions. Doesn't work for Per-Printer subscriptions.
c. Can multiple-responses get through CGI and other web servers?

3. **In-band Get-Notifications operation for any of owner's subscriptions** - The client requests notification in the Job Creation request, Create-Printer-Subscriptions, and Create-Job-Subscriptions as in [ipp-not] using "notify-events" and "notify-recipient". Any time thereafter, the client requests notifications using a new Get-Notifications operation to return all subscriptions for which the client was the owner. The notification returned are for any subsequent events that occur using the current "notify-events" operation attribute. The server keeps the channel open and returns multiple MIME-multi-part responses on the same open channel, one for each requested event as it occurs. In other words, one Get-Notifications request has multiple responses over an indefinite period of time. The client can perform Get-Notifications before any other requests.

   PRO:
   a. Only keep one HTTP channel open for ALL notifications for one Notification Recipient.
   b. Don't miss events, if the client does the Get-Notifications before submitting jobs.
   c. Works for Per-Job and Per-Printer subscriptions.

   CON:
   a. Get all notifications delivered to one place.
   b. Can multiple-responses get through CGI and other web servers
   c. Forces server to relay inband notification.

4. **In-band Get-Notifications operation with Subscription Ids -** Same as 3, in addition the Get-Notifications operations specifies the subscription Ids for which events are wanted.

   PRO:
   a. Only keep one HTTP channel open for ALL notifications for one Notification Recipient.
   b. Works for Per-Job and Per-Printer subscriptions.
   c. Same owner from different workstations can get notifications from different subscriptions.

   CON:
   a. May miss events between the Job Creation and the Get-Notification

request.
b. Can multiple-responses get through CGI and other web servers?
c. Notification Recipient has to keep track of the subscription ids in order to request.

5. **In-band Get-Notifications operation with choice of owner or Subscription Ids -** combination of 3 and 4. Client chooses which in Get-Notifications request.

   PRO:
   a. Same as 3.
   d. Client has choice to get all its subscriptions or selective subscriptions.

   CON:
   a. Can multiple-responses get through CGI and other web servers
   b. Forces server to relay inband notification.

6. **In-band Get-Notifications without Subscription objects** - Same as 3, except that the client supplies the "job-id" and "notify-events" operation attribute in the Get-Notifications operation, instead of the Job Creation request. This is one of the few in which there isn't a Subscription object.

   PRO:
   a. Keeps the HTTP channel open for notifications for one job.
   b. Subscription object is temporary for the life of the Get-Notification operation and its responses.
   c. Clear up when client disconnects.

   CON:
   a. Same as Create-Job-Subscription in [ipp-not]. except that Create-Job is optional.
   b. May miss events between the Job Creation and the Get-Notification request.
   c. One channel open for each active job.

7. **In-band Get-Notifications with queued events** - Same as 3, except that events are queued from the time of the creation of the subscription in the Job Creation request. Therefore, the Get-Notifications gets each event whether it happened before or after the Get-Notifications request.

   PRO:
   a. Same as 3.

   CON:
   a. Queuing is harder to implement, especially for low end implementations.
   b. Queue can build up if notifications not picked up.

8. **In-band Get-Notification with queued events but only one event per Get-Notification request** - Same as 7, except instead of returning multiple responses, one for each event, only one event is returned in a response, and the client must perform another Get-Notification to get the next request.

   PRO:
   a. No multiple responses, so can get through any web infrastructure.
   b. No need for the client to close the channel when no longer interested in getting Notifications.
   c. HTTP channel not tied up between events.

   CON:
   a. Poor man's Get-Attribute (but with queuing).
   b. Can result in many operations to be executed, one for each Notification with a connection per Notification.
   c. Not useful for page level notifications.

9. **In-band Get-Notification with queued events with all events in one Get-Notification response -** Same as 7, except get back all events in one response that have been queued, rather than one per multipart MIME response.

   PRO:
   a. Can be used with page level notification.

   CON:
   a. Same as 7.

10. **HTTP-Based IPP Notification Delivery Protocol -** Subscriptions are created as in the [ipp-not]. The "notify-recipients" URL has a new scheme with a new default port number assigned: ipp-ntfy. The IPP Printer does a new operation Send-Notifications operation to the Notification Recipient using a new MIME media type: "application/ipp-ntfy" with the same encoding as "application/ipp". The Notification Recipient returns an "application/ipp-ntfy" response. See the Hugo's proposal: **<draft-ietf-ipp-not-http-delivery-00.txt>**.

    PRO:
    a. Returns notification. Notification Recipient can cancel when not the subscription owner.
    b. Anyone can be the notification recipient, not just the subscriber.
    c. Events can be batched (also across subscriptions) into one notification message.

    CON:
    a. Requires that the Notification recipient is an HTTP server.
    b. Firewalls may not let this through.
    c. Needs a new format (MIME-type) and a new default port.

11. **SNMP traps over UDP** - Subscriptions are created as in the [ipp-not]. The SNMP traps for the Printer MIB and Job Monitoring MIB are used to deliver the notification using SNMP. See Ira's and my proposal: **<draft-ietf-ipp-not-over-snmp-00.txt>** (.pdf).

    PRO:
    a. No connection overhead.
    b. Builds on Printer and Job MIBs.
    c. SNMP v1 over UDP transport is ubiquitous and can use existing infrastructures (also for management).

    CON:
    a. End user workstation may not have SNMP implemented.
    b. End user workstation may not have SNMP implemented.
    c. SNMP may have problems to go through firewalls.
    d. UDP may be unreliable over a wider area.
    e. Limited content length.

12. **SNMP informs over UDP** - Same as 11, except using SNMP informs.

    PRO:
    a. No connection overhead.
    b. More reliable than 11

    CON:
    a. SNMP v3 not widely deployed.
    b. Limited content length.

13. **UDP delivery as Send-Notifications without response -** Subscriptions are created as in the [ipp-not]. Notifications are delivered using a new IPP Send-Notification request that is encoded as an 'application/ipp' MIME type and sent as a raw UDP packet. No response.

    PRO:
    a. No connection overhead.
    b. Good for page level notifications.

    CON:
    a. We may need to add congestion control. Discouraged by IETF AD.
    b. Users likely to want other events than page level events, via other delivery methods, so requires multiple subscriptions which may not be supported.
    c. Firewall problems likely.
    d. Needs special IPP notification listener code.
    e. Limited content length.

14. **UDP delivery as Send-Notifications with response -** Same as 13, except the Notification Recipient returns an 'application/ipp' MIME type response as

a raw UDP packet.

PRO:
a. More reliable than 13, allows to make retries if no response.
b. Returns notification. Notification Recipient can cancel when not the subscription owner.
c. Anyone can be the notification recipient.

CON:
a. We may need to add congestion control. Discouraged by IETF AD.
b. Users likely to want other events than page level events, via other delivery methods, so requires multiple subscriptions which may not be supported.
c. Need retry counter which isn't in our current Notification model.
d. Firewall problems likely.
e. Needs special IPP notification listener code.
f. Limited content length.

15. **TCP delivery as Send-Notifications without response -** Same as 13, except TCP is used instead of UDP.

    PRO:
    a. More reliable than 13.
    b. More likely that IETF will approve than UDP.

    CON:
    a. Needs a new format (MME-type) and a new default port.
    b. Needs special IPP notification listener code.

16. **TCP delivery as Send-Notifications with response -** Same as 14, except TCP is used instead of UDP.

    PRO:
    a. More reliable than 14.
    b. More likely that IETF will approve than UDP.
    c. Returns notification. Notification Recipient can cancel when not the subscription owner.
    d. Anyone can be the notification recipient.
    e. Events can be batched (also across subscriptions) into one notification message.

    CON:
    a. Needs a new format (MME-type) and a new default port.
    b. Needs special IPP notification listener code.

17. **Email -** Subscriptions are created as in the [ipp-not]. The notifications are delivered by email either as Human Consumable OR Machine Consumable according to the "notify-text-format".

PRO:
a. Goes through most firewalls.
b. Proven in shipping products.
c. Can send to paging service which contacts Ultimate Recipient via phone.

CON:
a. Not good for page level notifications
b. High spam risk.
c. Store-and-forward may slow down delivery time.

18. **Email with only Human Consumable (in message body)**

PRO:
a. Same as 17.
b. Can go through all fire walls (no attachments).
c. Can be made without attachments; attachments are not supported by all email systems.
d. Can send to paging service which contacts Ultimate Recipient via phone.

CON:
a. A program can't process.
b. Forces Notification Source to localize messages.

19. **Email with both Human Consumable (in message body) and Machine Consumable (as MIME attachment) in the same notification -** same as 17 with addition that Machine Consumable is sent as an attachment**.**

PRO:
a. Same as 17.
b. Both forms always sent.

CON:
a. Harder to implement.
b. Some firewalls don't accept attachments.
c. Some mail systems don't accept attachments; the attachment is discarded.

20. **Email with either or both Human Consumable and Machine Consumable -** Same as 17 or 19 as subscriber/client choice

PRO:
a. Same as 18.
b. More flexible.

CON:
a. More difficult to implement

b. More difficult for the end user to understand the options

21. **Instant Messaging** - Subscriptions are created as in the [ipp-not]. The "notify-recipient" specifies the Instant Messaging service as the Notification Recipient. This delivery method is an example where a notification service is used by the client transparently to the IPP Printer because the IPP Printer sends the notifications to the Notification Recipient which just happens to be the Instant Messaging service. The IM subscriber number identifying the Ultimate Recipient is passed as a parameter in the URL specified by the client.

    PRO:
    a. Several IM services in use today.
    b. Faster than email.
    c. UI pop-up messages available.
    d. Some IM service store messages if they cannot be delivered immediately.

    CON:
    a. No common standard for IM yet. Need to check how private IMs use URLs today.
    b. May require new URL scheme.
    c. Limited content length.
    d. Limited to Human Consumable notifications.
    e. Special port needs to be enabled for traffic. May not be allowed over firewalls.

22. **Paging** - Same as 21, except that the notification service is a Paging System. The phone number of the Ultimate Recipient is sent as a parameter in the URL specified by the client.

    PRO:
    a. More common than IM.
    b. Notification Recipient does not need Internet connection.
    c. Recipients likely to keep pager with them at all times.

    QUESTION: Can mailto: scheme be used?

    CON:
    a. Even more limited content length.
    b. Only Human Consumable.
    c. Have to check which URL scheme to use. Can IFAX scheme be used?

23. **Configured operator paging** - There is no subscription object created. Only a new "operator-paging-number (text)" Printer Description attribute is added that the System Administrator can configure. Only Printer events that stop

the device are sent by calling the indicated number. No subscription object needed.

PRO:
a. Dead simple to implement in IPP Printer.

CON:
a. Even more limited content length.
b. Only Human Consumable.
c. Have to check which URL scheme to use. Can IFAX scheme be used?
d. Different model for how to make subscription. Poor man's version. Can conflict with other ways to subscribe.
e. We can already do this with regular subscriptions.
f. No common standard on how to introduce messages into the paging service, though some mail services provide an interface.

24. **Configured operator delivery -** Same as 23, except that the new Printer Description attribute is "operator-recipient (url)" so that any delivery method may be used that has text delivery.

    PRO:
    a. Simple to implement in IPP Printer.
    b. Allows any delivery scheme to be used.

    CON:
    a. Same as 23.

25. **Configured operator delivery with event filter -** Same as 24, with the addition of an "operator-events (1setOf type2 keyword)" Printer attribute as well, so that the event can be configured.

    PRO:
    a. Simple to implement in IPP Printer.

    CON:
    a. Same as 23.

26. **Configured operator delivery with reasons filter -** Same as 25, with the addition of an "operator-printer-state-reasons (1setOf type2 keyword)" Printer attribute as well, so that specific printer state reasons can be specified to cause notifications to be sent.

    ISSUE for [ipp-not] - Add filtering on state reasons?

    PRO:

    CON:

a. Would require adding filtering by the Notification Source on state reasons to be usable, because Notification Recipient can't.
b. Same as 23.

27. **Alternatives 1, 2, and 3 with all Subscriptions -** Same as 1, 2, and 3 with the addition that the client can use Create-Printer-Subscription and Create-Job-Subscription operations to request the notifications, as well as Job Creation requests.

    PRO:
    a. More flexibility, will allow Per-Job and Per-Printer subscriptions.

    CON:

28. **Configured Notification Service methods** - A third party notification service that the Printer is configured to work with (see Hugo's White Paper) offers a number of notification delivery methods that the Printer wants to offer transparently to itself. So the IPP Printer adds these schemes to its "notify-schemes-supported" Printer Description attribute.