IPP Protocol White Paper

1.Overview

IPP clients send requests to IPP printers and get responses back in return. A request or a response is transmitted in one or more IPP messages.

```
      Request-Message = IPP Request-line
                        Entity-Header
                        CRLF
                        [Entity-Body]

      Response-Message = IPP Status-line
                         Entity-Header
                         CRLF
                         [Entity-Body]
```

2.The Request-Line

The first line of any IPP request is the request-line. It has the form

```
      IPP-Request-Line = Operation-token  “IPP/1.0”  CRLF

      Operation-token = “Print”
                      | “Cancel-Job”
                      | “Get-Attributes”
                      | “Get-Jobs”
```

3. The Entity-Body

The data associated with an IPP request or response is transmitted in the entity-body of one or more messages. An IPP entity-body also contains Content-Headers which aid the receiver in interpreting and responding to the data.

3.1.Content-Header Fields

Content-Header fields define the type of data and provide information required by the receiver to properly interpret and respond to the data transmitted in each entity-body.

```
      Content-Header = Content-Type
                     | Content-Length
                     | Content-Segment-Flag
                     | Content-Sequence-Number
                     | Content-Response-Required-Flag
```

3.1.1. Header Syntax

IPP headers defined in this document conform to the rules of RFC 822. All IPP headers are of the form:

```
IPP Header = “Content-” field-name “:” [field-value] CRLF.
```

IPP defines the octet sequence CRLF as the end-of-line marker for all protocol elements except the entity-body. All IPP Headers begin with the phrase “Content-“.

3.1.2. Content-Type

The Content-Type field identifies the type of data that follows.

```
Content-Type = Print-Job
             | Attribute-List
             | Document-Part
```

3.1.3. Content-Length

Content-Length defines the length of the following content, in bytes. For example, a document-part of 4096 bytes would have a content-length field of the form

```
Content-Length : 4096
```

3.1.4.Content-Segment-Flag

The Content-Segment-Flag field provides a mechanism for senders to break up the content of a document into pieces. This allows client to transmit the document on the fly, as it is being generated, without having to know the length of the entire document beforehand.

Use of a length field is preferred over the Boundary-string notion of the Multipart/mixed MIME because the sender does not have to determine a unique boundary string for each segment, which may be difficult for some PDLs.

In addition, in combination with Content-Sequence-Number and Content-Response-Required-Flag, this field enables an application to more reliably recover from situations where a print request is being sent to a Printer that cannot queue the document, and printing fails during transmission.

```
Content-Segment-Flag = “only” | “first” | “middle” | “last”
```

A value of ONLY means that the entire document is contained within this entity-body. FIRST, MIDDLE, and LAST refer to this content being the first, a middle, or the last of a series of content-segments to be sent. Each segment would be sent in a separate IPP message.

3.1.5.Content-Sequence-Number

When document content is being sent in segments, it is required that each segment have an appropriate sequence number associated with it, using this field. This field would have the format:

```
      “Content-Sequence-Number” “:” integer
```

For example, if this segment were the third segment in a sequence of content segments, the field would be

```
      Content-Sequence-Number : 3.
```

3.1.6.Segment-Response-Required-Flag

This field allows the sender to request that a response be sent back on each data segment. By setting this flag on, the sender promises not to send another segment until receiving a positive response from the prior segment. The receiver is obligated to send a response to each segment. When the flag is off, the sender will not expect a response to each segment and should send segments continuously until the entire document has been transmitted. The receiver, on the other hand, would only send a response if there were an error condition.

3.2. The Print Job

A Print Job contains print job attributes and one or more documents. A Print Job is always terminated with an End-Of-Job-Marker. The End-Of-Job Marker is required to complete a Print Request. If no End-Of-Job Marker is sent, the Printer will wait for it until an established time-out period has elapsed.

```
      Print-Job = Print-Job-Header
                  [Job-Attribute-list]
                  1#(Documents)
                  End-Of-Job-Marker
```

3.2.1. Print-Job-Header

The Print-Job-Header identifies the following data as an IPP Print-Job.

```
            Print-Job Header = “Content-Type : IPP Print-Job CRLF”
```

3.3. The Document

An IPP document contains the attributes of the document and optionally the document content. If no content is not present, a reference to the document must be provided as one of the document attributes.

```
      Document =  Document-Header
                  [Document-Attribute-List]
                  #1(Document-Part)
```

3.3.1. Document-Parts

An IPP Document may be split into multiple Document-Parts for transmission. This makes it possible for IPP clients to send documents a piece at a time, without requiring them to know the length of the entire document beforehand.

3.3.2.Document-Content-Header Fields

Content-Header Fields are used to describe content of the Document-Part.

3.3.2.1. Content-Type

For document content, Content-Type is defined as :

```
      Content-Type = Vendor “/” Data-Stream-Format “/” Version
```

Thus, for example, if the document to be printed was a Postscript Level 2 document, the Content-Type would be specified as:

```
      Content-Type: Adobe/Postscript/2.0
```

3.3.2.2.Content-Length

For document content, Content-Length defines the length of this Document-Part, in bytes.

3.3.3.End-Of-Job-Marker

An end-of-job marker is required to tell the receiver that no more documents are to be sent as part of this job.

End-Of-Job-Marker = “Content-Type : End-Of-Job **CRLF**”

3.3.4. Attributes

Previous sections identified two types of attribute lists, which will be further defined here.

```
      Job-Attribute-List = Job-Attribute
                           0#[Job-Attribute]

      Document-Attribute-List = Document-Attribute
                                0#[Document-Attribute]

      Job-Attribute = Job-Informational-Attribute
                    | Job-Status-Attribute
```

```
                        | Notification-Attribute
                        | Job-Production-Attribute
                        | Conversion-Attribute
                        | Job-Resource-Attribute.

      Document-Attribute = Document-format
                             | document-name
                             | document-URL
```

All attributes will be of the form

```
      Attribute type = attribute value.
```

4. Mapping to MIME Types

If it is thought useful to map the IPP Entity-Body described in the previous sections to a MIME type, the simplest approach would be to define a new MIME-type, Application/IPP. Then one could simply declare the Application/IPP MIME to be the IPP Message, as it has been defined in this paper. However, it should be noted that this MIME type would only operate within the IPP protocol.

5. Mapping to HTTP

If HTTP is used as the “transport” protocol, then the IPP Request Message, as defined in this document, would be the Entity-Body of an HTTP Post method. The IPP Response Message would be the Entity-Body of the corresponding HTTP Response.

6. Example flows

Several examples will be shown to illustrate the use of the protocol as defined in this section. Only the IPP operations and the contents of the entity-bodies will be shown in these scenarios.

6.1.1.Scenario 1

In this scenario, a client sends a print job stored as a complete file to an IPP printer implemented in a server. The server is capable of spooling jobs. The job contains a single document which is received by the server with no error and is queued for printing at a later time.

```
Client                                                     Server

----------------------------------------------------------->
      Print IPP/1.0                            ; Request-Line
      Content-Segment-Flag : only
      Content-Response-Required-Flag : Yes
      Content-Type : IPP Print Job             ; Print Job Marker
        <Job-Attribute-List>
      Content-Type : IPP Document              ; Document Marker
        <Document-Attribute-List>
      Content-Type : Adobe/Postscript/2.0
      Content-Length : 12,150
        <12,150 bytes of Postscript data>
      Content-Type : End of Job                ; End of job  Marker


< -----------------------------------------------------------
      IPP Response = Received and Queued
         Current-job-state = processing
         Job-Identifier = 12
```

6.1.2.Scenario 2

This case is identical to scenario #1, except that an error occurred during the transmission that made the request invalid. An error response is returned.

```
Client                                                     Server

---------------------------------------------------------- >
      Print  IPP/1.0                          ; Request-Line
      Content-Segment-Flag : only
      Content-Response-Required-Flag : Yes
      Content-Type : IPP Print Job            ; Print-Job Marker
        <Job-Attribute-List>
      Content-Type : IPP Document             ; Document-Marker
        <Document-Attribute-List>
      Content-Type : Adobe/Postscript/2.0
      Content-Length : 12,150
        <12,150 bytes of Postscript data>
      Content-Type : End of Job               ; End of job marker


< -----------------------------------------------------------
      IPP Response = Invalid Request
```

6.1.3. Scenario 3

This case is identical to scenario #1 except that the document to be printed is being sent a piece at a time. In this case, the driver generates 4K segments. This requires 2 segments of 4K each and a final segment of 3,958bytes (total = 12,150 bytes). Since the server can spool the data, it is not necessary to ask for a response on each segment.

```
Client                                                     Server

---------------------------------------------------------->
      Print  IPP/1.0                          ; Request-Line
      Content-Segment-Flag : first
      Content-Response-Required-Flag : no
      Content-Sequence-Number : 1
      Content-Type : Print Job                ; Print-Job Marker
        <Job-Attribute-List>
      Content-Type : IPP Document             ; Document-Marker
        <Document-Attribute-List>
      Content-Type : Adobe/Postscript/2.0
      Content-Length : 4096
        <4096 bytes of Postscript data>

---------------------------------------------------------->
      Print  IPP/1.0
      Content-Segment-Flag : middle
      Content-Response-Required-Flag : no
      Content-Sequence-Number : 2
      Content-Type : Adobe/Postscript/2.0
      Content-Length : 4096
        <4096 bytes of Postscript data>

---------------------------------------------------------->
      Print  IPP/1.0
      Content-Segment-Flag : last
      Content-Response-Required-Flag : yes
      Content-Sequence-Number : 3
      Content-Type : Adobe/Postscript/2.0
      Content-Length : 3958
        <3958 bytes of Postscript data>
      Content-Type : End of Job               ; End of job marker

< ----------------------------------------------------------
IPP Response = Received and Queued
      Current-job-state = processing
      Job-Identifier = 12
```

6.1.4. Scenario 4

This scenario is similar to the previous one except that the client knows the Printer is not capable of spooling the print job before starting to print. Therefore, printer errors may occur during transmission, and the job may have to be aborted. The client therefore will ask for a response on each message. In this case a printer jam occurs during the processing of the second message. The client responds with last segment containing an End-of-Job Marker, which terminates the job.

```
Client                                                     Server

---------------------------------------------------------->
      Print  IPP/1.0                             ; Request-Line
      Content-Segment-Flag : first
      Content-Response-Required-Flag : Yes
      Content-Sequence-Number : 1
      Content-Type : IPP Print Job               ; Print-Job Marker
        <Job-Attribute-List>
      Content-Type : IPP Document                ; Document-Marker
        <Document-Attribute-List>
      Content-Type : Adobe/Postscript/2.0
      Content-Length : 4096
        <4096 bytes of Postscript data>

< -----------------------------------------------------------
      IPP Response = Received
         Current-job-state = printing

--------------------------------------------------------->
      Print  IPP/1.0
      Content-Segment-Flag : middle
      Content-Response-Required-Flag : Yes
      Content-Sequence-Number : 2
      Content-Type : Adobe/Postscript/2.0
      Content-Length : 4096
        <4096 bytes of Postscript data>

< -----------------------------------------------------------
      IPP Response = Received
         Current-job-state = printer-jammed

------------------------------------------------------------- >
      Print  IPP/1.0
      Content-Segment-Flag : last
      Content-Response-Required-Flag : Yes
      Content-Sequence-Number : 3
      Content-Type : End of Job

< ------------------------------------------------------------
      IPP Response = job Terminated
```

6.1.5. Scenario 5

This scenario is identical to the previous one, except that the end-user walks to the printer and clears the jam. The client starts transmitting at the next unsent document-part. The Printer must recover any pages not printed in the last document-part sent.

```
Client                                                        Server

------------------------------------------------------------ >
      Print  IPP/1.0                              ; Request-Line
      Content-Segment-Flag : first
      Content-Response-Required-Flag : Yes
      Content-Sequence-Number : 1
      Content-Type : IPP Print Job                ; Print-Job Marker
        <Job-Attribute-List>
      Content-Type : IPP Document                 ; Document-Marker
        <Document-Attribute-List>
      Content-Type : Adobe/Postscript/2.0
      Content-Length : 4096
        <4096 bytes of Postscript data>

< ------------------------------------------------------------
      IPP Response = Received
         Current-job-state = printing

----------------------------------------------------------- >
      Print  IPP/1.0
      Content-Segment-Flag : middle
      Content-Response-Required-Flag : Yes
      Content-Sequence-Number : 2
      Content-Type : Adobe/Postscript/2.0
      Content-Length : 4096
        <4096 bytes of Postscript data>

< ------------------------------------------------------------
      IPP Response = Received
         Current-job-state = printer-jammed

-------------------------------------------------------------- >
      Print  IPP/1.0
      Content-Segment-Flag : last
      Content-Response-Required-Flag : Yes
      Content-Sequence-Number : 3
      Content-Type : Adobe/Postscript/2.0
      Content-Length : 3958
        <43958 bytes of Postscript data>
      Content-Type : End of Job

< -------------------------------------------------------------
      IPP Response = job Completed
```

6.1.6. Scenario 6

In this scenario, the client send a print job containing two documents to the Printer. The Job is sent as one IPP Message. It is spooled on the Printer.

```
Client                                                     Server

----------------------------------------------------------->
      Print IPP/1.0                            ; Request-Line
      Content-Segment-Flag : only
      Content-Response-Required-Flag : Yes
      Content-Type : IPP Print Job             ; Print Job Marker
        <Job-Attribute-List>
      Content-Type : IPP Document              ; Document Marker
        <Document-Attribute-List>
      Content-Type : Adobe/Postscript/2.0
      Content-Length : 12,150
        <12,150 bytes of Postscript data>
      Content-Type : HP/PCL/5e
      Content-Length : 3,568
        <3,568 bytes of PCL/5e>
      Content-Type : End of Job                ; End of job  Marker


< -----------------------------------------------------------
      IPP Response = Received and Queued
         Current-job-state = processing
         Job-Identifier = 12
```

6.1.7. Scenario 7

This scenario is identical to the previous one, except that the documents are generated on the fly, in 4K blocks.

```
Client                                                    Server

---------------------------------------------------------->
      Print IPP/1.0                           ; Request-Line
      Content-Segment-Flag : first
      Content-Response-Required-Flag : no
      Content-Sequence-Number : 1
      Content-Type : IPP Print Job            ; Print Job Marker
        <Job-Attribute-List>
      Content-Type : IPP Document             ; Document Marker
        <Document-Attribute-List>
      Content-Type : Adobe/Postscript/2.0
      Content-Length : 4096
        <4096 bytes of Postscript data>

-------------------------------------------------------------- >

      Client Print IPP/1.0                    ; Request-Line
      Content-Segment-Flag : middle
      Content-Response-Required-Flag : no
      Content-Sequence-Number : 2
      Content-Type : Adobe/Postscript/2.0
      Content-Length : 4096
        <4096 bytes of Postscript data>

---------------------------------------------------------->

      Print IPP/1.0                           ; Request-Line
      Content-Segment-Flag : middle
      Content-Response-Required-Flag : no
      Content-Sequence-Number : 3
      Content-Type : Adobe/Postscript/2.0
      Content-Length : 4096
        <4096 bytes of Postscript data>

------------------------------------------------------------ >

      Print IPP/1.0                           ; Request-Line
      Content-Segment-Flag : last
      Content-Response-Required-Flag : yes
      Content-Sequence-Number : 4
      Content-Type : IPP Document             ; Document Marker
        <Document-Attribute-List>
      Content-Type : HP/PCL/5e
      Content-Length : 3568
        <3568 bytes of PCL/5e data>
      Content-Type : End of Job
```

```
< -------------------------------------------------------------

      IPP Response = Received and Queued
         Current-job-state = processing
         Job-Identifier = 12
```