

1 **IPP Protocol White Paper**

2
3 1.Overview

4 IPP clients send requests to IPP printers and get responses back in
5 return. A request or a response is transmitted in one or more IPP
6 messages. When multiple IPP messages are required to transport a request
7 or a response, each message will be called a segment of that request or
8 response.

9
10 Request-Message = IPP Request-line
11 Entity-Header
12 **CRLF**
13 [Entity-Body]

14
15 Response-Message = IPP Status-line
16 Entity-Header
17 **CRLF**
18 [Entity-Body]

19
20 2.The Request-Line

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

23
24 IPP-Request-Line = Operation-token "IPP/1.0" **CRLF**

25
26 Operation-token = "Print"
27 | "Cancel-Job"
28 | "Get-Attributes"
29 | "Get-Jobs"

30
31 3.The Status-Line

32 After receiving and interpreting a request-message, a server must
33 respond with an IPP response-message if asked. The first line of a
34 response is the status line, consisting of the protocol version followed
35 by a numeric status code and its associated reason-phrase. When the
36 requested IPP operation requires data (e.g. an attribute list) to be
37 returned in the response, the data will be contained within the entity-
38 body of the response. Elements of the status-line are separated by space
39 characters and the line is terminated with **CRLF**. Thus, a status line
40 might look like:

41
42 "IPP/1.0" status-code text **CRLF**

43
44 3.1.Status Codes

45
46 The status code is a three digit integer result code which defines the
47 response of the server in attempting to understand and execute the

48 operation specified in the request. A reason-phrase provides additional,
49 human readable information about the status condition posted.

50

51 The first digit of the reason code defines the class of response:

52

- 53 • 1xx: Reserved
- 54 • 2xx: Success - the request was successfully received,
55 understood, and accepted
- 56 • 3xx: Conditional Success - Recovery may be required
- 57 • 4xx: Client Error: the request contains bad syntax or cannot be
58 fulfilled
- 59 • 5xx: Server Error: the server failed to fulfill an apparently
60 valid request

61

62 4.Header Syntax

63 IPP headers defined in this document (Entity-Headers and Content-
64 Headers) conform to the rules of RFC 822. All IPP headers are of the
65 form:

66

67 IPP Header = field-name ":" [field-value] **CRLF**.

68

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

72

73 5. Entity-Headers

74 Entity-Headers contain optional information which applies to the entire
75 entity-body, and is required to correctly process the entity-body.
76 Examples include things like data compression, security tokens,
77 localization information, and the like. Specific Entity-headers are to
78 be defined.

79

80 6. The Entity-Body

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

85

86 6.1.Content-Header Fields

87 Content-Header fields appear throughout an entity-body, and define
88 boundary markers and protocol flags that are required to correctly parse
89 and respond to the content of the entity-body. Five types of Content-
90 Headers are defined, which are described in subsequent sections of this
91 document:

92

93 Content-Header = Content-Type
94 | Content-Length

```
95         | Content-Segment-Flag
96         | Content-Sequence-Number
97         | Content-Response-Required-Flag
98
99
```

100 6.1.1. Content-Type

101 The Content-Type field is used to mark content boundaries within the
102 entity-body, e.g. the beginning of a print job. Four boundary markers
103 are defined in subsequent sections of this document:

```
104
105     Boundary-Marker = Print-Job-Marker
106                     | Document-Marker
107                     | Document-Part-Marker
108                     | End-of-Job Marker
109
```

110 6.1.2. Content-Length

111 Content-Length defines the length of the following data, in bytes. It is
112 used specifically to define the length of a document-part. For example,
113 the content-length field

```
114
115     Content-Length : 4096
116
```

117 indicates that the next 4096 bytes in the stream is document data and is
118 not to be interpreted by the IPP process. Use of a length field is
119 preferred over the Boundary-string notion of the Multipart/mixed MIME
120 because the sender does not have to determine a unique boundary string
121 for each segment, which may be difficult for some PDLs.

124 6.1.3. Content-Segment-Flag

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

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

```
134
135
136     Content-Segment-Flag = "only" | "first" | "middle" | "last"
137
```

138 A value of ONLY means that the entire document is contained within this
139 entity-body. FIRST, MIDDLE, and LAST refer to this content being the
140 first, a middle, or the last of a series of content-segments to be sent.
141 Each segment would be sent in a separate IPP message. If no Content-
142 Segment-Flag is present in a message, Content-Segment is assumed to be
143 "ONLY".

144

145 6.1.4.Content-Sequence-Number

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

149
150 "Content-Sequence-Number" ":" integer

151
152 For example, if this segment were the third segment in a sequence of
153 content segments, the field would be

154
155 Content-Sequence-Number : 3

156
157 6.1.5.Segment-Response-Required-Flag

158 This field allows the sender to request that a response be sent back on
159 each data segment. By setting this flag on, the sender promises not to
160 send another segment until receiving a positive response from the prior
161 segment. The receiver is obligated to send a response to each segment.
162 When the flag is off, the sender will not expect a response to each
163 segment and should send segments continuously until the entire document
164 has been transmitted. The receiver, on the other hand, would only send a
165 response if there were an error condition. If no Segment-Response-
166 Required-Flag is present in a message, Segment-Response-Required is
167 assumed to be "YES".

168
169
170 6.2. The Print Job

171 A Print Job contains print job attributes and one or more documents. A
172 Print Job is always terminated with an End-Of-Job-Marker. The End-Of-Job
173 Marker is required to complete a Print Request. If no End-Of-Job Marker
174 is sent, the Printer will wait for it until an established time-out
175 period has elapsed. A Printer that does not spool print jobs must not
176 receive any intervening jobs until it has received an end-of-job marker
177 for the current job, or it times out.

178
179
180 Print-Job = Print-Job-Marker
181 [Job-Attribute-list]
182 1#Document
183 End-Of-Job-Marker

184
185
186 6.2.1. Print-Job-Marker

187 The Print-Job-Marker identifies the data that follows as an IPP Print-
188 Job.

189
190 Print-Job-Marker = "Content-Type : IPP-Print-Job"

191

192 6.3. The Document

193 An IPP document contains the attributes of the document and optionally
194 the document content. If no content is present, a reference to the
195 document must be provided as one of the document attributes.

```
196  
197     Document = Document-Marker  
198               [Document-Attribute-List]  
199               0#Document-Part  
200
```

201 6.3.1. Document-Marker

202 A Document Marker is defined as

```
203     "Content-Type : IPP-Document"
```

204
205 6.3.2. Document-Parts

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

```
210  
211     Document-Part = Document-Part-Marker  
212                   Document-Part-Length-Field  
213                   Document-Part-content  
214
```

215 6.3.2.1. Document-Part-Marker

216 A Document-Part-Marker is defined as

```
217     "Content-Type" ":" Vendor "/" Data-Stream-Format "/" Version  
218
```

219 Thus, for example, if the document-part contained part of a Postscript
220 Level 2 document, the Document-Part-Marker would be specified as:

```
221  
222     Content-Type: Adobe/Postscript/2.0  
223
```

224 An alternative scheme would be to use existing registered MIME types to
225 identify the data stream format. Some PDLs would have to obtain
226 registered MIME types.

227
228
229 6.3.2.2. Document-Part-Length-Field

230 For a document-part, Content-Length defines the length of this document-
231 part, in bytes.

232
233 6.3.2.3. Document-Content

234 Document-content is the actual PDL of the document-part being sent.

235 6.3.3.End-Of-Job-Marker

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

238
239 End-Of-Job-Marker = "Content-Type : End-Of-Job"
240

241 6.3.4. Attribute-Lists

242 6.3.4.1.1.IPP defines three types of attribute lists:

243 Job-Attribute-List = Job-Attribute
244 0#(; Job-Attribute)
245

246 Document-Attribute-List = Document-Attribute
247 0#(; Document-Attribute)
248

249 Printer-Attribute-List = Printer-Attribute
250 0#(; Printer-Attribute)
251

252
253 Attributes are described in detail in "Internet Printing Protocol/1.0:
254 Model and Semantics". All attributes will be of the form

255
256 Attribute type = attribute value
257

258
259 7. Mapping to MIME Types

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

266
267 8. Mapping to HTTP

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

272
273 9.Mapping directly to TCP Sockets

274 To use a TCP connection to transport IPP messages, a new port number has
275 to be defined. The suggested port number to be registered with the IANA
276 is nnn. When using this method, an IPP server opens the socket in LISTEN
277 mode and a client connects to it with an unused source port in the
278 unrestricted range. This socket pair is the unique identifier for the
279 connection. Once the connection reaches ESTABLISHED state, it transports
280 at least one complete IPP message exchange. One complete IPP message

281 exchange is defined by one Request-Message followed by one Response-
282 Message. A multi-segment print message, terminated by an End-of-Job
283 Marker, is treated as one request. After the exchange the TCP connection
284 can be disconnected by the server but may stay up to support further
285 transactions. An idle timer of nn minutes must be used to terminate the
286 connection if the option of keeping it open is used.

287

288 10. Example flows

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

292
 293 10.1.1.Scenario 1

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

298
 299 Client

Server

```

300 ----- >
301
302 Print IPP/1.0 ; Request-Line
303 Content-Segment-Flag : only
304 Content-Response-Required-Flag : Yes
305 Content-Type : IPP-Print-Job ; Print Job Marker
306 <Job-Attribute-List>
307 Content-Type : IPP-Document ; Document Marker
308 <Document-Attribute-List>
309 Content-Type : Adobe/Postscript/2.0
310 Content-Length : 12,150
311 <12,150 bytes of Postscript data>
312 Content-Type : End of Job ; End of job Marker
313
314
315 < -----
316 IPP/1.0 2xx Job Received and Queued
317 Current-job-state = processing ;attribute-list
318 Job-Identifier = 12
319
    
```


320
 321 10.1.2.Scenario 2

322 This case is identical to scenario #1, except that the request is
 323 invalid for some reason. An error response is returned.

```

324 Client
325                                     Server
326
327 ----->
328     Print IPP/1.0                               ; Request-Line
329     Content-Segment-Flag : only
330     Content-Response-Required-Flag : Yes
331     Content-Type : IPP-Print-Job                ; Print-Job Marker
332     <Job-Attribute-List>
333     Content-Type : IPP-Document                ; Document-Marker
334     <Document-Attribute-List>
335     Content-Type : Adobe/Postscript/2.0
336     Content-Length : 12,150
337     <12,150 bytes of Postscript data>
338     Content-Type : End of Job                  ; End of job marker
339
340
341 < -----
342     IPP/1.0 4xx Invalid Request
343
    
```

344
345 10.1.3.Scenario 3

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

352
353 Client

Server

```

354 ----- >
355
356 Print IPP/1.0 ; Request-Line
357 Content-Segment-Flag : first
358 Content-Response-Required-Flag : no
359 Content-Sequence-Number : 1
360 Content-Type : IPP-Print-Job ; Print-Job Marker
361 <Job-Attribute-List>
362 Content-Type : IPP-Document ; Document-Marker
363 <Document-Attribute-List>
364 Content-Type : Adobe/Postscript/2.0
365 Content-Length : 4096
366 <4096 bytes of Postscript data>
367
368 ----- >
369
370 Print IPP/1.0
371 Content-Segment-Flag : middle
372 Content-Response-Required-Flag : no
373 Content-Sequence-Number : 2
374 Content-Type : Adobe/Postscript/2.0
375 Content-Length : 4096
376 <4096 bytes of Postscript data>
377
378 ----- >
379
380 Print IPP/1.0
381 Content-Segment-Flag : last
382 Content-Response-Required-Flag : yes
383 Content-Sequence-Number : 3
384 Content-Type : Adobe/Postscript/2.0
385 Content-Length : 3958
386 <3958 bytes of Postscript data>
387 Content-Type : End of Job ; End of job marker
388
389 < -----
390 IPP/1.0 2xx Job Received and Queued
391 Current-job-state = processing ; Attribute-list
392 Job-Identifier = 12

```

392 10.1.4.Scenario 4

393 This case is identical to the previous one except that the first segment
 394 contains a syntax error and cannot be processed. Since Content-Response-
 395 Required is set to NO, a response to the first segment is sent
 396 asynchronously. The server flushes any subsequently received segments.
 397 The client cannot recover so terminates the job with an end-of-job
 398 marker.

```

399
400 Client                                     Server
401
402 ----->
403     Print IPP/1.0                           ; Request-Line
404     Content-Segment-Flag : first
405     Content-Response-Required-Flag : no
406     Content-Sequence-Number : 1
407     Content-Type : IPP-Print-Job           ; Print-Job Marker
408     <Job-Attribute-List>
409     Content-Type : IPP-Document           ; Document-Marker
410     <Document-Attribute-List>
411     Content-Type : Adobe/Postscript/2.0
412     Content-Length : 4096
413     <4096 bytes of Postscript data>
414
415 ----->
416     Print IPP/1.0
417     Content-Segment-Flag : middle
418     Content-Response-Required-Flag : no
419     Content-Sequence-Number : 2
420     Content-Type : Adobe/Postscript/2.0
421     Content-Length : 4096
422     <4096 bytes of Postscript data>
423
424 < -----
425     IPP/1.0 4xx Client syntax error
426     Content-Sequence-Number : 1
427
428 ----->
429     Print IPP/1.0
430     Content-Segment-Flag : last
431     Content-Response-Required-Flag : yes
432     Content-Sequence-Number : 3
433     Content-Type : End-Of-Job
434
435 < -----
436     IPP/1.0 2xx Job Terminated
437

```

438 10.1.5. Scenario 5

439 The client knows the Printer is not capable of spooling the print job
440 before starting to print. The client therefore will ask for a response
441 on each message. In this case a printer jam occurs The client can
442 receive the second message, but cannot continue printing. The client
443 responds with last segment containing an End-of-Job Marker, which
444 terminates the job.

```
445
446 Client                                     Server
447 ----->
448
449     Print IPP/1.0                               ; Request-Line
450     Content-Segment-Flag : first
451     Content-Response-Required-Flag : Yes
452     Content-Sequence-Number : 1
453     Content-Type : IPP-Print-Job               ; Print-Job Marker
454     <Job-Attribute-List>
455     Content-Type : IPP-Document                ; Document-Marker
456     <Document-Attribute-List>
457     Content-Type : Adobe/Postscript/2.0
458     Content-Length : 4096
459     <4096 bytes of Postscript data>
460
461 < -----
462     IPP/1.0 2xx Segment Received and Printing
463     Content-Sequence-Number : 1
464     Current-job-state = printing
465
466 ----->
467
468     Print IPP/1.0
469     Content-Segment-Flag : middle
470     Content-Response-Required-Flag : Yes
471     Content-Sequence-Number : 2
472     Content-Type : Adobe/Postscript/2.0
473     Content-Length : 4096
474     <4096 bytes of Postscript data>
475
476 < -----
477     IPP/1.0 3xx = Segment Received, cannot print
478     Content-Sequence-Number : 2
479     Current-printer-state = printer-jammed
480
481 ----->
482
483     Print IPP/1.0
484     Content-Segment-Flag : last
485     Content-Response-Required-Flag : Yes
486     Content-Sequence-Number : 3
487     Content-Type : End of Job
488
489 < -----
490     IPP/1.0 2xx job Terminated
```

489 10.1.6. Scenario 6

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

494

495 Client

Server

496

497 ----- >

498 Print IPP/1.0 ; Request-Line

499 Content-Segment-Flag : first

500 Content-Response-Required-Flag : Yes

501 Content-Sequence-Number : 1

502 Content-Type : IPP-Print-Job ; Print-Job Marker

503 <Job-Attribute-List>

504 Content-Type : IPP-Document ; Document-Marker

505 <Document-Attribute-List>

506 Content-Type : Adobe/Postscript/2.0

507 Content-Length : 4096

508 <4096 bytes of Postscript data>

509

510 < -----

511 IPP/1.0 2xx Segment received and printing

512 Content-Sequence-Number : 1

513 Current-job-state = printing

514

515

516 ----- >

517 Print IPP/1.0

518 Content-Segment-Flag : middle

519 Content-Response-Required-Flag : Yes

520 Content-Sequence-Number : 2

521 Content-Type : Adobe/Postscript/2.0

522 Content-Length : 4096

523 <4096 bytes of Postscript data>

524

525 < -----

526 IPP/1.0 3xx Segment received, cannot print

527 Content-Sequence-Number : 2

528 Current-printer-state = printer-jammed

529

530 ----- >

531 Print IPP/1.0

532 Content-Segment-Flag : last

533 Content-Response-Required-Flag : Yes

534 Content-Sequence-Number : 3

535 Content-Type : Adobe/Postscript/2.0

536 Content-Length : 3958

537 <43958 bytes of Postscript data>

538 Content-Type : End of Job

539

540

541 < -----
542 IPP/1.0 2xx = Segment received and printing
543 Content-Sequence-Number : 3

544 10.1.7. Scenario 7

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

```

548 Client
549 Client
550
551 ----->
552 Print IPP/1.0 ; Request-Line
553 Content-Segment-Flag : only
554 Content-Response-Required-Flag : Yes
555 Content-Type : IPP-Print-Job ; Print Job Marker
556 <Job-Attribute-List>
557 Content-Type : IPP-Document ; Document Marker
558 <Document-Attribute-List>
559 Content-Type : Adobe/Postscript/2.0
560 Content-Length : 12,150
561 <12,150 bytes of Postscript data>
562 Content-Type : HP/PCL/5e
563 Content-Length : 3,568
564 <3,568 bytes of PCL/5e>
565 Content-Type : End of Job ; End of job Marker
566
567
568 < -----
569 IPP/1.0 2xx Job received and Queued
570 Current-job-state = processing
571 Job-Identifier = 12
572
    
```

573 10.1.8. Scenario 8

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

```

576
577
578 Client                                     Server
579
580 ----- >
581     Print IPP/1.0                               ; Request-Line
582     Content-Segment-Flag : first
583     Content-Response-Required-Flag : no
584     Content-Sequence-Number : 1
585     Content-Type : IPP-Print-Job                 ; Print Job Marker
586     <Job-Attribute-List>
587     Content-Type : IPP-Document                 ; Document Marker
588     <Document-Attribute-List>
589     Content-Type : Adobe/Postscript/2.0
590     Content-Length : 4096
591     <4096 bytes of Postscript data>
592
593 ----- >
594
595     Client Print IPP/1.0                         ; Request-Line
596     Content-Segment-Flag : middle
597     Content-Response-Required-Flag : no
598     Content-Sequence-Number : 2
599     Content-Type : Adobe/Postscript/2.0
600     Content-Length : 4096
601     <4096 bytes of Postscript data>
602
603 ----- >
604
605     Print IPP/1.0                               ; Request-Line
606     Content-Segment-Flag : middle
607     Content-Response-Required-Flag : no
608     Content-Sequence-Number : 3
609     Content-Type : Adobe/Postscript/2.0
610     Content-Length : 3958
611     <3958 bytes of Postscript data>
612
613 ----- >
614
615     Print IPP/1.0                               ; Request-Line
616     Content-Segment-Flag : last
617     Content-Response-Required-Flag : yes
618     Content-Sequence-Number : 4
619     Content-Type : IPP Document                 ; Document Marker
620     <Document-Attribute-List>
621     Content-Type : HP/PCL/5e
622     Content-Length : 3568
623     <3568 bytes of PCL/5e data>
624     Content-Type : End of Job
    
```



```
625 < -----  
626  
627     IPP/1.0 2xx = Job Received and Queued  
628         Current-job-state = processing  
629         Job-Identifier = 12  
630
```