Internet Printing Protocol: Model and Protocol Details

IPP Analysts Briefing

Boston

August 27, 1997

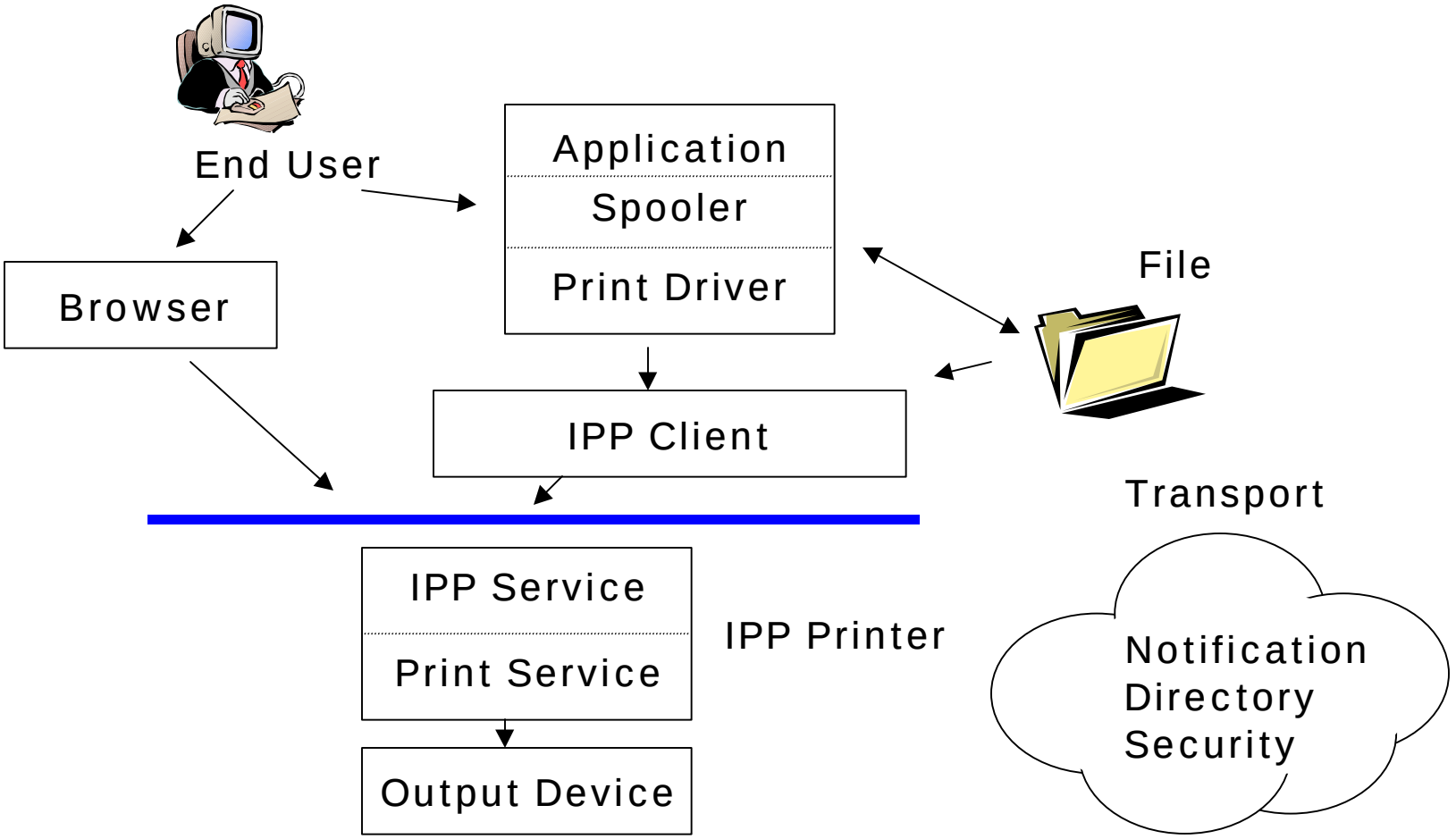
Model and Protocol Details

- IPP - Model and Semantics
  - Abstractions independent of encoding
  - Client/Server
  - Alignment with other Standards
    - Printer MIB, Job MIB, Host Resource MIB
- IPP - Protocol Specification
  - “on-the-wire” data representation
  - Transport specific mapping - HTTP/1.1

Architecture

- Distributed Environment - Internet
- Client
  - Query the Printer
  - Submit Jobs
  - Query Jobs
  - Cancel Jobs
- Server (IPP Printer)
  - Implement and support the Protocol
  - Conform to the Model semantics
  - Integrated with other services
    - Naming, Directory, Notification, Security

Layering


End User
Application
Spooler
Print Driver
Browser
File
IPP Client
Transport
IPP Service
Print Service
IPP Printer
Notification
Directory
Security
Output Device

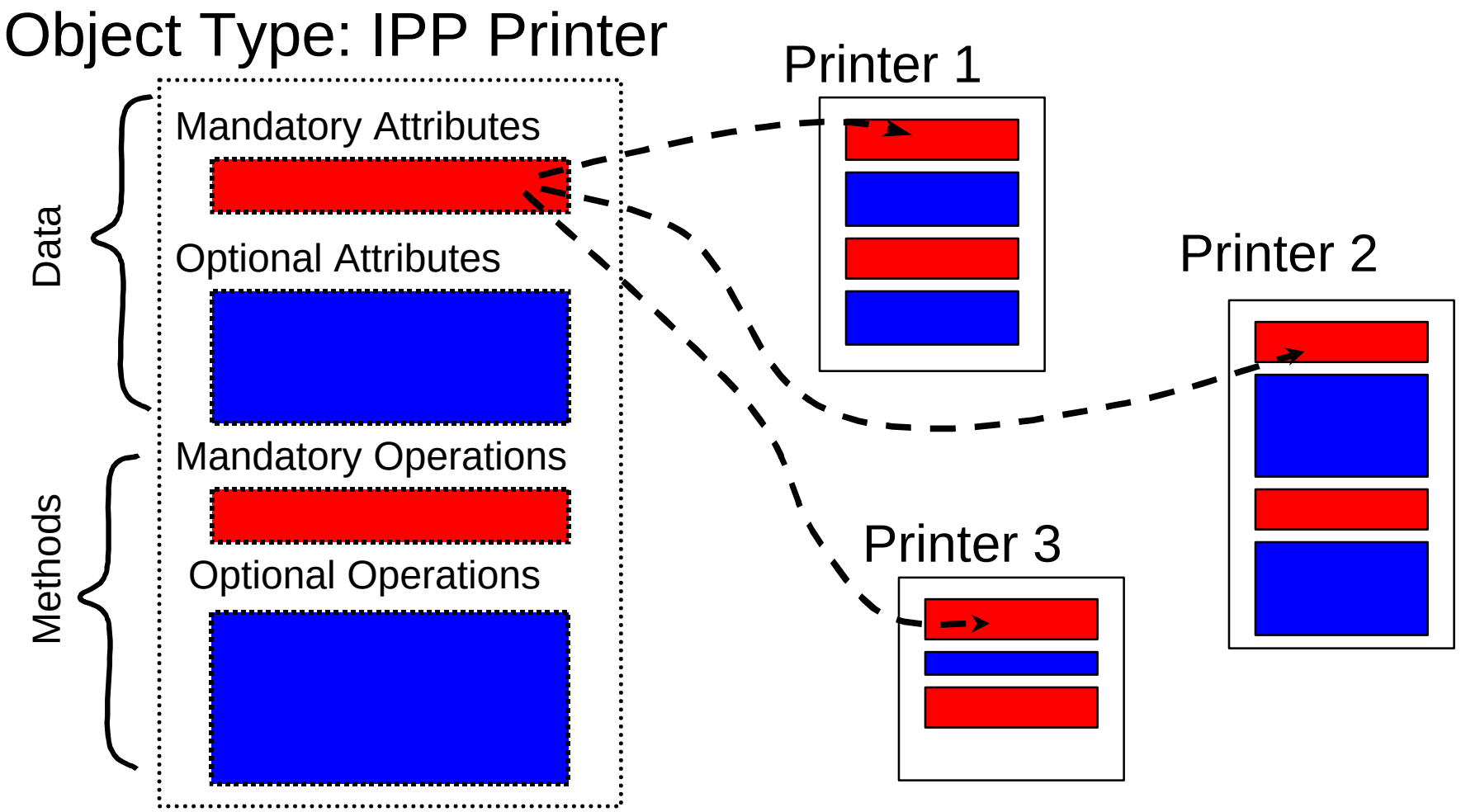
Database Model

- Follow the industry lead
  - Objects with attributes
  - Operations to manipulate those objects
  - Operations to query object (status, attributes)
- Submitting a print job creates a Job object
  - Client supplied attributes
  - Printer supplied attributes (e.g., state, submitter's authenticated identity)

Object Types

- Object Type
  - Definition (schema)
  - Attributes (mandatory, optional)
  - Operations (mandatory, optional)
- Objects
  - Uniquely identifiable implementations
  - Supported attributes (mandatory +)
  - Supported operations (mandatory +)

Object Types: Example



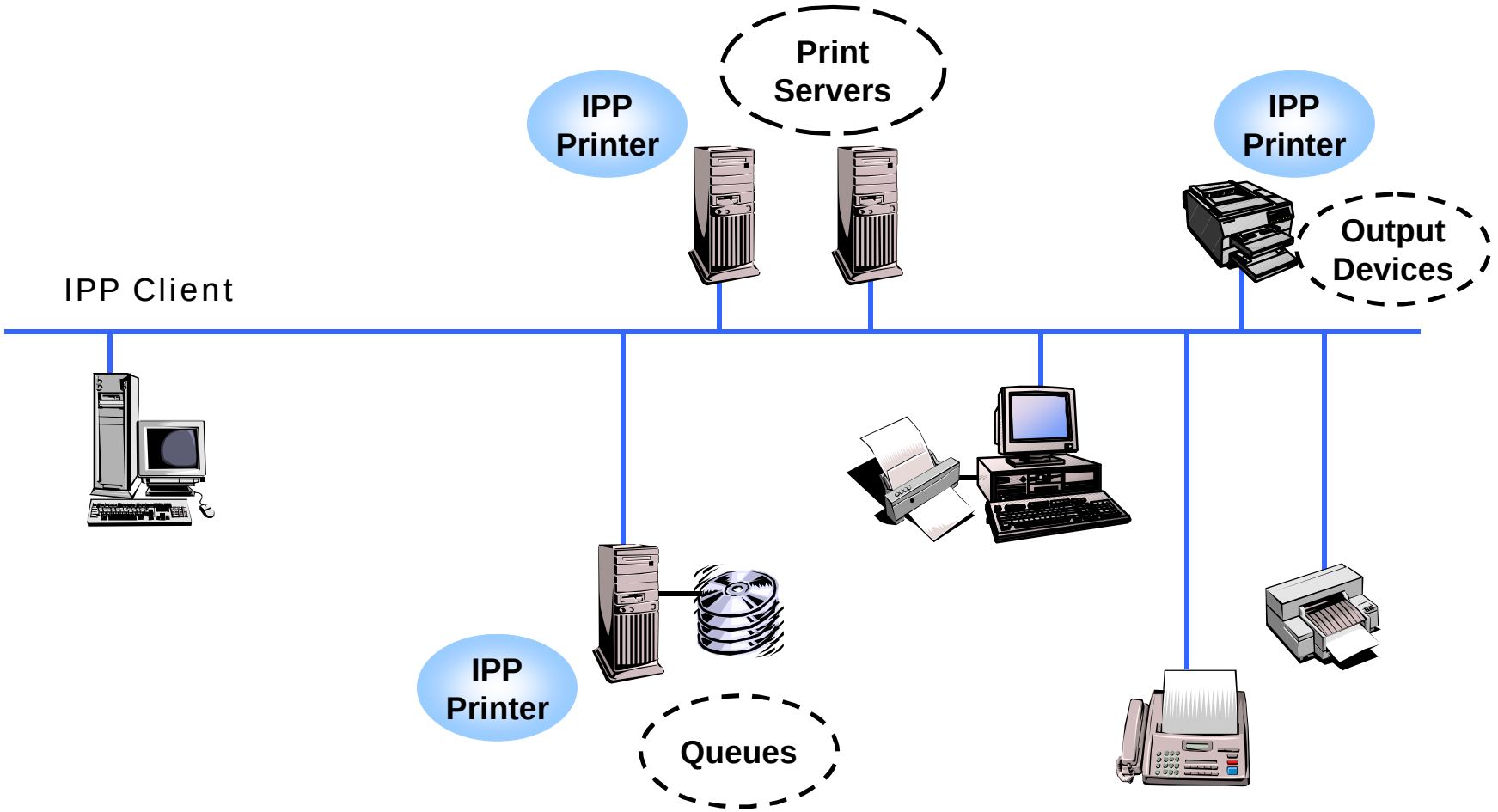
IPP Objects

- Printer
  - Abstraction of logical or physical device
  - Any type of marking or “publishing” device
- Job
  - Descriptive attributes (name, PDL, message)
  - Processing attributes (copies, finishings)
- Document
  - Optionally multiple documents per job

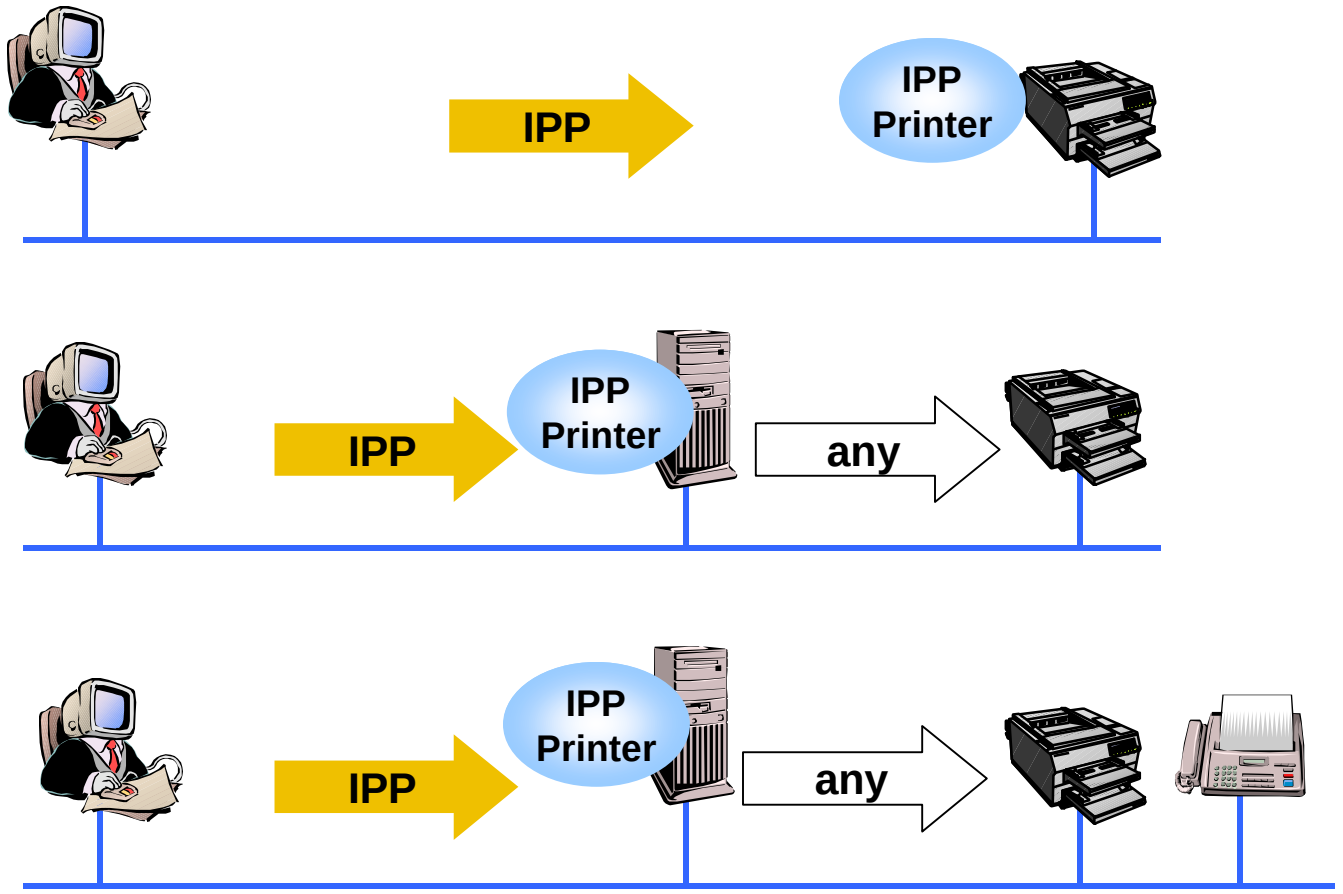
IPP Printer

- Implements IPP
- Logical or physical device
- Multiple configurations
  - server
  - embedded
- Job processing
  - spooling, scheduling, status

IPP Printer Implementations



IPP Printer Configurations



Printer Attributes

- Mandatory
  - URI
  - Name
  - State
  - Accepting jobs
  - Default languages
  - Languages supported

Printer Operations

- Get-Operations
  - Responds with a list of supported operations
- Print-Job
  - Submits a Job, “pushes” job data to the Printer
- Validate-Job
  - Validates client supplied attributes (no job data)
- Get-Jobs
  - Lists jobs at the Printer
- Get-Attributes
  - Responds with supported attributes

Job Attributes

- Mandatory
  - Job Identifier
  - Job Owner's Identity
  - Job State

Job Operations

- Cancel-Job
  - End User initiated abort
- Get-Attributes
  - Responds with current attribute values

Document Attributes

- Name
- Format (PDL)
  - MIME types
    - “application/postscript”
    - “text/html”
  - Printer MIB “enums”
    - PCL, PJL, PS, IPDS, EscapeP, Interpress, etc.

Multiple Document Jobs

- Create-Job
  - Creates a Job object
  - Validates attributes
  - Multiple “Send-Document”
- Optional Operations
- Existing practice

Print by Reference

- Print-URI operation
  - Client supplies attributes
  - Client supplies a URI reference to document data
  - Printer “pulls” document data
- Optional
- Multiple Document
  - “Send-URI”

Job Template Attributes

- Printer supports some values
  - Can be queried
  - Printer has a default value
- Client requests a certain value
- Example: priority
  - Printer supports: 1-50 (model defines: 1-100)
  - Printer default: 20
  - Client requests: 50

Job Template (cont.)

- Cover and separator sheets
- Events
- Priority
- Hold
- Media
- Number UP
- Sides
- Copies
- Resolution
- Quality
- Document Format (PDLs)
- Compression

Fidelity Printing

- Printer supports: cover, bind, punch
- Client requests: staple
- Client semantics - “fidelity” attribute
  - “I expect the job exactly as specified - don’t print it if you can’t do it” (TRUE)
  - “Just print the job the best you can - ignore or make substitutions as needed” (FALSE)

IPP Override of PDL

- Client supplies Job Template attributes
  - Affect rendering, production, and finishing
- Similar statements in the document data
  - Embedded PDL commands
- Submit a “A4” job to a “letter” printer
- Printer’s “PDL Override” attribute
  - attempted
  - not-attempted

Optional Printer Attributes

- Location
- More Info
  - HTML Page
  - Site specific
- Description
- Make and Model
- Driver Installer
- Color Printing Supported
- PDL Override
- State Reasons
- Job Count
- Privacy Supported
- Security Supported

Optional Job Attributes

- More Info
  - HTML page
- Language
- State reasons
- Output device assigned
- Size
  - Octets, impressions
- Time submitted
- Time since
  - Pending
  - Processing
  - Completed
- Number of intervening jobs

IPP Operations

- Request/Response
- Operation attributes
  - Print-Job: Job Template attributes
  - Cancel-Job: Message
  - Get-Attributes: Set of attribute names
- Status Codes
  - OK, Server-Error, Client-Error

Extensibility

- Typed sets of keywords and enums
  - Type 1: Update the specification
  - Type 2: Approval of the PWG
  - Type 3: IANA registered
  - Type 4: site-by-site
- Well defined space for
  - Private
  - Experimental

Extensibility (cont.)

- New operations: type 2
- New attributes: type 2
- New syntaxes: type 2
- Existing attributes: varies
  - state: type 1
  - sheets: type 4
  - finishings: type 2

Security

- Do not reinvent the wheel
  - Look to HTTP (basic, digest, and beyond)
  - Look to Transport (TSL) and below (IPSec)
- Assume
  - Authenticated Identities
  - Authorization is possible
- Mandate
  - An “interoperable” subset

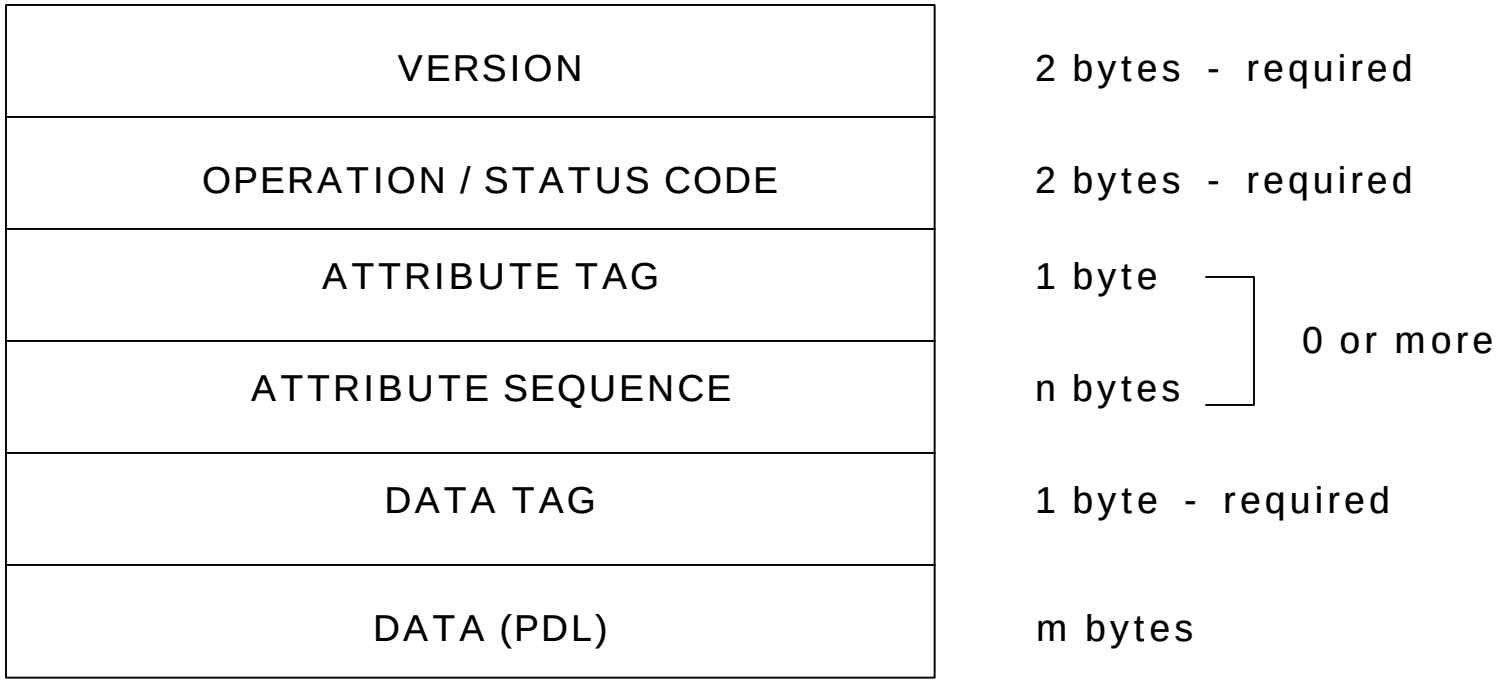
Conformance

- RFC 2119
  - MUST
    - synonyms: SHALL, REQUIRED
    - negation: MUST NOT (SHALL NOT)
  - SHOULD
    - synonyms: RECOMMENDED
    - negation: SHOULD NOT
  - MAY
    - synonyms: OPTIONAL

IPP Protocol Specification

- Rationale: Encoding
  - Simple, regular, binary
  - Embedded solutions
  - “application/ipp”
- Rationale: HTTP/1.1
  - Ubiquitous (HTTP/1.0 still possible although not optimal)
  - Leverage features (URI naming, chuncking, etc.)
  - Printing has already embraced HTTP servers (administration)
  - Focus on simplicity (complexity relates to proxy servers)
  - Security

Encoding Diagram: Operations



Encoding Diagram: Attributes

VALUE TAG	1 byte
NAME LENGTH (value is u)	2 bytes
NAME	u bytes
VALUE LENGTH (value is v)	1 byte
VALUE	v bytes

Augmented BNF

```
ipp-message = ipp-request / ipp-response
ipp-request = version operation
          [parameter-tag parameter-sequence ]
         *(attribute-tag  attribute-sequence) data-tag data
ipp-response = version status-code
          [parameter-tag parameter-sequence ]
         *(attribute-tag attribute-sequence)  data-tag  data

version = major-version minor-version
major-version = SIGNED-BYTE  ; initially %d1
minor-version = SIGNED-BYTE  ; initially %d0

operation = SIGNED-SHORT    ; mapping from model defined below
status-code = SIGNED-SHORT  ; mapping from model defined below
```

Augmented BNF (cont.)

```
parameter-sequence = *compound-parameter
attribute-sequence = *compound-attribute
compound-parameter = parameter *additional-values
compound-attribute = attribute *additional-values

parameter = value-tag name-length name value-length value
attribute = value-tag name-length name value-length value
additional-values = value-tag zero-name-length value-length value

name-length = SIGNED-SHORT    ; number of octets of 'name'
name = LALPHA *( LALPHA / DIGIT / "-" / "_" / "." )
value-length = SIGNED-SHORT  ; number of octets of 'value'
value = OCTET-STRING

data = OCTET-STRING
```

Augmented BNF (cont.)

```
zero-name-length = %x00.00        ; name-length of 0
parameter-tag = %x01             ; tag of 1
attribute-tag = %x02          ; tag of 2
data-tag = %x03                   ; tag of 3
value-tag = %x10-FF

SIGNED-BYTE = BYTE
SIGNED-SHORT = 2BYTE
DIGIT = %x30-39    ;  "0" to "9"
LALPHA = %x61-7A   ;  "a" to "z"
BYTE = %x00-FF
OCTET-STRING = *BYTE
```

Value Encoding

Syntax of Attribute Value	Encoding
text	an octet string where each character is a member of the UCS-2 coded character set and is encoded using UTF-8. The text is encoded in "network byte order" with the first character in the text (according to reading order) being the first character in the encoding.
name	same as text
language	same as text but with a syntax specified by RFC 1766
keyword	same as text. Allowed text values are defined in the IPP model document
uri	same as text
uriScheme	same as text
boolean	one binary octet where 0x00 is 'false' and 0x01 is 'true'

Value Encoding (cont.)

Syntax of Attribute Value	Encoding
integer	a SIGNED-INTEGER, defined previously as a signed integer using two's-complement binary encoding in four octets with big-endian format (also known as "network order"
enum	same as integer. Allowed integer values are defined in the IPP model document
dateTime	eleven octets whose contents are defined by "DateAndTime" in RFC 1903.
resolution	nine octets consisting of 2 SIGNED-INTEGERs followed by a SIGNED-BYTE. The values are the same as those specified in draft-ietf-printmib-mib-info-02.txt [30].
1setOf X	encoding according to the rules for an attribute with more than more value.
rangeOf X	same 1setOf X where the number of values is 2.

HTTP Header Usage: General

General-Header	Request		Response		Values and Conditions
	Client	**Server**	**Server**	**Client**	
Cache-Control	must	not	must	not	“no-cache” only
Connection	must-if	must	must-if	must	“close” only. Both client and server SHOULD keep a connection for the duration of a sequence of operations. The client and server MUST include this header for the last operation in such a sequence.
Date	may	may	must	may	per RFC 1123 [9]
Pragma`	must	not	must	not	“no-cache” only
Transfer-Encoding	must-if	must	must-if	must	“chunked” only . Header MUST be present if Content-Length is absent.
Upgrade	not	not	not	not	
Via	not	not	not	not	

HTTP Header Usage: Request Headers

- Should I add a slide for these headers?
  - General?
  - Request?
  - Response?
  - Entity?