HTTP: how we got here and where we should go

Larry Masinter
(former chair, HTTP-WG)
HTTP 1.1: how we got here

• HTTP 1.0 original requirements: one transaction per document
• Oops: IMG tags for images
  – (compound document? What’s that?)
• Oops: proxy? Cache? What’s that?
• Protocol wars: competing interests
  – 4 connections! 8! 12!
• Theory vs. Reality in Protocol design
  – Theory: optimize for performance & reliability
  – Reality: competing interests simultaneously optimizing for different things
    • Clients: browse performance, privacy
    • Middle network admins: use of network & facilities
    • Origin servers: marketing information
    • Hackers: your private data & account information
    • Application developers: reuse of HTTP stack for other than HyperText Transfer
HTTP was already widely deployed well before RFC 2616

• HTTP/1.1 was difficult to introduce
  – Was hard to require *any* changes
• Don’t imagine you can fix HTTP *now*
  – Couldn’t manage that a long time ago
Some non-goals

- **Don’t try to** help naïve readers understand the spec.
  - Clean up is fine if it helps you get the important tasks done.
  - It’s not a textbook or a tutorial.

- **Don’t try to** make HTTP a better protocol.
  - Great idea, just not this working group!
- **Don’t try to** help HTTP support other applications.
  - Printing, method invocation, streaming video, controlling coffee pots
  - There are other protocols
- **Don’t try to** change the behavior of current implementations or implementors.
  - *They probably won’t change: widely deployed* means something
  - Certainly they won’t change because someone adds a “MUST” to a spec

- **Don’t try to** put messes back into the bottle. Don’t...
  - pick winners when different interpretations are widely deployed
  - specify response to non-compliant behavior:
    MUST do A, but if not, MUST do B, but if not, MUST .... –never ends
So what’s the point?

• Keep new implementations from making things worse!

• *Focus on places where the implementing the spec as written causes things to break*

• Describe what *is*, not what *should have been*
Interoperability Testing

• “multiple independent interoperable implementation of every feature”
  – What’s a HTTP feature?
    • Every **MUST**?
    • Every paragraph?
    • Every header?

• Is it clear how to test interoperability?
  • Clarify places where testing is hard to figure out.
Progressing to Standard

• RFC 2616 is **Draft** Standard  
  – But HTTP is more widely deployed than many Standard protocols

• Don’t get hung up in IETF process details  
  • Down-references, timing on introduction of changes

• Focus (here) on real barriers:  
  – Remove broken stuff  
  – Document interoperability & widespread deployment