The Waka Protocol

Roy T. Fielding, Ph.D
Principal Scientist, Adobe Systems
Director, The Apache Software Foundation

© 2012 Roy T. Fielding
Waka

A new protocol designed to match the efficiency of well-designed Web Applications

Why “waka”?

— Mäori word (pronounced “wah-kah”) for the outrigger canoes used to travel safely on the Pacific Ocean, across hundreds of islands, to Aotearoa (New Zealand)

— Also, one of the few four-letter words suitable for a protocol name

Deployable within an HTTP connection

— via the HTTP/1.1 Upgrade header field
— defined mapping to HTTP/1.1 for proxies
The role of HTTP in Web Architecture

- Extend uniform interface across the net
- Minimize user-perceived latency
- Enable layered processing
- Enable caching
- Enable extension and evolution

Already survived two decades of evolution

- 1991-93: HTTP/0.9 [Berners-Lee]
- 1993-97: HTTP/1.0 [RFC 1945]
- 1996-now: HTTP/1.1 [RFC 2068/2616/HTTPbis]
HTTP Syntax

GET /Test/hello.html HTTP/1.1
Host: kiwi.ics.uci.edu:8080
Accept: text/html, text/*, */*
User-Agent: GET/7 libwww-perl/5.40

HTTP/1.1 200 OK
Date: Thu, 09 Mar 2000 15:40:09 GMT
Server: Apache/1.3.12
Content-Type: text/html
Content-Language: en
Transfer-Encoding: chunked
Etag: “a797cd-465af”
Cache-control: max-age=3600
Vary: Accept-Language

4090

<HTML><HEAD>
...

4090

typical headers
325-400B +
cookies =
1079 Bytes

300-400B + set-
cookies =
3487 Bytes
Issues: Wasted Syntax

GET /Test/hello.html HTTP/1.1\r\nHost: kiwi.ics.uci.edu:8080\r\nAccept: text/html, text/*, */*\r\nUser-Agent: GET/7 libwww-perl/5.40\r\n\r\nHTTP/1.1 200 OK\r\nDate: Thu, 09 Mar 2000 15:40:09 GMT\r\nServer: Apache/1.3.12\r\nContent-Type: text/html\r\nContent-Language: en\r\nTransfer-Encoding: chunked\r\nEtag: “a797cd-465af”\r\nCache-control: max-age=3600\r\nVary: Accept-Language\r\n\r\n4090\r\n
<HTML><HEAD>
...

content negotiation is a waste of bits

Dates need 8 bytes max

Useless product advertising

Header names need 1-2 bytes

Mostly impacts low-power and bandwidth-limited devices
Issues: Not Self-descriptive

GET /Test/hello.html HTTP/1.1\r\nHost: kiwi.ics.uci.edu:8080\r\nAccept: text/html, text/*, */*\r\nUser-Agent: GET/7 libwww-perl/5.40\r\n\r\nHTTP/1.1 200 OK\r\nDate: Thu, 09 Mar 2000 15:40:09 GMT\r\nServer: Apache/1.3.12\r\nContent-Type: text/html\r\nContent-Language: en\r\nTransfer-Encoding: chunked\r\nEtag: “a797cd-465af”\r\nCache-control: max-age=3600\r\nVary: Accept-Language\r\n\r\n4090\r\n
<HTML><HEAD> ...

Extensions cannot indicate scope or mandate
Issues: Head-of-Line Blocking

Message Ordering

- Pipelining depends on pairing requests to responses
  A slow response delays all later requests
  Servers can’t send unsolicited event notifications

Envelope Ordering

- Control data must be sent first
  Server must indicate success before it is actually successful

- Metadata must be sent before Data
  Low-priority metadata is excluded for performance reasons
  Data cannot be sent until all filters supply metadata
  Dynamically generated metadata is lost

- Data must be entirely delivered
  No signal for abnormal termination
  Limited support for small-memory devices (Range requests)

- Control data cannot be updated to reflect events
  What if the sender encounters a time-out condition?
  What if an intermediary is caught in the middle of a bad stream?
A replacement for HTTP (under development)

- Token-based, length-delimited syntax*
  considering changing this to a derivative of msgpack
- Self-descriptive messages

Interleaved message (meta)data packets:

- Up to 64 channels per connection
- Up to 63 payload streams per message

Complete transport independence

- TCP, UDP, SCTP, TLS, multicast, ...
New Request Semantics

Multiple request targets (GET many subrequests)

Request control data
- request/transaction identifier
- relative priority (high, low, HiLo)
- explicit indication of context (main, embed, js, test)

Methods
- RENDER for display/print/speak this representation
- MONITOR for notify me when resource state changes

Authoring methods (DAV simplified)
- elimination of non-resource identifiers
- reintroduction of PATCH
New Response Semantics

Self-descriptive binding to the request

- Echo of request id, method, target URI
- Cache key explicitly described
  Caches no longer need to save request fields
  Caches don't have to guess about Vary info
- Enables asynchronous transport

Response indicates authoritative or not

- Semantics formerly in status code

Unsolicited Responses

- Cache invalidation messages
- Multicast event notices

© 2012 Roy T. Fielding
Waka Syntax

Uniform syntax
- Regardless of message type, direction
- Padding allowed for 32/64bit alignment

Self-descriptive
- Explicit typing for message structure, fields
- Indication of mandate and scope of fields
- Association of metadata (control, resource, rep.)
- Premature termination of request or response

Efficient and Extensible
- Tokens for all standard elements
- A URI reference can be used in place of any token
- Macros (client-defined syntax short-hand)*
- Interleaved data and metadata delivery
Future Plans

1. Finish HTTPbis
2. Finish drafting Tracking Protection
3. Write a specification for Waka
4. Decide whether to submit it here.