Internet Engineering

241-461

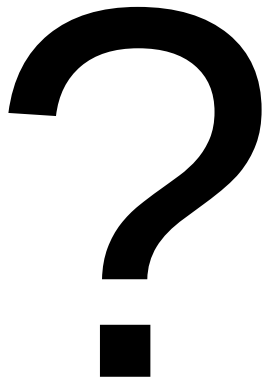
Robert Elz

kre@munnari.OZ.AU
kre@coe.psu.ac.th

http://fivedots.coe.psu.ac.th/~kre

Contents

Application Protocol Requirements



Application Protocols

- All the work that seems useful
  - transport just moves data
- Determining
  - what data to transfer
  - how to transfer it
  - from where
- Usually Client / Server
  - Server
    - continually running
    - takes requests
    - processes & returns replies
  - Client
    - dynamic - runs when needed
    - initiates connections
    - decides what needs to be done

Client / Server

- Client/Server
  - Refers to role in a particular connection
  - Sometimes to particular processes
  - Less accurately to systems running processes
- Client
  - The system that initiates the connection
- Server
  - The system that receives the connection
- Client Today, Server Tomorrow
  - Or in 5 minutes
- Client for one application
  - Server for Another
- Client & Server for same application
  - At the same time
  - Different connections

The Web

- Large distributed data collection
  - With links between elements
- Thousands of web servers
  - Provide access to data
    - web pages
    - content
- Millions of web clients
    - browsers
    - & others
  - Retrieve web pages from servers
  - Display to user
- HyperText Markup Language HTML
  - Defines web content
  - Allows browser to layout web pages
  - Provides links to other pages
    - Outside scope of this course

Browsing the Web

- Need to identify web location
  - Uniform Resource Locator
    - Identifier of a resource
    - A web page
      - Or part of a web page
      - Or almost anything
    - Provides the address of the page
  - Also
    - URI - Uniform Resource Identifier
    - URN - Uniform Resource Name
- Uniform
  - Common, Standard
  - The same for everything
- Resource
  - Anything needed to be identified
  - Anything that may need to be accessed
- Locator
  - Address - provides location information

Browsing the Web (2)

- Browser needs to
  - fetch data identified by URL
  - build data into web page
  - display
- Only data fetch relevant here
  - need a protocol to fetch data
- Several protocols exist
  - FTP
    - complicated
  - TFTP
    - too simple & restricted
  - ...
- HTTP
  - new file transfer protocol
  - suited to URL fetch

HTTP Model

- HyperText Transfer Protocol
  - Protocol for transferring HyperText
    - text containing links
  - Originally HyperText
    - Now transfers anything
- Operation
  - Client connects to server
  - Client requests a web page from server
  - Server sends web page
    - plus some status information
  - Connection closed
    - very simple protocol
- Issues
  - How to request file ?
  - How to receive file & status ?
  - How to discover type of file ?

HTTP

- Application Protocols
  - Text based
    - commands & replies are all words
  - Binary
    - commands & replies are numbers (not digits)
- HTTP is text based
  - Easy to debug
    - Can connect to server and type
      - And read responses
  - Easy to extend
    - Define meanings for new words
  - Compatible with other protocols
    - FTP, SMTP, ... all text based

HTTP(2)

- Start with URL
  - protocol://host-name/path
    - http://fivedots.coe.psu.ac.th/~kre
    - Protocol: http
    - Host-name: fivedots.coe.psu.ac.th
    - Path: ~kre
- If protocol is not http
  - go elsewhere
    - plenty of other valid protocols
  - here we consider http only
- Connect to host-name
  - Use DNS to translate host-name to address
  - Connect to standard http port (80) at address
- Send
  - GET path
    - Plus some other information

HTTP(3)

- Send
  - GET path
- Server responds with
  - Status Information
  - The file data
    - contents of the file at the path requested
    - HTTP object
  - Terminate Connection
    - Indicates end-of-file

Request Format

- For each request
  - Send command path HTTP-version
  - Then send header
    - 0 or more header lines
  - Then send blank line
    - Request complete
- Commands
  - GET
    - fetch an object
  - POST
    - send an object
      - object follows headers
  - HEAD
    - Get just header information
      - No object content included

Request Format (2)

- Header Lines
  - Derived from e-mail headers
    - Similar syntax
  - Host: host-name
    - Host name for path
    - Identifies which virtual server
  - User-agent: browser name and version
    - Allows response suitable for browser
  - Connection: close
    - Should connection be closed after each file

Reply Format

- Information to send
  - Did request succeed?
    - Yes, or No and why not
  - Type of data
    - text/html
    - image/jpeg
      - MIME data types
      - Multipurpose Internet Mail Extensions
  - Last Modification Time
    - Client can tell if any data is new
    - Or whether nothing has altered
  - Size of data
    - Byte count of data to be transmitted
  - (and more)

Reply Format (2)

- Request status
  - 3 digit code
    - First digit indicates overall status
      - 1 this is not a response
      - 2 success
      - 3 need more information
      - 4 Error in Request
        - No such object
        - Syntax errors
      - 5 Not supported
        - Requested something server unable to handle
    - Later digits make error precise.
  - Explanation
    - Human readable explanation of error