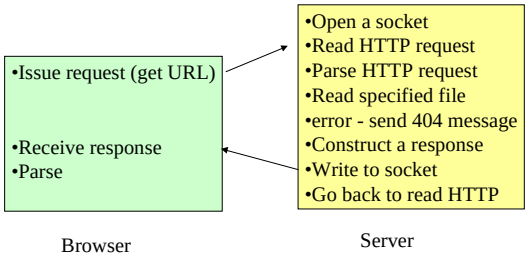
Web Engineering and XML Processing

- HTTP Details
  - –HTTP Server Design
  - –HTTP/1.1
- Web performance measurements
- Web workload characteriszation
- XML Processing

Amjad Umar

HTTP Server Design



Specification: www.ietf.org
http://developer.java.sun.com/developer/codesamples/examplets/java.net

HTTP 1.1 Highlights

- Many issues with HTTP 1.0
  - – Depletion of IP addresses
  - – Inefficiencies of using TCP
  - – Statelessness
  - – Security - sending passwords in clear text
  - – Dealing with proxies

HTTP 1.1 Key Concepts

- Virtual web hosting (Internet Address Conservation)
  - suppose you (zombie) want to run a web site
  - Your web hoster (www.graveyard.com) can give you a site:
    - www.graveyard.com/zombie.html (put zombie.html in server directory)
  - Suppose you insist on your own URL (www.zombie.com)
  - HTTP 1.0 problem:
    - URL: www.graveyard.com/zombie.html is translated to
      GET zombie.html HTTP 1.0 (server name stripped)
      cannot put more than one server on same IP address
    - Had to get multiple IP addresses (one per server) on the same machine.
    - DNS found the IP address, Operating system matched IP to server
  - HTTP 1.1 Solution: retain host name
    GET zombie.html HTTP 1.0
    Host: www.graveyard.com
  - A Web host can install multiple servers on same IP address
    www.graveyard.com
    www.zombie.com

HTTP 1.1 (cont.)

- Hop by hop mechanism
  - HTTP 1.0 used same techniques (e.g., compression) between end-points
  - HTTP 1.1 allows different techniques between intermediaries (can use different compression with a local proxy)
  - implemented through connection header (connection header1)
  - Impact on traffic between end nodes
- Headers, messages and Transfer coding
  - Many new formats and refinements, new error codes

HTTP 1.1 (cont.)

- Caching - many options to control
  - control requests and responses
  - no-cache, only-if-cached (access cached only), max age (do not use older than this), etc
  - Entity tag (ETAG): compare cached with new
  - Others
- Proxies in HTTP 1.1
  - Formally recognized
  - Conversion of HTTP 1.0 and HTTP 1.1 messages
  - Can add additional information (e.g., ETAG)

HTTP 1.1

- Bandwidth optimization: reduce workload
  - minimize resends (deltas)
  - do not send at all if receiver cannot handle it
  - transform (sophisticated compression)
  - RANGE (e.g., location of PDF) pages)
  - Expect/continue: ask server what to expect
- Connection Management-

  GET /home.html html 1.0

  Connection: Keep-alive (keep connections alive)
  - Many proposals in HTTP 1.1
  - Persistent connections
  - Pipelining (multiple requests)
- Message transmission: ensure receipt safely
  - HTTP 1.0 used message length, close for dynamic content
  - HTTP 1.1 uses "chunking" to send long messages (break into chunks, send a zero length to indicate end)

HTTP/TCP Interactions

- Several performance implications
- TCP uses timers heavily for retransmission of lost packages - how it impacts HTTP
- HTTP traffic on TCP - some implications
- Handling multiple connections between clients and servers (e.g., download of images)
- How to handle large no of requests for web servers

Web Performance, Workload Measurement

- Simple Web Performance Model
- Performance Measurement
- Workload Characterization
- Web Mining

Simple Web Performance Model

. Performance means different things: we assume response time

. Response time per request = sum of all service times

S = s1 + s2 + s3 + ...

where s = service time, I/O time, transmission time,,

. Example: file transfer (download) from computer C1 (web server) to C2 (your machine) :

S = ∑ s1 + s2 + s3

s1=read time, s2=transmission time, s3=write time

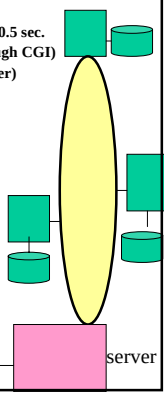
Simple performance analysis (Best Case)

Assume that there is no queuing (lower bound) or
assume that service time includes all queuing

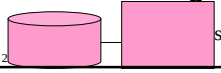
Example 1: 12 node network, 1 Mbps average data rate, disk i/o = 0.5 sec.
- one file server (web server with a customer file accessed through CGI)
- assume that users at each workstation issue (through a browser)
  1 request per minute and each request requires
  60 accesses of local file, 20 of server files
- each remote message is 100 bytes, 10 bits per byte

Response time per transaction =
- s1: origin node = 60 x 0.5 = 30 sec
- s2: server node (without queuing) = 20 x 0.5 = 10 sec
- s3: transmission proc = 20 x 100 x 10 /1000000 = 0.02 sec

. resp. time = 30 + 10 + .02 = 40 secs



Bottleneck =?



Queuing Analysis

Queuing causes waits, increases service time

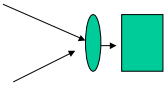
A = arrival rate

S = service time

U = utilization = A x S

To avoid queuing, U should be below 0.5

Q = no. of people waiting = U / 1-U = A.S / 1-A.S



If U = .1 Q =

U = .5 Q =

U = .8 Q =

U = .9 Q =

To reduce queuing, reduce U

Example 2: same as example 1, include queuing

. need to calculate queuing at server

. arrival rate A at server = 12x 20 = 240/min = 240/60 = 4 per sec

. service time S at server = 0.5 sec

. server utilization = U = 4 x 0.5 = 2.0

serious trouble, server queues can be infinite

Example 3: workstations on LAN

. One ethernet LAN (10 MBPS)

. Each workstation generates 1 message per second, each message is 1000 bytes long (about a screen)

. How many workstations can be supported on this LAN

. Solution: Assume 10 bits per byte for communications

S = service time per message=1000x10/10,000,000=.001 sec

A = arrival rate = 1 per second for one workstation

U = A x S = 1 x .001 = .001 (Virtually no queuing)

For 100 workstations with similar traffic, A = 1 x 100

U = 0.1

For 100 workstation with color graphic data

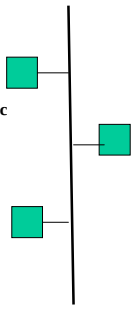
Message = 40 times bigger

S = 40 x 10000 /10,000,000 = .04 sec

A = 100 per second

U = 100 x .04 = 4 (too high, forget it)

For 10 workstations: U = 0.4 (still may be too high)



Multimedia Performance Analysis

Large screen sizes, with images and sound sent across the network very rapidly

Example:

Consider a high definition large screen 1024x1024 = 1 million bits

Very fancy, extremely large colors= 2Million bits

Moving video at 30 screens per second

Traffic sent per second=1 million bits x 24 x30 =720 Mbps

Cannot be handled by fast networks (e.g., Fast Ethernet or FDDI LAN)

Tradeoffs:

- Use compression (can be a by a factor of 50)
- Reduce the number of colors to 10 bits
- Reduce moving video ro 20 per second
- Reduce screen size and ressolution
- Carry only differences in images
- Many multimedia systems at present operate at 1 to 1.5 Mbps per user

Performance Measurements

- Key things to measure:
  - Arrival rate A
  - Service time S
  - Others?
- Motivation for measurement (why measure)
  - content creators
  - web hosting
  - network operators
  - web/network researchers
  - others?
- Sources of measurement
  - Server logs: mostly default, mostly request header, coarse grain (time is coarse), difficult to assign request with users (proxies, dynamic addresses, etc)
  - Proxy log: similar to server logs - common log format
  - client logs: can be very detailed, no common log format
  - TCP/IP logs: routers keep track of information, can be very detailed

- Issues:
  - Granularity (too coarse, e.g. time in seconds)
  - Data you need absent (e.g., no service time)
  - Encrypted data difficult to log
  - Details versus performance
- May need to capture own data
- Common log format (CLF): used in servers and proxies. Common fields are:
  - remote host - client IP address
  - remote identity - client application
  - authenticated user - user name
  - Time - time request received (roughly)
  - Response code - HTTP response code
  - Content length - response length
- Processing logs and drawing inferences from logs
  - connecting client/server, network logs to build a complete behaviour model
  - Manu research efforts (universities)

Workload Characterization

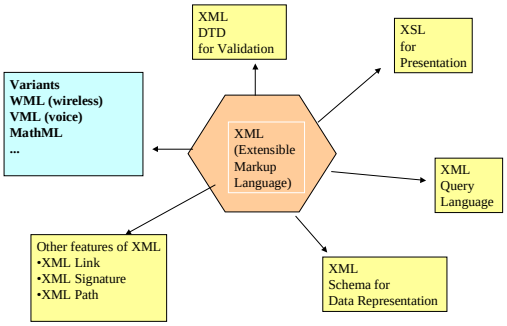
- Workload shows what needs to be serviced (arrival rates, service times, etc)
- Important to characterize workload correctly
  - what is the real workload (worst case, best case, average case)
  - Benchmarks - typical profiles of users
- Workloads used in performance and simulation models to:
  - determine how many servers needed to handle the workload
  - evaluate new servers and or proxies
  - Evaluate protocol efficiency
- Workloads for Web mining (clickstream mining) is an active area of work

Web mining: develop patterns from usage

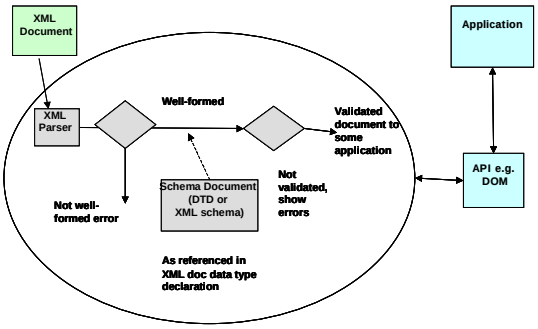
A web-site is an interconnected network for presenting information where each web page is a node. Users navigate the nodes to discover information in ways that can be patterned and grouped. The assumption is that the identified patterns reveal useful and usable information about user preferences and motivations.

- Web-site content
- Web-site design
- Web-page content
- Web-page design

XML Family



XML Processing



Handling XML Data

XML data can be handled at present in two different ways:

a). Use relational database to store XML data
- •Translate the XML schema to relational
- •Convert RDBMS data to XML (supported by many RDBMSs)

b). Use Native XML databases
- Store data in XML databases
- Use XML Query languages to access XML data

Tradeoffs between the two choices for XML data