COSC1300, COSC1301

Web Servers and Web Technology

Lecture 11: Advanced Research Topics
Lecture Overview

Here we cover some further advanced research topics relevant to WSWT to give you a flavour of what research in this area is all about.

We will cover:

- The Mobile Context
- Content Distribution Networks
- Distributed Caching and Cache Invalidation
- Fragment Caching (AJAX)
- Data Hoarding

Beyond WSWT?

- An overview of where to go further for those interested in this topic.
The Mobile Context
(Connectivity)

A mobile device context has the following properties:

- Minimise the *number* of packets (connections), *packet size* is less important
  - cheaper to send one 300kb packet than ten 30kb packets

- Asymmetric power usage in a mobile context
  - important for transmission but not for reception
  - keep #transmissions down

- Not always connected,
  - *always-on* is not feasible

- But, a mobile user is almost always reachable
  - *Cache invalidation reports vs cache propagation.*

- Objective: Reduce network latency
• In a mobile-mesh scenario, where all devices share one or a few base stations, there is significant cost involved if a device changes base-station to one not in the existing set.
  — Important for session management

• Some mobile communication is proximity-based, whether by sharing base stations or the devices are physically close to each other.
  — This is *most simply* modelled as two nodes in a network suddenly communicating - prompted only by influences outside of the network. *This is not always correct, tough.*

• Server broadcasting is *much* cheaper than 1:1 client:server communication.
• Typical mobile browsing behaviour is different to typical desktop browsing behaviour.
  — More application-targeted and involve fewer different web sites than typical desktop usage, but involve this subset of web sites more frequently.
  — This subset often contains material targeted specifically to mobile devices.
  — Content is more likely to be dynamic rather than static (caching implications are...?).

• Mobile devices are more secure
  — Audit trails, tracing is feasible
• Micro-payments are feasible, secure and affordable in a mobile-phone context, but not so much in a desktop environment.
  — Can significantly affect website topology and design (think of downloadable ring-tones – rarely a single click).

• Mobile business model is different from internet
  — Telco plays non-trivial role in influencing user behaviour
    ▶ through rate plans, internet access metering rules, 'walled garden' preferred services (market distortion)
    ▶ Influence mobile device purchase behaviour and (indirectly) device properties.
  — Slowly moving to more internet-style behaviour
    ▶ through Iphone and Gphone independent purchasing
• Content Distribution Networks (CDNs) are:
  — special-purpose networks that provide scalability by distributing many servers across the Internet close to consumers.
  — a *coordinated* caching systems.

• Consumers obtain content from *edge servers* directly rather than from the *origin server*.
  — A typical content delivery service uses multiple Web servers distributed across the world to deliver copies of your Web site content.
Content Distribution Networks

Content Delivery Network overview.
• Objectives:
  — move content closer to the clients
  — Reduce response time for client request.
  — Reduce traffic on an institution’s access link.
  — Internet dense with caches enables “poor” content providers to effectively deliver content
  — avoid server bottlenecks

• CDNs are a multi-million-dollar business already.
• The content providers are the CDN customers.
• Content replication
• CDN company installs hundreds of CDN servers throughout Internet
  — in lower-tier ISPs, close to users
• CDN replicates its customers’ content in CDN servers.
• When provider updates content, CDN updates servers
A “Distributed Database”: is a logically interrelated collection of shared data (and a description of this data), physically distributed over a computer network.

A “Distributed DBMS” (DDBMS): is a Software system that permits the management of the distributed database and makes the distribution transparent to users.

**Fundamental Principle:** make distribution transparent to user.

The fact that fragments are stored on different computers is hidden from the users.
Techniques for CDN

• **IP Anycast:**
  - Route an IP packet to one-of-many IP addresses
  - Some research but not deployed or supported by IPV4

• **HTTP Redirection:**
  - Server must be aware of CDN network
  - 302 response from Web server should always include an alternative URL to which redirection should occur. If it does, a Web browser will immediately retry the alternative URL
• DNS Redirection:
  — When client asks for server IP address, tell them based on where they are in the network
  — Used by most CDN providers (e.g., Akamai)

• TCP Redirection:
  — Client TCP packets go to one machine, but responses come from a different one
Managing the update process between origin and surrogate servers can involve:

- ICAP
  - Invisible HTTP wrapper protocols
- ESI
  - Telling the server which parts to cache and how
- OPES
  - Optimally combining these two methods
The ICAP Protocol

• Aimed at intermediate servers
• Handles server to server communications
• Encapsulates HTTP
• Bring intelligence to the edges of network.
• Daisy-chain ICAP servers
  — Distributed Caching!
• ESI: markup language to split web page into parts, each having cache information
  – Edge refer to client-side proxies, server-side reverse proxies, and cache proxies in CDN
  – ESI support Inclusion, Variable support, Conditional processing, and Exception and error handling

• We add tags on our code to split between pages according to some condition.
Eg: Edge Server Includes

```xml
<esi:choose>
  <esi:when test="$(HTTP_COOKIE{country})=='au'">  
    <esi:include src="http://www.xyz.com/au.html"/>
  </esi:when>
  <esi:when test="$(HTTP_COOKIE{country})=='uk'">  
    <esi:include src="http://www.xyz.com/uk.html"/>
  </esi:when>
  <esi:otherwise>  
    <esi:include src="http://www.xyz.com/new.html"/>
  </esi:otherwise>
</esi:choose>
```

- Obviously this requires the content writer to be aware of cache behaviour
  - This may not be desirable or practical
  - The alternative is to automate the process
• Web server will interpret the ESI tags and will send the fragment to the cache server with specific HTTP headers for each fragment to give information about how to handle cache on each fragment.

1. User request for my.xyz.com is directed to optimal Akamai server.
2. EdgeSuite parses the template, which may be cached already, looking for tags and instructions to assemble the page and send individually.
3. If necessary, EdgeSuite calls xyz.com over a persistent TCP connection to obtain new or uncached HTML fragments.
4. xyz.com responds by sending objects to EdgeSuite. Each HTML object has its own tags, response headers, or configuration data.
5. EdgeSuite assembles and delivers page.
• Open Pluggable Edge Services
  – Aimed at the architectural level
  – Provides the overall framework
  – Incorporates ICAP, ESI and other protocols
  – Attempts to address security / integrity concerns

• RFC3835: An Architecture for Open Pluggable Edge Services (OPES)
• RFC3836: Requirements for OPES Callout Protocols
• RFC3837: Security Threats and Risks for OPES
• RFC3838: Policy, Authorization, and Enforcement Requirements
• http://www.ietf.org/rfc/rfc3835.txt ... etc
• Biggest CDN in business
• Around 20,000 CDN servers around the world
• CDN architecture for distributing, storing, and delivering high quality streaming media over IP networks.
• transfer content from a live source to one or more portals
• pass content from a portal to one or more clients.
Prism Architectural Elements

• Live sources
  — receive content from a content provider
  — encode and packetize it
  — stream it into the Prism IP network infrastructure.

• Portals
  — receive multimedia content from live sources and other portals
  — transmit it to Prism clients
  — are positioned between clients and live sources

• Portals can
  — store and archive live content (to be viewed on demand)
  — provide VCR-like functions (fast-forward and rewind).

• Clients
  — receive content from a portal
  — Display it to end users.

• Clients are:
  — networked settop boxes, PCs using broadband access.
• If CDN operation is to be invisible, then caching/replication must happen automatically.
Cache Invalidation Reports

• We replicate, when the origin server content has changed, to each of the surrogates
  – Typically, this involves creating an invalidation report which specifies:
    ▶ which parts of server content has changed
    ▶ which parts, how and where they are to be cached
      ■ locally
      ■ at intermediate server (mobile gateway)
      ■ at intermediate server (online)
      ■ at origin server
  – An IR can be sent in various ways
For locally cached content, IR can be used to flush the cache.

- Eg: Traffic conditions by location
  - Maps updated regularly based on locality
- But for mobiles, what if mobile is not turned on at the time an IR is sent?
  - Cache can be completely flushed, but this may not be practical
  - Cache can be partially flushed, but now we need to know which parts to be flushed.
• Broadcasting Timestamps (TS)
  – The server broadcasts an IR every N seconds
    ▶ IR consists of the current timestamp and a list of data items to be updated
    ▶ During the power off mode might miss some IRs
      ■ Discard all the saved cache and save it again
  – Only effective if the clients have not been disconnected for a period exceeding an algorithm specific parameter
  – Cons
    ▶ Does not handle disconnections in a good manner
    ▶ Even though some of the cached items might be valid the entire cache is discarded
    ▶ Does not utilize the bandwidth effectively
Cache Invalidation Reports
(Mobile Issues)

• Bit Sequence + Time Stamp (BS)
  – Hierarchical structure of binary bit sequences, set of timestamps
    ▶ Each bit represents a data item on the server. 1 means updated.
    ▶ Use bit sequences and the time-stamps to decide what data items
      in their local cache should be invalidated
  – Very flexible and it can deal with the long disconnections
  – BS is an improved version of TS
  – Client uses the bit-sequence to check a validity of an
    updated items (space efficient)
  – Saves on bandwidth utilization
  – Cons:
    ▶ Uses much larger IR and consumes downlink bandwidth
• Grouping with cold update-set retention
  — Cache validity checked after reconnection
  — To reduce the bandwidth requirement for validity checking, data objects are partitioned into groups
  — Instead of group invalidation, invalidates a cold update set of objects which are in a group
  — Improved version of TS
  — Handle disconnections in a good manner
  — Does not use large IRs and downlink bandwidth
  — Cons:
    ▶ Requires uplink bandwidth and an update history window
• Adaptive Cache Invalidation Scheme (ACIS)
  — Hybrid of all the above algorithms
  — Overall performance optimization
    ▶ Packet efficiency
    ▶ Energy efficiency
  — Low uplink cost – save more energy
  — Real solution for the disconnection problem
  — The uplink and downlink channels are more optimally utilised
It is possible for an intelligent caching arrangement to split a single page into static and dynamic components.

- The static components might be:
  - Fixed Headings, company logo, etc
  - Links to regular content,
  - The framework of the whole page

- The dynamic components might be:
  - News items, specials,
  - Database-generated content
  - Syndication from other sources (e.g., YouTube links)
Multi-component Web Pages
(Fragment Caching)
Multi-component Web Pages (Fragment Caching)

• The different parts can be sourced from different web sites
  — The static parts could be
    ▶ sourced close to the user as they rarely change
    ▶ occasionally sync'ed with origin server
  — The dynamic content can
    ▶ be kept centrally
    ▶ regularly sync'ed with local servers
    ▶ be involved in a hybrid of the above

• This requires content writer be aware of caching behaviour
• AJAX = Asynchronous JavaScript and XML
  ► See Ajax "Hello World" at http://www.dynamicajax.com/fr/AJAX_Hello_World-.html

• Requests are sent asynchronously (ie without page refresh) using the XMLHttpRequest object
  – The Document Object Model (DOM) interface is then used to merge results from AJAX requests with the currently displayed page

• Well known examples:
  – RMIT's Learning Hub, Google Maps/street-view, Gmail

• May not work well with user-generated content
  – Since user-fragment behaviour can interfere with overall document management (eg CSS misbehaviour)
  – Security issues around cross-site scripting
Fragment Caching

• Pros:
  – Can be highly efficient and fast
  – Only sent updates
    ▶ can exploit task-specific caching policies

• Cons:
  – Non-deterministic invalidation schematics
    ▶ Cannot make general assumptions about update behaviour
    ▶ Operation within CDN requires the CDN to know about task-specific update policies (eg Gmaps)
Data Hoarding

• When a client is disconnected for an extended period of time, it may be good to 'pre-fetch' or hoard the data in advance
  — Predict the data items a client may need prior to disconnection
  — Focus on voluntary disconnection
  — Typically application-oriented as this improves the prediction

• Example:
  Google maps may hoard all map tiles surrounding recent movements
**Manual Hoarding**

- **Pro**
  - Prior knowledge ensures efficient hoarding
  - User has control over what is hoarded

- **Con**
  - Tedious
  - Time-consuming
  - May miss novelty

**Automated Hoarding**

- **Pro**
  - User-friendly
  - Performed in background
  - Picks up new items

- **Con**
  - Assumes consistency and predictability in user behaviour
  - User has less control over what is hoarded
• Summary
  – The main aim of all the methods described is to improve the efficiency with which data is communicated over a data channel
  – Also considered some of the special needs of the mobile environment

• Topics
  – The Mobile Context
  – Content Distribution Networks
  – Distributed Caching and Cache Invalidation
  – Fragment Caching (AJAX)
  – Data Hoarding

• Beyond WSWT?
Beyond WSWT?

If the topics of this course interested you, here is a list of courses that naturally follow the topics in this course, that you may wish to enrol in later.
Further topics in Networking - Clients and Servers

COSC1174/5 Adv Client/Server Architecture

- PG/Hons/DD only
- Design oriented
  - Content Distribution Networks (Beyond this lecture)
  - Scalable Web-Server Systems (Beyond L10)
  - Advanced Performance Analysis of Multi-server Systems (Beyond Lecture 8)
  - Audio, Video Streaming Servers, VOD servers
  - Client Server Arch for Massive Multiplayer Games
  - Distributed Grid/Cluster Systems
  - Distributed Sensor Systems
  - Efficient Server Selection
  - Distributed Peer-Peer Architectures
COSC1178/9 Web Services

- PG/Hons/DD only
- Programming oriented
- XML, WSDL, SOAP, REST
- Middleware
- Web services architecture
- WS-Security
- Service coordination protocols
- Service composition
- UDDI, WS-*
- Service Oriented Architecture (SOA) and Design
- Using SOAP for Web Services
COSC1111 Data Comm & Net-Centric Computing

- Design oriented
  - Internetworking, IPv4, IPv6 (Beyond L1-3)
  - Physical Aspects of Data Communications, Data Encoding
  - Error Detection
  - Multiplexing
  - Flow Control, Error Control, Routing
  - LAN and Medium Access Control (MAC) Methods
  - Wireless Networking
  - Wide Area Networks, Packet / Circuit Switching, ATMs
  - Transport Protocols
  - Emerging Networking Technologies
COSC1195/7 Distributed Systems

- Design oriented
  - Interprocess Communication
  - Remote Procedure Calls
  - Measuring IPC / RPC Invocation Time
  - Distributed File / Operating Services
  - Distributed Operating Systems
  - Name (directory) and Time Services
  - Replication (Beyond this lecture)
  - Transaction Processing
  - CORBA Basics
  - Introduction to SOAP
  - Using SOAP for Distributed (non-web) Services
COSC1233/5 Broadcast Networks & Apps

- Design oriented
  - Intro / History of P2P
  - Network Topologies in P2P
  - Management of distributed systems
  - Distributed Hash Tables (DHT)
  - Load-balancing and reliability in DHTs
  - Implementation of Peer to peer applications
  - Issues in Grid and Web Service Applications.
  - Overlay networks
  - Self-organization and P2P in mobile environment
  - Business Applications
  - Peer-to-Peer Computing Security-related issues
COSC2276/7 Web Development Technologies

- Programming oriented
  - .NET 3.5 Architecture
  - Component Oriented Programming
  - ASP.NET 3.5
  - Introduction to AJAX 3.0
  - Introduction to SQL Server 2005, SQL review and T-SQL (Stored Procedures, View and Triggers)
  - ADO.NET 3.5
  - LINQ (Language Integrated Query)
  - Web Services
Further topics in Networking –
The Sun/Java View

COSC2353/4 Elec Commerce & Enterprise Sys

- Programming oriented
  - eBusiness Models & Enterprise Java Fundamentals
  - Naming and Directory Services and Introduction to Enterprise Java Beans (EJBs)
  - Introduction to EJB, Session beans and Entity Beans
  - Container managed Persistence (CMP) and EJB/QL
  - Servlets
  - Java Server Pages (JSP)
  - Guidelines for developing J2EE applications (I)
  - Enterprise Design Patterns
THE END

Thank You!

Next week: Review