From RPC to Web Apps: Trends in Client-Server Systems

George Coulouris

Digital Technology Group

UNIVERSITY OF CAMBRIDGE
Computer Laboratory

Overview

- Motivation - to consider the effect of client-server interaction on the development of interactive apps
- Style of client-server interaction is driven by the need for user interaction
  - we'll look at some web apps to exemplify this
- A look at the history of client-server systems
  - shared files ⇨ shared objects ⇨ shared data resources
- The current technology
  - Dynamic HTML, XML, Javascript, AJAX, REST or SOAP
- Evaluation and discussion

Style of client-server interaction

- We now see many highly interactive web applications
  - Direct manipulation of application entities
  - MVC for program structure
    - *Model*: application logic and data
      Views: presentation logic
      Controllers: action logic (in response to user input events)
  - The interaction components (*Controllers* and *Views*) potentially operate over the entire application state
- Most interesting applications operate with:
    - very large datasets
    - shared data
  - In those the *Model* is shared between client and server and communication mediates to 'hide' the split, with potential performance, security, consistency, concurrency control and other issues

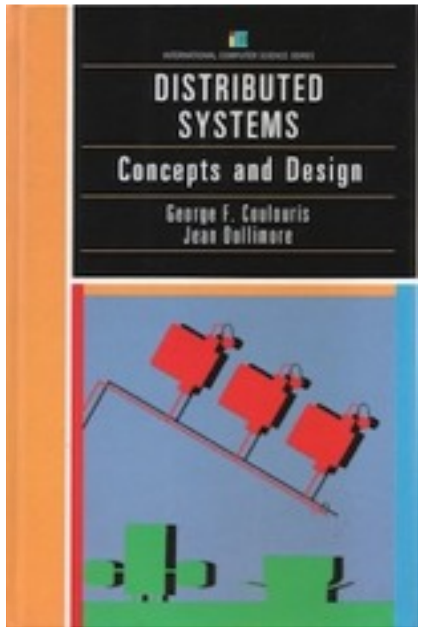
Illustrative web applications

- Not representative, just some concrete examples
- Two of my apps, because I know how they are implemented:
  - A) SVG example: (not available in the PDF slides) SESAME Sensor Visualizer
    - ~2500 lines of JavaScript
  - B) maps.camdencyclists.org.uk (available in the PDF slides)
    - ~5000 lines of JavaScript
- Google Maps
  - C) 'Draw along roads'
    - (Not available in the PDF slides because it requires a Google login)

From RPC to Distributed Objects ...

- 1975 to 1986: RPC-based applications
  - Only shared hardware and shared files (FTP, Telnet, email, print, Sun NFS)
  - No objects or structured data
  - Advanced distributed apps used *ad hoc* RPC between peers
  - Interactive applications got useful structure with Smalltalk (1976 - single computer)
- 1987: Distributed shared objects
  - Research-based distributed object systems (Emerald, Argus, Arjuna)
  - CORBA (v1: 1991, v2: 1996, v3 1999)
  - More system services: name servers, secure key distribution, database, ...

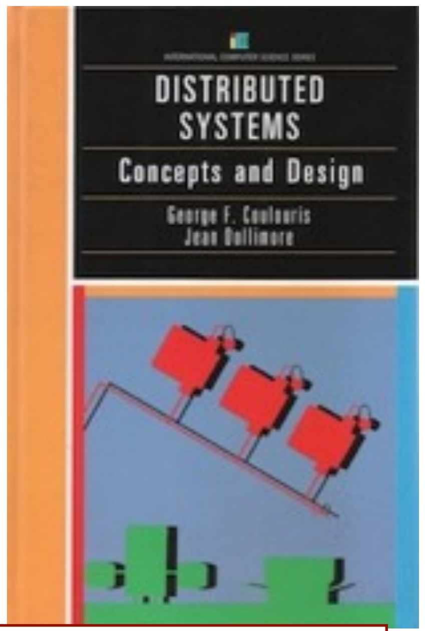
Ed1 1988



From RPC to Distributed Objects ...

- 1975 to 1986: RPC-based applications
  - Only shared hardware and shared files (FTP, Telnet, email, print, Sun NFS)
  - No objects or structured data
  - Advanced distributed apps used *ad hoc* RPC between peers
  - Interactive applications got useful structure with Smalltalk (1976 - single computer)

Ed1 1988



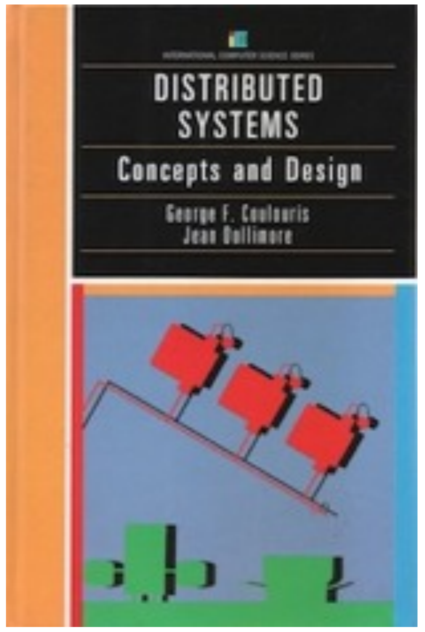
J Dollimore, E Miranda, W Xu (1991). The Design of a System for Distributing Shared Objects. The Computer Journal, 1991

-
  - Research-based distributed object systems (Emerald, Argus, Arjuna)
  - CORBA (v1: 1991, v2: 1996, v3 1999)
  - More system services: name servers, secure key distribution, database, ...

From RPC to Distributed Objects ...

- 1975 to 1986: RPC-based applications
  - Only shared hardware and shared files (FTP, Telnet, email, print, Sun NFS)
  - No objects or structured data
  - Advanced distributed apps used *ad hoc* RPC between peers
  - Interactive applications got useful structure with Smalltalk (1976 - single computer)

Ed1 1988



- 1987: Distributed shared objects
  - Research-based distributed object systems (Emerald, Argus, Arjuna)
  - CORBA (v1: 1991, v2: 1996, v3 1999)

'CORBA has moved from being a bleeding-edge technology for early adopters, to being a popular middleware, to being a niche technology that exists in relative obscurity.

.... CORBA's history is one that the computing industry has seen many times, and it seems likely that current middleware efforts, specifically Web services, will re-enact a similar history'.

The Rise and Fall of CORBA, Michi Henning, ACM Queue, June 2006

... to Web Applications ...

- 1992: WWW emerges
  - just one application model - hypertext
  - Introduces HTML, HTTP, URI's
- 1996: JavaScript introduced in Netscape Navigator
  - and JScript dialect 'ported' to Internet Explorer in same year
- 1996-2004: DOM and Dynamic HTML
  - 1998: DOM Level 1: Full Dynamic HTML (and CSS) *W3C standard*
- 1999: AJAX
  - Then known as XMLHTTP (initially MS ActiveX) *W3C Standard 2006*
- 2004-: Major AJAX deployments
  - Google Mail (2004) and Google Maps (2005), ... now thousands

Arguably, MS let an AJAX genie out of thr bottle and it became a instrument for their own frustration.

... to Mobile Platforms

- 2008: iPhone SDK
  - MVC based
  - Traditional data structures (without garbage collection)
  - Support for devices: GPS, camera, accelerometer, mic, ..
  - Disconnected operation ⇨ local storage of data models
  - HTTP <u>or</u> TCP/IP
- 2008: Android SDK
- To come - ubiquitous systems that:
  - adapt continually to context
  - run and evolve 24/7
  - use peer-peer and ad hoc networks

Web Application Technologies

- Client side: Views and controllers implemented in JavaScript with Dynamic HTML + DOM + CSS + SVG + ...
- Server side: data model managed by a service - implemented in Java, Python, PHP, Pearl with an SQL database
- Communication: AJAX, XML, JSON, Protocol Buffers, ...
- Architectures: SOAP, REST, ...

REST (Representational State Transfer)

- GET, POST, PUT and DELETE applied to representations of *resources*
  - How to define the scope of resources?
- A useful post-hoc statement of WWW principles
  - Restatement of goals derived from DS research:
    - keep servers stateless
    - provide the client with a sizeable chunk of application state
- Is REST consistent with object sharing?
  - Transactions?
  - Semantics?

Conclusions

- The web and XML were inevitable reactions to the ‘data explosion’.

Tim Berners-Lee gets a lot of the credit for being almost alone in foreseeing that an object-oriented model would be too complex in a world where everyone can contribute data - and every company/organisation can define new types.

Conclusions

- The web and XML were inevitable reactions to the ‘data explosion’.
  - but they are entirely data-oriented
  - So each interactive application must create semantics for the data
  - direct manipulation -> lots of JavaScript programming
  - can we escape from JS and develop a more productive environment?
- Is the object-oriented vision still viable?
  - In which data carries its own (operational) semantics, with interface specifications for public operations
  - achievable via components?
- Future challenges: mobility and ubiquity
  - Need to address: physical context, resource discovery, intermittent connection, ...

Early 2011

DISTRIBUTED SYSTEMS
Fifth Edition
Coulouris
Dollimore
Kindberg
Blair

Including new chapters on:

- **Distributed Objects and Components**
- **Case Study: Google Infrastructure**

Questions?
Comments?

My WebApp Development Course (2009), http://www.cl.cam.ac.uk/~gfc22/webAppCourse/

More references on the next slide.

References

Distributed Shared Objects and GUIs:

J Dollimore, E Miranda, W Xu (1991). The Design of a System for Distributing Shared Objects. *The Computer Journal*, vol. 34 , No. 6, pp. 514 - 521 (Dec. 1991)

REST:

Fielding, R. (2000). *Architectural Styles and the Design of Network-based Software Architectures*, PhD. Dissertation. http://www.ics.uci.edu/~fielding/pubs/dissertation/top.htm

CORBA:

Seetharamanan, K. (ed.) (1998). Special Issue: The CORBA Connection, *Comms. ACM*, October, Vol. 41, No. 10.

Web Services:

Birman, K.P. (2004). Like it or not, Web Services are Distributed Objects! *Comm. of the ACM*. Vol. 47, No. 12, pp. 60-62. December.

Vinoski, S. (2002). Putting the `Web' into Web Services. *IEEE Internet Computing*. July-August 2002.

Vogels, W. (2003). Web Services are not Distributed Objects. *IEEE Internet Computing*. Nov-Dec 2003.

Google Infrastructure:

Jeff Dean (2009), Challenges in Building Large-Scale Information Retrieval Systems, WSDM09. (*slides*)

Our book:

Coulouris, Dollimore, Kindberg, Blair. Distributed Systems: Concepts and Design, Pearson Education/Addison Wesley, fourth edition 2005, fifth edition to be published 2011.

Introductory course on Web App development:

George Coulouris (2009), WebApp Development Course, http://www.cl.cam.ac.uk/~gfc22/webAppCourse/