Mashup Development Seminar

Tampere University of Technology, Finland

Fall 2008

http://www.cs.tut.fi/~taivalsa/kurssit/MADS2008/

Prof. Tommi Mikkonen
Dr. Antero Taivalsaari

Background

- History of computing and software development is full of disruptive periods and paradigm shifts.
- The computing industry reinvents itself every 10-15 years.
- Examples of disruptive eras:
  - Minicomputers in the 1970s
  - Personal computers in the 1980s
  - Mobile software and Web 1.0 in the late 1990s

The Next Paradigm Shift!

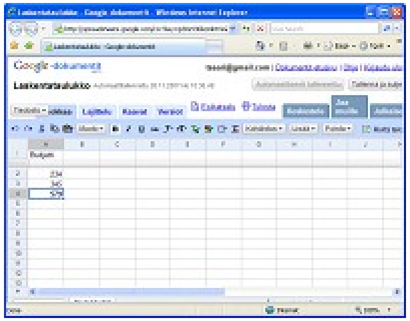
- The widespread adoption of the World Wide Web is reshaping our world in various ways.
- Documents, photos, music, videos, news and various other artifacts and services have already started migrating to the Web.
- Many industries (e.g., publishing and entertainment) are currently undergoing dramatic transformations.
- The software industry is on the brink of a similar transformation, or a paradigm shift.

Evolution of the Web

1) Simple pages with text and static images only
(e.g., http://www.google.com)

2) Animated pages with plug-ins
(e.g., http://www.cadillac.com)

3) Rich Internet Applications
(e.g., docs.google.com)



What's Next?

Web Software – Implications

- Web-based software will dramatically change the way people develop, deploy and use software.
- No more installations!
  - Applications will simply run off the Web.
- No more upgrades!
  - Always run the latest application version.
- Instant worldwide deployment!
  - No middlemen or distributors needed.
- No CPU dependencies, OS dependencies, ...
  - The Web is the Platform.

Unfortunately...

- The web browser was not designed for running real applications.
  - It was designed in the early 1990s for viewing documents, forms and other page-structured artifacts – *not* applications.
  - Programming capabilities on the web were an afterthought, not something inherent in the design of the browser.
- Various Rich Internet Application (RIA) technologies have been introduced recently to retrofit application execution capabilities into the web browser.

Best Known RIA Technologies

- At this point, the following Rich Internet Application development systems are best known:
  - Ajax
  - Ruby on Rails
  - Google Web Toolkit & Google Gears
  - JavaFX
  - Adobe AIR (Apollo)
  - Microsoft Silverlight

Landscape of RIA Technologies

Browser-based

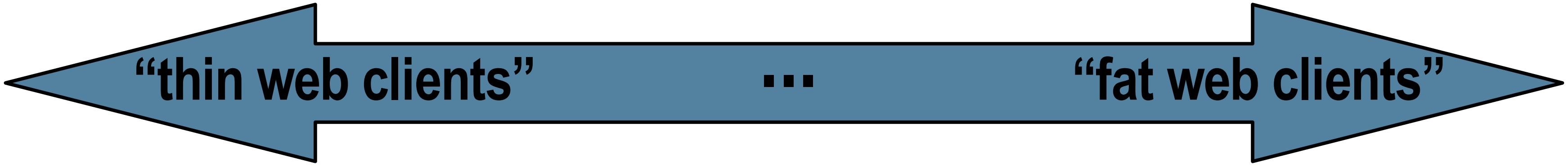
- Ajax
- Google Web Toolkit
- Sun Labs Lively Kernel

Plugin-based

- Flash & Flex
- (Java FX, AIR)
- (Microsoft Silverlight)

Custom runtime

- Java, Java FX
- Adobe AIR
- Silverlight



- Run in a standard browser
- No plug-ins needed
- Platform-independent
- Browser-based UI

- Browser plug-in required
- Custom UI

- Custom execution engine required
- Runs outside the browser
- Custom/native UI

RIA technologies were investigated in our previous seminars in 2006/2007

Mashups
and
Mashup Development

Web 2.0 – What Is It Really About?

- *Interaction*. Bringing back some of the best qualities that desktop software had before the Web, such as direct manipulation, instant feedback, piecemeal display updates.
- *Collaboration*. Allowing the users across the planet to work with each other, and share the same data, applications and services over the Web, regardless of their physical location.
- *Mashups*. Being able to combine content available on the Web in novel ways.

An Important Observation

- Web applications are ***not*** just conventional desktop applications running in the web browser.
  - Not just word processors, spreadsheets, e-mail clients, ...

- The Web enables the creation of entirely new types of applications and services that combine content from other web sites dynamically.
  - This would not have been possible with conventional shrink-wrapped applications distributed in binary form.

Mashups

- *Mashup*: A web site that combines content from more than one source (multiple web sites) into an integrated experience.
- Mashups leverage the power of the Web to support worldwide sharing of content that would not have been easily accessible or reusable before the Web.
- In principle, the content to be combined can be anything (text, source code, maps, video, blogs, product reviews, price data, ...) as long as it can be meaningfully combined with other content.

Examples

- Chicago Police Department crime statistics mashup (http://chicago.everyblock.com/crime/)
- Parking availability mashups (e.g., http://www.parkingcarma.com/)
- Traffic tracking and congestion mashups (e.g., http://dartmaps.mackers.com/)
- Real estate sales and rental mashups (e.g., http://www.housingmaps.com/)

Observations on Mashups

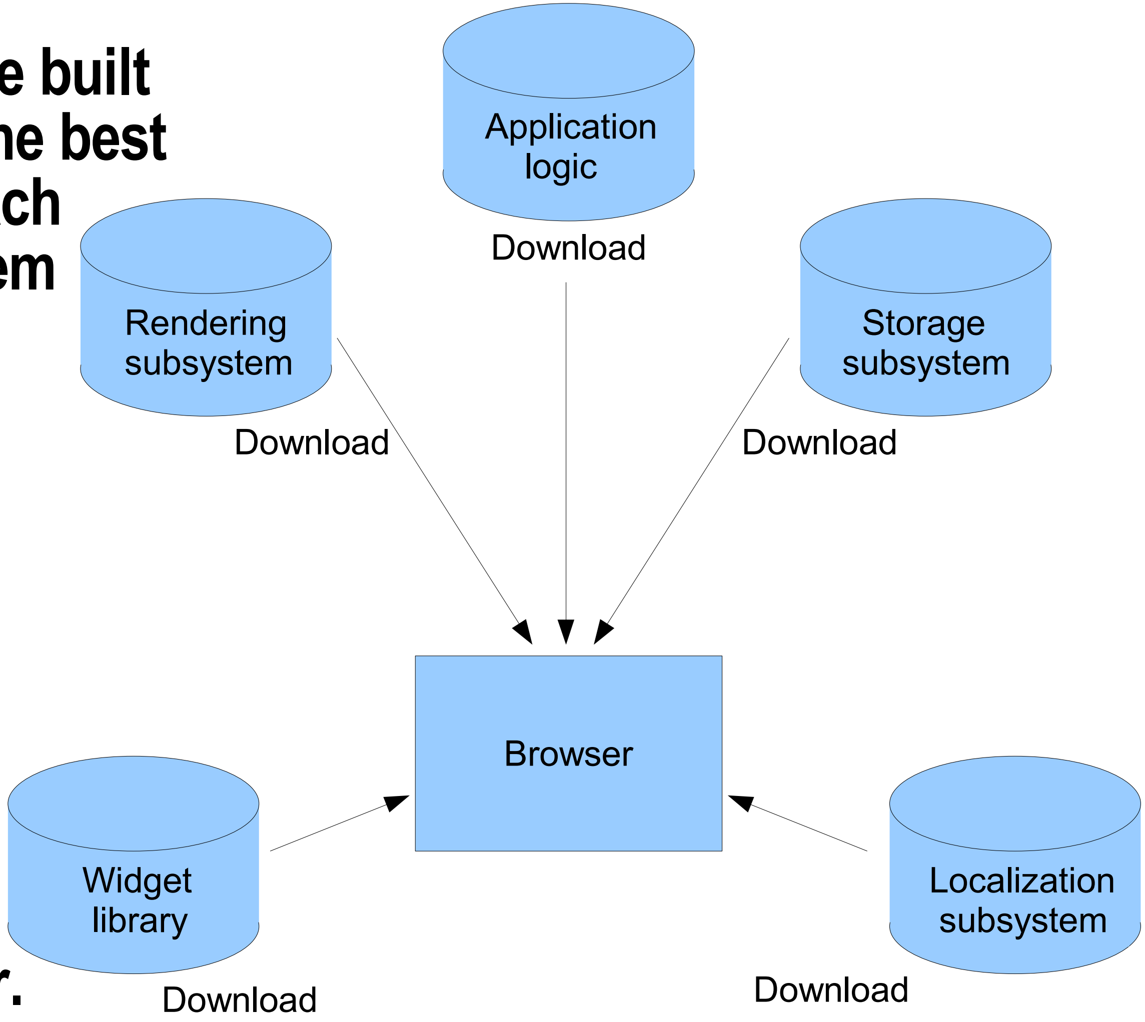
- Today, most mashups are built around *maps*.
- However, in principle the content can be anything as long as it can be digitalized and shared over the Web.
- Mashups are usually generated dynamically with no static linking; textual representations such as HTML, XML or JSON favored.
- In principle, it would be possible to build software as a mashup as well.

“Software as a Mashup”

In the future, software can be built by dynamically combining the best available components for each purpose by downloading them dynamically from different web sites.

No static linking; everything downloaded on demand.

Software development will be an inherently social activity between developers who do not necessarily know each other.



Today's web browsers do not support these kinds of applications yet!

Comments on Mashup Development

- Mashup development is still an *ad hoc* activity.
  - No established development principles.
  - Similar comments apply to web development more generally.
- The lack of proper interface descriptions and modularity makes it difficult to combine content in a systematic fashion.
- The current security model of the web browser is poorly suited to mashup development.
  - The *Same Origin Policy* restricts access to other web sites, see http://en.wikipedia.org/wiki/Same_origin_policy

Web Development vs. Conventional Software

The Impedance Mismatch

Web Development	Conventional SW Development
- Documents	- Applications
- Page / form oriented interaction	- Direct manipulation
- Managed graphics, static layout	- Directly drawn, dynamic graphics
- Instant worldwide deployment	- Conventional deployment
- Source code and text favored	- Binary representations favored
- Development based mostly on conventions and “folklore”	- Development based on established engineering principles
- Informal development practices	- More formal development
- Target environment not designed for applications	- Target environment specifically intended for applications
- Tool-driven development approach	- A wide variety of development approaches available

Tools for Mashup Development

- Manual development of mashups can be very tedious.
- Manual mashup development requires:
  - (1) manual "scraping" of content from different web sites,
  - (2) coming up with an algorithm (usually JavaScript code) for combining the content,
  - (3) visualizing the resulting information using existing web technologies.
- A lot of tools for mashup development are currently under development to simplify the process.

Commercial Mashup Tools

- Dapper, http://www.dapper.net/
- Google Mashup Editor, http://code.google.com/gme/
- IBM Mashup Center, http://www-01.ibm.com/software/info/mashup-center/
- IBM Project Zero, http://www.projectzero.org/
- Intel Mash Maker, http://mashmaker.intel.com/
- LiquidApps, http://www.liquidappsworld.com/
- Microsoft Popfly, http://www.popfly.com/
- Mozilla Ubiquity, https://wiki.mozilla.org/Labs/Ubiquity
- Open Mashups Studio, http://www.open-mashups.org/
- Yahoo Pipes, http://pipes.yahoo.com/

Academic & Less Widely Known Tools

- d.mix (Stanford University), http://hci.stanford.edu/mashups/
- Marmite (Carnagie Mellon University), http://www.cs.cmu.edu/~jasonh/projects/marmite/

=========

- Anthracite, http://metafy.com/products/anthracite/
- C3W (Clipping, Connecting and Cloning for the Web), http://www.iw3c2.org/WWW2004/docs/2p444.pdf
- Internet Scrapbook (one of the first mashup systems), http://www.sigchi.org/chi97/proceedings/short-talk/as.htm
- And many others...

About the Seminar: Practical Arrangements

Why This Seminar?

- Mashup development will be increasingly important in the future.
  - Web applications are *not* just desktop applications running in your web browser.
  - Mashup development enables the development of entirely new kinds of web applications and services by allowing web sites to combine content produced by thousands/millions of people across the planet.
- Examine mashup development and the emerging technologies and tools in this area.
  - Build real mashups using those technologies.
  - Drill deeper into those technologies that seem most likely to succeed.

Practical Arrangements

- The seminar will be held on Thursdays in TC103, 14:15 - 15:45.
- Seminar dates:
  - 11.9., 18.9., **23.9.**, (2.10.), 9.10., (16.10.), 23.10., 30.10., (6.11.), 13.11., 20.11., 27.11., 4.12., 11.12., (18.12.)
- NOTE: No seminar on September 25 (two weeks from now). Seminar will be held on Tuesday, September 23 instead, in TC210.

How to Get Credits?

- Maximum number of credits: 3-5 op

- Attendance: 1 op
- Seminar presentation on selected mashup development technology: 2 op
- Successfully written, new demo application using the technology: additional 2 op

Choosing Presentation/Demo Topics

- Please choose your presentation topic, demo application and the preferred presentation date as soon as possible.
- E-mail to: *tjm[at]cs.tut.fi* and *taivalsa[at]cs.tut.fi.*
- Topics allocated on a “first-come-first-serve” basis.
- Seminar web site page will be updated [ir]regularly to list the chosen presentation topics:
  - http://www.cs.tut.fi/~taivalsa/kurssit/MADS2008/
- Presentations can be held in Finnish or English.
  - English preferred if there are non-Finnish-speaking participants.

Action Items for Next Week (Sep 18)

1. Choose your preferred presentation topic.

2. Come up with a great idea for a new kind of a mashup.

Thank You! Questions?

Tommi Mikkonen
Antero Taivalsaari

http://www.cs.tut.fi/~taivalsa/kurssit/MADS2008/