beta version

Rich Internet Applications

Building Blocks & Trends

Sebastian Salvucci &
Mariano Cilia

Web 2.0…

"Web 2.0 is a set of economic, social, and technology trends that collectively form the basis for the next generation of the Internet—a more mature, distinctive medium characterized by user participation, openness, and network effects"

Source: Web 2.0 Principles and Best Practices
(John Musser with Tim O'Reilly and the O'Reilly Radar Team)

3

Agenda

- Web Evolution
- Rich user experiences
  - User Interaction Evolution
    - RIA / AJAX
  - OpenAPIs, Mashups
  - Lightweight programming and business models
    - Software as a service (SaaS)
    - The end of software release cycle
      - Perpetual beta
- The User Participation
  - The read/write web
  - Social software, network effects
  - Collective Intelligence
- The web as an open/global platform
  - Feeds
  - Architecture of participation
  - The Long Tail
- Enterprise 2.0
  - Web 2.0 in the enterprise
- Wrap-up

4

The Web as Platform: Netscape vs. Google

- Netscape
  - Flagship product was the web browser
  - Strategy: dominate the client-side to establish (and sell) server-side products
  - Webtop to replace Desktop with information updates, applets, …
  - Information providers would purchase Netscape servers
- Both web browsers & servers are today commodities
  - The value moved up to services delivered over the web platform

[O'Reilly'05]

5

The Web as Platform: Netscape vs. Google (cont.)

- Google: a specialized Database!
  - Native web app
  - deliver services
  - customers paying (in)directly for their use
  - “no” software releases, just continuous improvement
  - No porting to different platforms
  - Massively scalable collection of PCs running Op-So OS + own apps (database management) and utilities (crawlers)
    - Without the data, the tools are useless, without the software, the data is unmanageable

6

The Web as Platform: DoubleClick vs AdSense

- Web Publishing
  - DoubleClick over 2000 big sites as customers
    - Internet dominated by top web sites
    - Formal sales contract
    - Publisher/ad-agency friendly advertising formats (banners, pop-ups)
  - AdSense (& Overture) serve hundreds of thousands
    - Ad placement on virtually any web page
    - Minimmaly intrusive, context-sensitive, consumer-friendly text advertising

7

The Web as Platform: DoubleClick vs AdSense (cont.)

- The long tail:
  - Collective power of the small sites that make up the bulk of the web's content
- Other stories
  - eBay: enables occasional TXs of only a few dollars, acting as automated intermediary
  - Napster: architected a system where every downloader also became a server

8

The Web as Platform: Akamai vs. BitTorrent

- Scalability aspects
  - Akamai: business with the big companies
    - Replication of high-demand sites
    - Revenues from those sites to benefit individuals
  - BitTorrent: Internet decentralization
    - Every client is also a server
    - The more popular the file, the faster is can be served
- New principle:
  - The service automatically gets better the more people use it

9

Collective Intelligence

- Yahoo! born as a catalog of links
- Google's PageRank exploits the link structure of the web rather than the document characteristics
- eBay is based on the collective activity of its users
- Amazon uses "most popular" suggestion
- Wikipedia is an experiment of trust since any web user can add or modify an encyclopedia entry
- Cloudmark, a collaborative spam filter, aggregates the individual decisions of email users
- Viral marketing: recommendations propagating directly from user to user
- Peer-Development: the Internet infrastructure (linux, apache, mySQL, PHP, Perl, …) relies on open-source (collective, net-enabled intelligence)

10

Rich User Experiences

- The web used to deliver "applets" and other kinds of active content within the web browser
  - java applets
  - JavaScript and DHTML as lightweight client-side programmability
  - Flash
- The potential of the web to deliver "full scale apps" appeared with GMail, followed by Google Maps
  - All these relying on AJAX
  - Web developers finally able to build web apps as rich as local PC-based apps

11

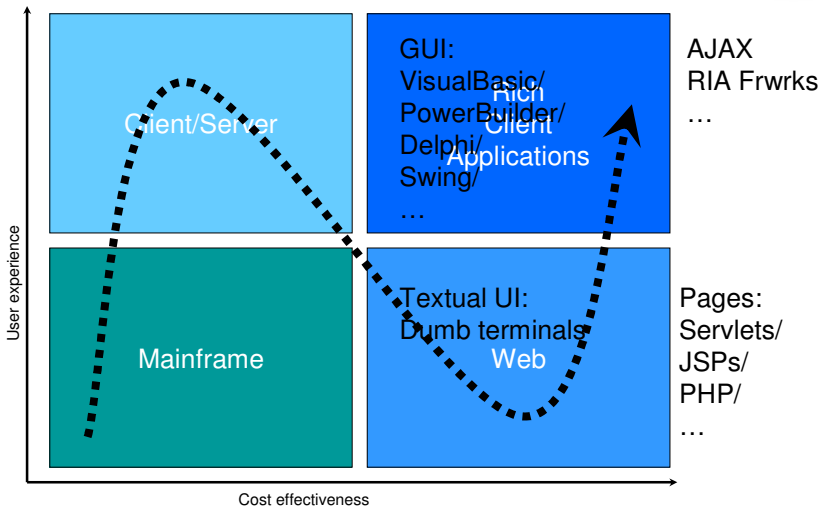
Web Evolution by Example

DoubleClick	-->	Google AdSense
Ofoto	-->	Flickr
Akamai	-->	BitTorrent
mp3.com	-->	Napster
Britannica Online	-->	Wikipedia
personal websites	-->	Blogging
domain name speculation	-->	search engine optimization
screen scraping	-->	web services
Publishing	-->	participation
content management systems	-->	wikis
directories (taxonomy)	-->	tagging ("folksonomy")

12

Rich user experiences

User Interaction Evolution



14

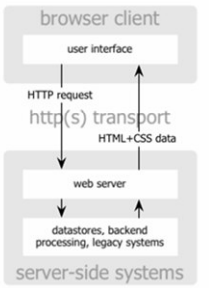
Classical Web Applications (1)

- Delivery means server-based, centrally manage deployment
  - Cost-effective
- Frees users to configure, install application updates
- Applications are virtually available anywhere

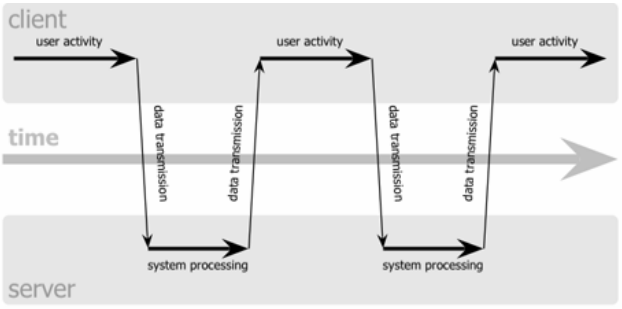
15

Classical Web Applications (2)

- User completes and submits form
- to a web server (user waits)
  - Sends a new web page as a response



classic
web application model



Classical Web Apps (cons)

- Poor user experience in contrast to traditional locally running client code
  - Inferior user experience
  - A step back in the evolution of user interfaces
- User is blocked waiting for server response (new web page)
- Clients need to refresh and re-render the complete HTML page even if a small change is needed
- The advantages of web app outweigh the flaws
  - So end-users pay the price!
- Web app development is based on lowest common denominator

17

Rich Internet Applications

From Web to RIA, 3 dimensions:

- Go beyond traditional web-page metaphors
  - Combine the best of desktop (GUI) and online software (Web)
  - Deliver rich user experiences
- A new application distribution technique
  - Browse to an application, rather than install an application
  - Significant implications on support and updates
- A new development methodology
  - markup meets OO

The potential of the web to deliver "full scale apps" appeared with GMail, followed by Google Maps

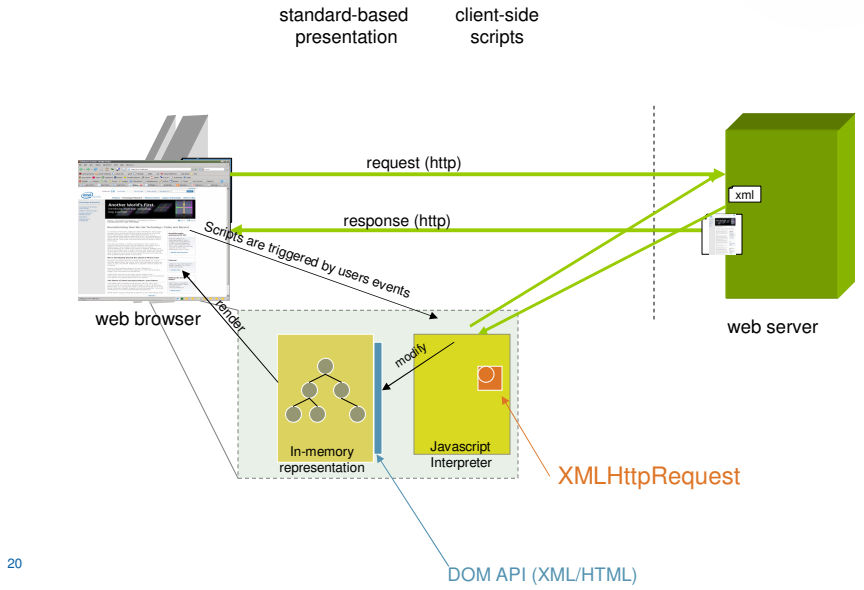
18

AJAX
(Asyncronous JAvascript & Xml)

- Not a technology, a web technique
  - standards-based presentation using XHTML and CSS
  - dynamic display and interaction using the Document Object Model
  - data interchange and manipulation using XML and XSLT
  - asynchronous data retrieval using *XMLHttpRequest*
  - JavaScript binding everything together
- Focus on a client fetching an HTML document that acts as a container into which to inject content
  - Using XML data (small pieces of data) retrieved from server-side without reloading the page
    - Quickly response to many interactions
    - Avoids sending unchanged info across the network
- No browser plug-in is needed
  - Java Applets, ActiveX, Flash are proprietary
- Standards-based key enabler for RIAs
- Page-specific control logic embedded as JavaScript code

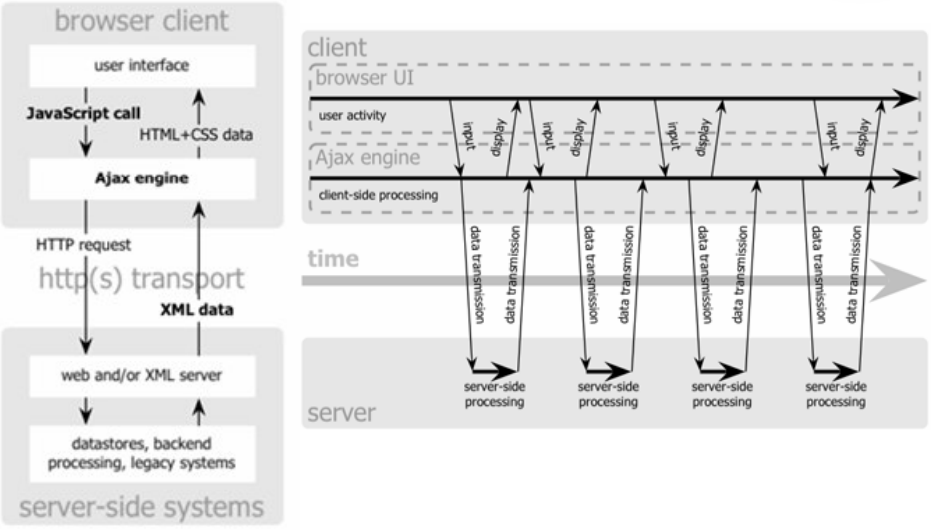
19

AJAX in action



20

AJAX Web App Model



XMLHttpRequest is the key!

- Not new!
  - introduced by M$ in IE years ago
- Not part of the JavaScript specification
- Nonetheless implemented in IE, Firefox, Safari,…
  - Small differences by creating XMLHttpRequest object

22

AJAX - Typical uses

- Online form data validation
  - e.g. userIDs, Serial Numbers, Codes, … that requires server-side validation
- Autocompletion
  - e.g. email addresses (from contact database), …
- Master-detail operations
  - e.g. an invoice form by fetching the description of codes of items
- Refreshing data on the page
  - e.g. Latest news (even relying on polling) does not need to reload the whole page

23

AJAX Cons

- Complexity now on the generation of web pages that embed complex JavaScript code
  - Design and maintainability, debugging
  - Good Ajax programmers are hard to find
- User might experience (unexpected) delays
- Server-side scalability
- Viewable source code
  - Open your apps to hackers
- Cannot be a complete rich user experience
  - Graphics, video, audio

RIA/Ajax Frameworks

- **Different flavors of AJAX**
  - Client JavaScript libraries
    - Prototype, ajax.net, Google XPath/XSLT
  - User Interface Components (AJAX enabled)
    - YUI, Dojo, OpenRico, script.aculo.us
  - RIA Frameworks
    - OpenLaszlo, Flex, Echo, GWT
  - RIA Frameworks + Visual Development Tools
    - Tibco General Interface
  - Push-based AJAX (messaging)
    - Based on keep-alive characteristic of HTTP
      - One request -> an infinite response
      - ActiveMQ AjaxClient, TIBCO AJAX Message Service

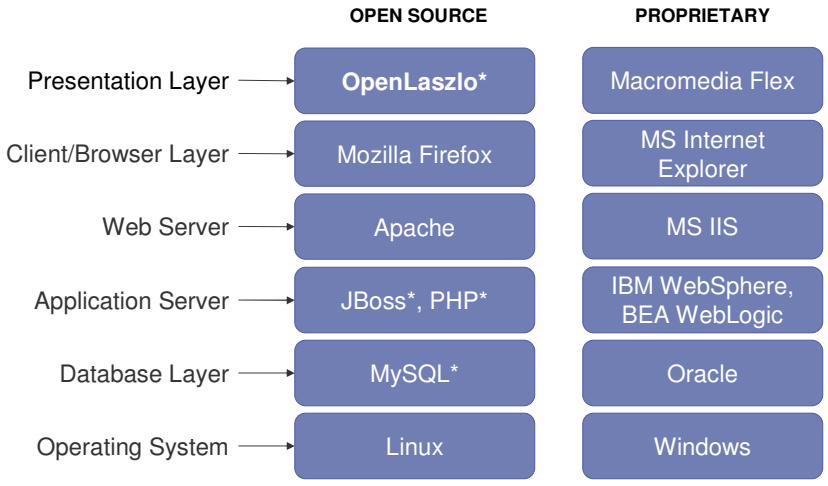
OpenLaszlo

- Free open source platform for RIA
- Applications written in LZX
  - XML + javaScript
- Object-oriented development and data binding
  - Compiled/transformed into SWF (flash) and DHTML
- Portable and stand-alone
  - Flash ubiquitous plug-in on browsers
  - Also runs outside a browser
- Supports rich media
- Firewall friendly
  - SOAP, XML-RPC, http,…

[OpenLaszlo.org]

26

Open source software stack



27

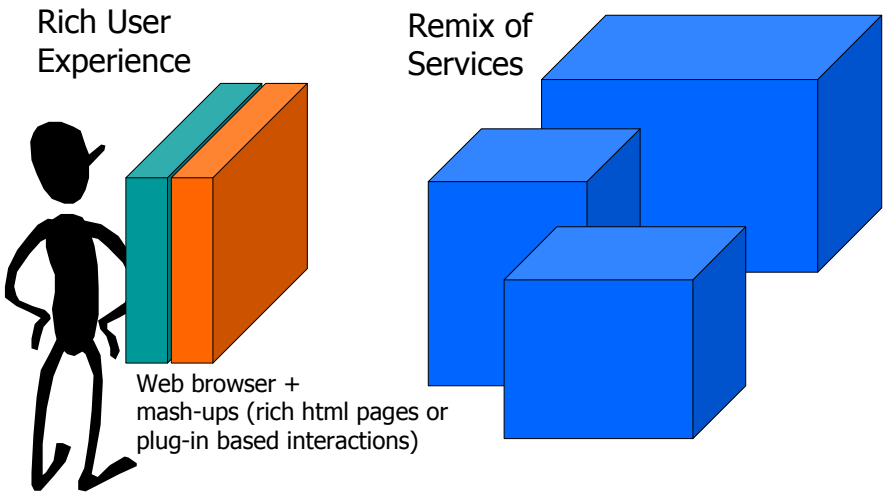
UI: The End of Page Refresh?

- Delivered into a Web browser without installation, and offers:

- Direct manipulation
- Declarative (vs. Procedural) Development
- Data Binding
- (Fully) Object-Oriented
- Scripting
- Event-Driven
- Pre-built Components
- Media
- Powerful Constraints System
- Keyboard/Mouse Control
- Drag-and-drop
- Animation
- Drawing API
- Layout
- Browser Integration

28

New Kind of Applications



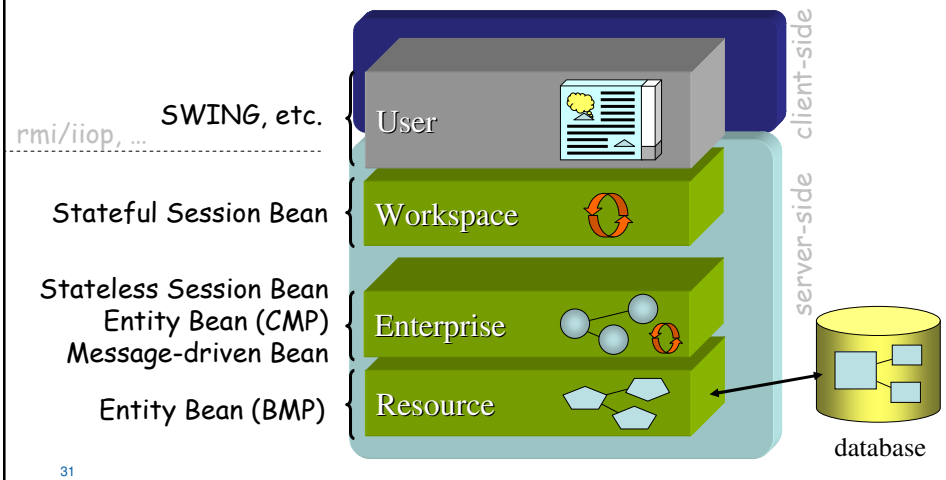
29

Fat Web Clients

- AJAX:
  - Browser: JavaScript enabled (no plug-in required)
  - Web page contains client-side code and optionally refers to some (javascript) libraries
  - Look&Feel can be changed (CSS)
  - Can fetch data from server without reloading the page
- OpenLaszlo:
  - Browser: Flash plug-in
  - Web page loads flash code that embeds the complete client-side code
  - Look&Feel embedded in code
  - Can fetch data from server without reloading the page

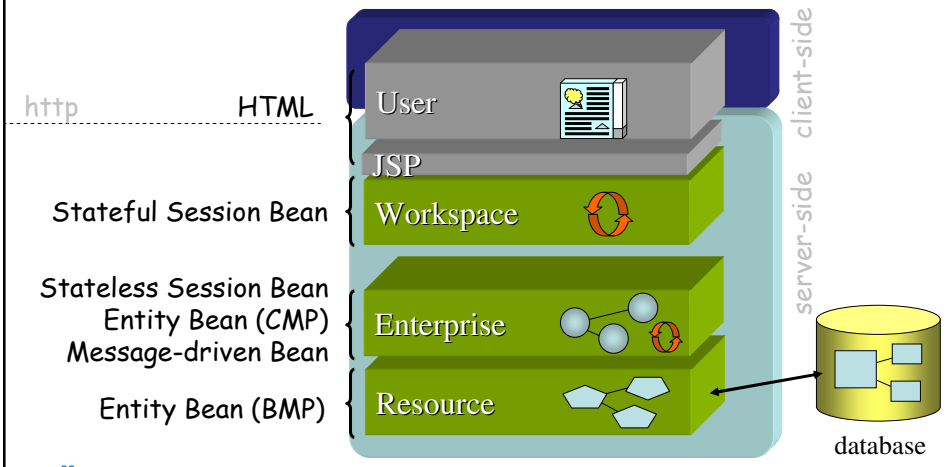
30

Multi-tier Architecture: Review



31

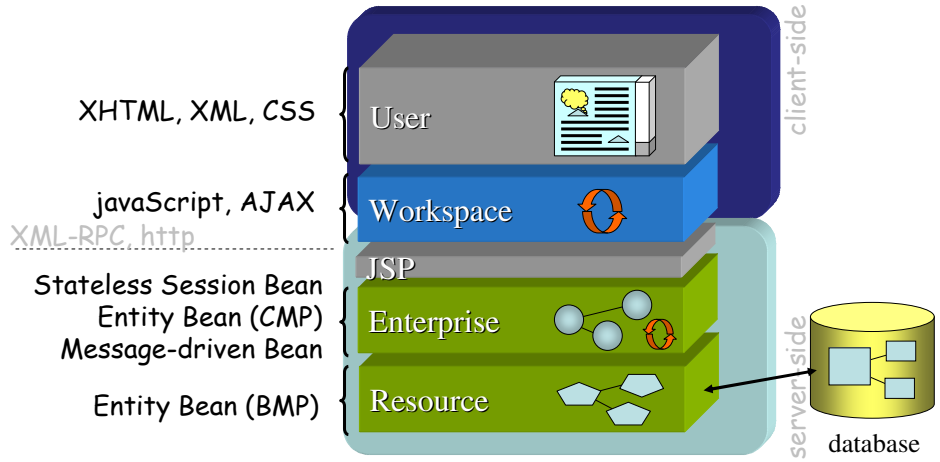
Multi-tier Architecture: Review



32

Multi-tier Architecture: Review

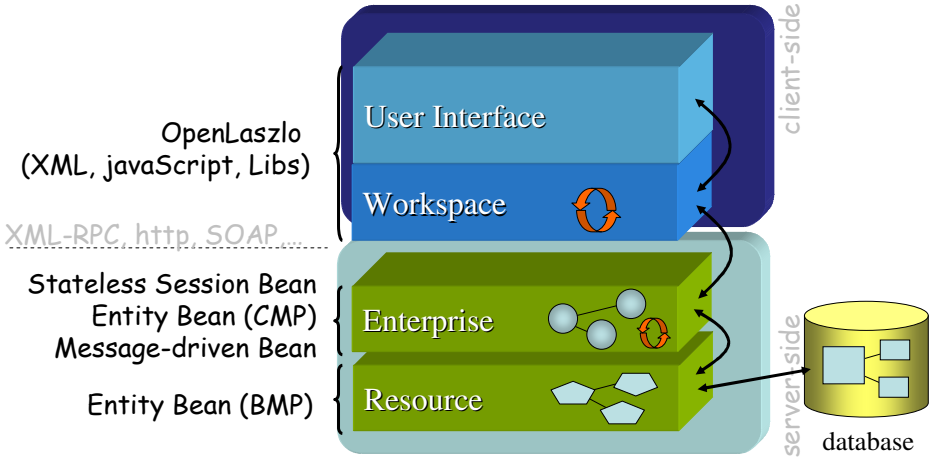
Separation of Concerns



33

Multi-tier Architecture: Review

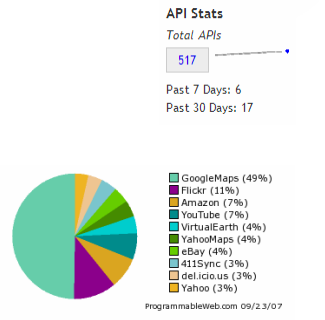
Separation of Concerns

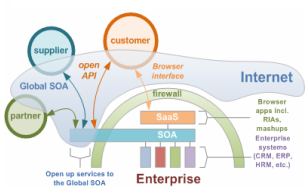


34

OpenAPIs

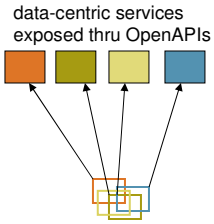
- Services public/globally exposed through an API
  - Allows the user/machine access programmatically to the data instead of browsing it
  - Separation of UI from the access to logic and data
- Initially based on WS
  - Lightweight protocols like REST become de-facto standards
  - Language-specific client libraries are becoming popular



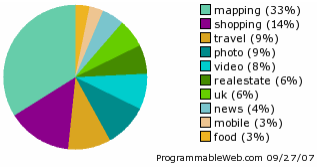


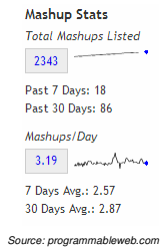
35

Mashups



Offering a new service by compositions of existing ones
(usually for not envisioned uses)





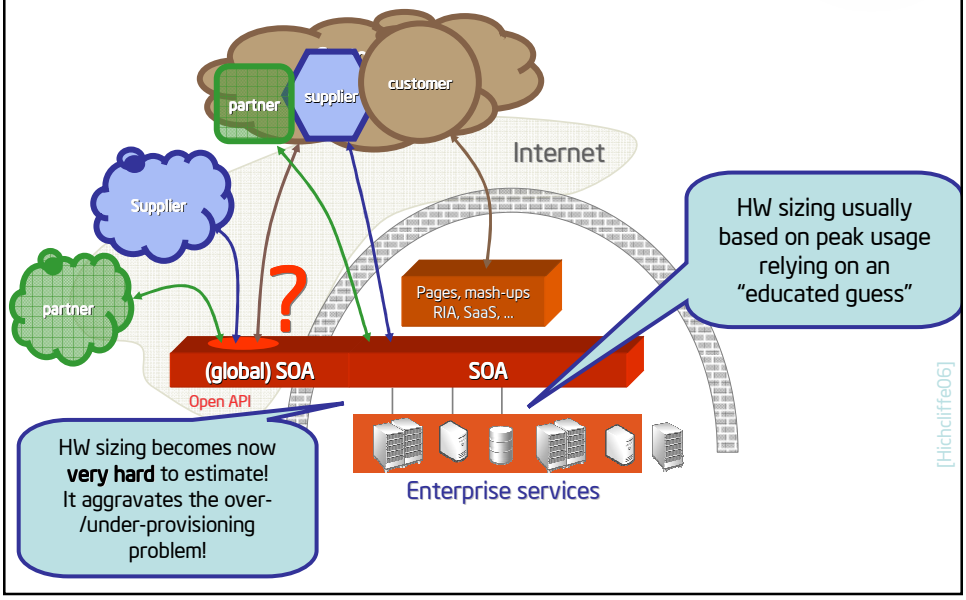
Compositions can take place on the server but also in the client

The first mashup… housingmaps.com

Mashup platforms

- Highly accessible and easy-to-use mashups creation tools
  - Could users solve their situational software integration problems on the spot themselves?
  - Will mashups unleash end-users to create the software they want, just like they are now creating the content they want?

Mash-ups and SaaS are Emerged



End of Software Release Cycle

Internet era: software delivered as service (not as a product) leading to:

- **Operations** must become a **core competency** → continuous maintenance
  - Expertise in product development + expertise in daily operations
  - Google's sys admin, networking, load balancing techniques are top secrets!
  - Scripting languages (Perl, Python, PHP, Ruby…) playing a key role
- Users must be treated as co-developers Open-source from "release early and release often" to "**the perpetual beta**"
  - New features on a monthly, weekly basis!
  - Real time monitoring of user behavior to analyze which features are used, how they are used.
    - Improve and/or remove features

39

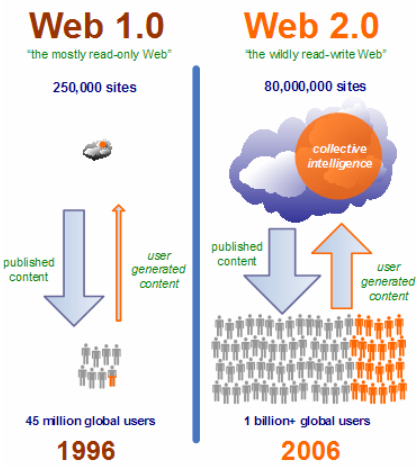
Software as a Service

- Software application delivery model
  - Software vendor develops a web-native software application and hosts and operates it for use by its customers over the Internet
  - Centralized feature updating, which obviates the need for downloadable patches and upgrades
  - Generally priced on a per-user basis
  - Closer to a one-to-many model (single instance, multi-tenant architecture) than to a one-to-one model
- Replacing the ASP generation
  - Rich Clients + APIs for their operation (relying on WS/REST) instead of client/server apps w/ HTML front-ends
  - Enables third-party apps based on public APIs
    - AppExchange (Salesforce.com)
- Cost-effective scalability is needed
  - Unpredictable demand, management issues
  - Emergent services based on pay-per-use for quickly adoption
    - Amazon EC2, S3

40

The user participation

The read/write web

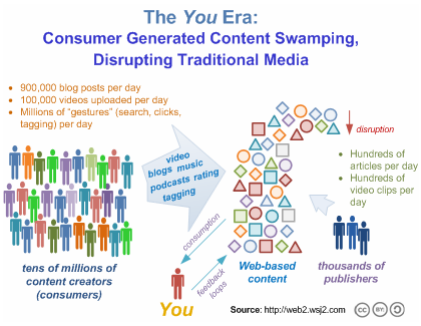


- The user becomes a contributor that adds value
  - Wikipedia, blogs, photo/video podcasts, folksonomies
  - pageRank, eBay reputation, Amazon reviews, comments, Digg this, etc
- Trusting users as co-developers
- Systems get better the more people use them

42

Social software

- Computer mediated-collaboration
  - Allows users to interact, share, collaborate and meet other users (conversation instead of monologue)
  - Community creation (social structures based on trust, reputation, meritocracy)
  - Cooperate, don't control
- Viral feedback loops, network effects



- Easy/ready to use tools
  - Blog systems, wikis, messaging, bookmarking, social networking, photo/video sharing, podcasting, etc..
- Makes it possible for anyone to reach the over 1 billion users that presently comprise the Web
- Shift control from institutions (centralized) to users (decentralized)

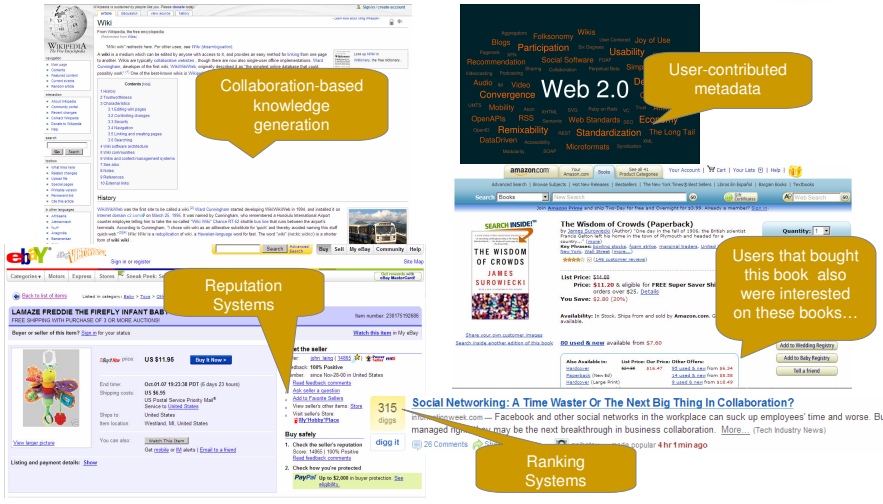
43

“Social features”

44

Collective Intelligence

Not just user-generated content (the wisdom of crowds)



45

Explicit and tacit knowledge

The web as an open/global platform

Web Feeds (Syndication)

- Used to publish frequently updated content such as blog entries, news headlines or podcasts.
  - Designed to be machine-readable
    - XML-based standard formats (RSS/Atom)
  - A section of a website is made available for other sites to use in a standard way
  - Can be read using a feed reader or aggregator
    - Making a collection of web feeds accessible in one spot
  - One-to-many communication
- Extended for specific usages
  - GeoRSS, Microformats, etc.
- Other potential uses ?!
  - Common glue for integration by exposing data from silos other than news

47

Data is key!

- The race is on to own certain classes of core data
  - Location, identity, calendaring of public events, …
  - In many cases, significant cost to create the data
  - In others, try to reach critical mass via user aggregation (and turn aggregated data into a system service)
    - Reputation systems, most popular, best ranked, related items, buyers of this item also bought…
- Unique, hard-to-recreate data sources OR similar to the free software movement, proprietary databases becoming free (free data movement) ?

48

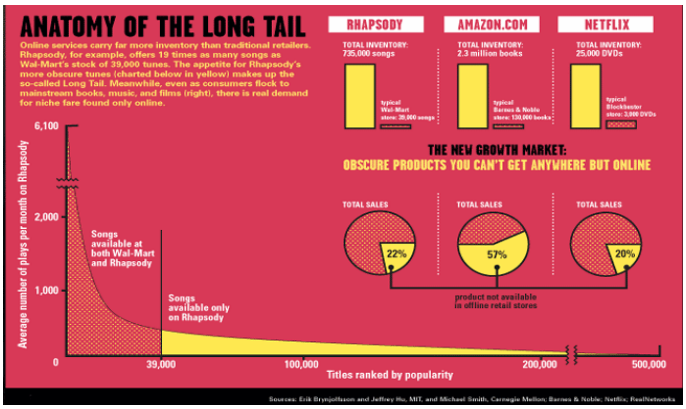
Architecture of participation

- Open architecture
  - Turning applications into Web Platforms
- Foster innovation in assembly, where remixing of data and services creates new opportunities and markets
  - The most successful web services are those that have been easiest to take in new directions (unimagined by their creators)
  - Support lightweight programming models
- Software that gets better the more people use them
  - Encourage every user to add value, directly to the application with implicit and explicit contributions
  - Social dimension that enables users to interact further adding application value
  - Think syndication, not coordination

49

The Long Tail

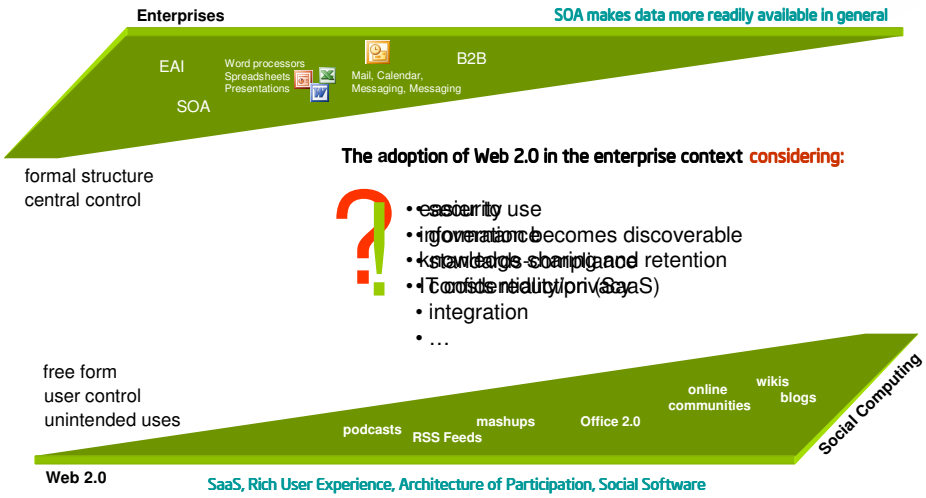
- The mass servicing of micro markets cost effectively via the Web is one of the primary "killer business models" made possible by the Internet in its present form.



50

Enterprise 2.0

Enterprise 2.0



52

Enterprise 2.0 Challenges

Some of them…

- Scalability, Manageability and Performance
  - Unpredictable throughput, capacity and difficult to monitor
- Mash-ups can result in overly complex integration
- Development changes
  - Trade-off: Fast and easy vs. well design and engineered
  - Develop and deploy constantly (perpetual beta)
  - Need proven tools and enterprise design patterns
- Lots of sensitive customer data!
- Not clear yet how to apply social networking to business models
  - Getting workers to move much of their work **from**
    - private, undiscoverable tools **to**
    - social, public, freeform tools

54

Wrapping-up

Eight Core Patterns

- Harnessing Collective Intelligence
- Data is the Next "Intel Inside"
- Innovation in Assembly
- Rich User Experiences
- Software Above the Level of a Single Device
- Perpetual Beta
- Leveraging the Long Tail:
- Lightweight Software/Business Models and Cost Effective Scalability

57

Some trends…

- New technology models
  - which software becomes a service;
  - the Internet becomes the development platform,
  - online services and data are mixed and matched;
  - syndication of content becomes glue across the network;
- New social models
  - user-generated content can be as valuable as traditional media,
  - social networks form and grow with tremendous speed,
  - truly global audiences can be reached more easily,
  - rich media from photos to videos is a part of everyday life online.
- New business models
  - facilitated by changes in infrastructure costs,
  - the reach of the Long Tail,
  - viral network-driven marketing,
  - new advertising-based revenue opportunities.

Source: Web 2.0 Principles and Best Practices: (John Musser with Tim O'Reilly and the O'Reilly Radar Team)

59

Wrap-up of 2.0

- Services, not packaged software
  - APIs
- You control your own data!
- Architecture of participation
  - Blogs, Wikis
- Trust your users (Users add value!)
  - User as contributor: pageRank, eBay reputation, Amazon reviews, Wikipedia, …
- The Long Tail
- Software that gets better the more people use it
  - Napster, BitTorrent
- Some rights reserved
  - Design for hackability and remixability
- The perpetual beta
- Cooperate, don't control
- Rich user experience
  - GMail, Google Maps, Yahoo! Maps, Zimbra

60

Wrap-up of Web 2.0 (cont)

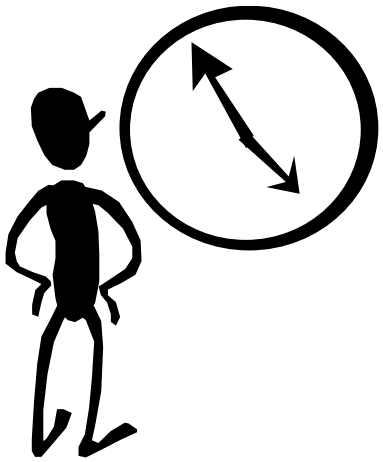
- Up in the stack (e.g. e-Bay)
  - INet as platform
  - Information businesses using software as a service
  - Harnessing collective intelligence
- Down in the stack (e.g. Google)
  - Data as the intel inside!

61

Wrap-up of Web 2.0 (cont)

- A lesson learned
  - The power may not actually be in the data itself but rather in the control to the access to that data (aggregators)
    - Adding intelligence (metadata) to the data making it easier to find things!
    - You can charge for this service (through advertising, subscriptions, …)

62



63