Web 2.0

Tutoriál

Tomáš Pitner[1,2] Pavel Drášil[1] Martin Hinca[3]

[1] Masarykova univerzita, Fakulta informatiky
tomp@fi.muni.cz, xdrasil@fi.muni.cz

[2] Universität Wien, Research Lab Educational Technologies
tomas.pitner@univie.ac.at

[3] Avitech s.r.o., Bratislava
martin@hinca.net

11. října 2009 — Srní, Czech Republic

Goal of the Tutorial

1. Organize the scattered information on Web 2.0 and present it in a comprehensible way for a wide audience
    - software analysts and developers
    - IT managers
    - business people
    - students and researchers

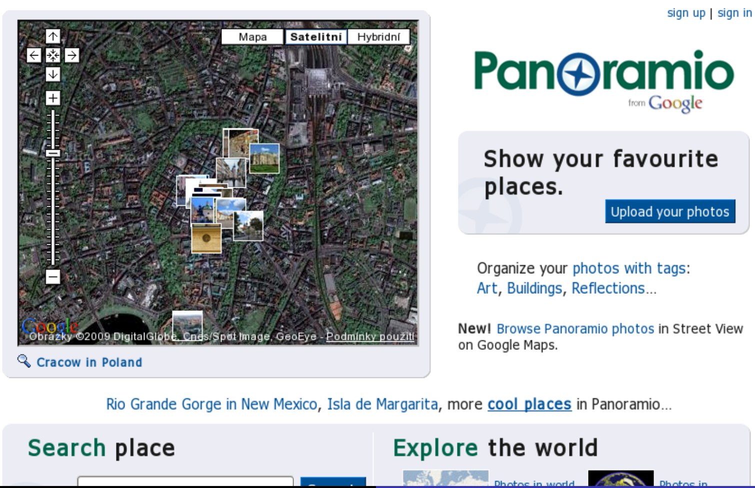
Structure of the Tutorial

1. Core concepts and principles revisited
2. Application categories
3. Web 2.0 context
   - people
   - technology
   - law.
4. Web 2.0 as development platform
5. Business in Web 2.0
6. Web 2.0 in business
7. Conclusion and consequences

History of ”Web 2.0”

- Darcy DiNucci, 1999 in *Fragmented Future* (. . . but remain unnoticed)
- **Web 2.0 Conference** in San Francisco, 2004
- Brainstorming led to its characterization
- Tim O'Reilly's paper *What is Web 2.0 . . .* in 2005 (. . . still the #1 in citations)
- Still no recognized **short definition** (we are not likely to see any. . .)

Panoramio



sign up | sign in

Cracow in Poland

Show your favourite places.

Organize your photos with tags: Art, Buildings, Reflections...

New! Browse Panoramio photos in Street View on Google Maps.

Rio Grande Gorge in New Mexico, Isla de Margarita, more cool places in Panoramio...

Why ”Web 2.0”

- Many current trends in **technology** but
- mainly in the **usage** of the Web

Technological Shift

- **incremental development** rather than a break-through
- sometimes criticized as "Nothing new!"

Exploiting the Existing Web Environment

- server/client communication
- IP-based network architecture
- web browsers as the user agents
- **interactive applications** (Rich Internet Applications – RIA)
- using old-hat client-side scripting technologies (JavaScript)
- or **specialized runtime environments** (Flash).

Mature & Consolidated Data Formats

Extensible Markup Language (XML)

- **mature** (industry standard since 1998)
- **supported** by all vendors and platforms
- **full stack** of technology (from parsers to native-XML DBMS)
- **Enterprise strength** (JavaEE, .NET) vs. dynamic, agile languages environments (Ruby)

Range of Develop&Deploy Platforms

All sizes: from enterprise strong to simple and cheap

- Enterprise strength (JavaEE, .NET) vs.
- **Dynamic, agile languages and environments** (Ruby)
- Simple **all-in-one stacks** (LAMP)
- Cheap **hosting** (PHP apps.)

Service-Oriented Architectures

Progress in networking and software architectures

- Service-Oriented Architectures in enterprise environment
- The same but simpler for the Web:
- legacy **protocols** (HTTP)
- **formats** (HTML, RSS/Atom, FOAF)

Easy to integrate: **reuse** data from a service, **remix** with others, **republish**. . .

New Computing Paradigms

- **Cloud computing**

Growth & Change

- **More users**
- Different activities:
  - **communication**
  - **collaboration**
  - **participation**

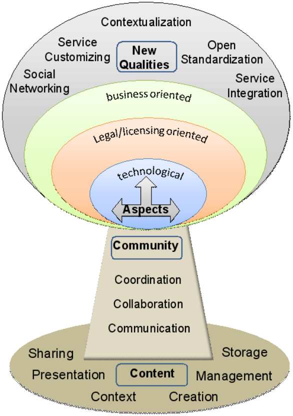
Social Networking

- Application/**platforms** like Facebook, MySpace: hundreds of millions users
- No need of special skills (often **neither computer skills**)
- Features boosting **participation**: *tagging*, *syndicating*
- **Low entry barrier**: both *end-users* & app. *providers*
- New **business models**

Rough Classification

1. **Creation**: wikis, blogs, online databases, text processing, calculations, presentations
2. **Sharing**: pictures, videos, music
3. **Storage**: P2P storage, filesystem-like services, middleware services
4. **Coordination**: communication, event and project management, IM systems
5. **Community**: social networking, orientation to professional or free-time activities
6. **Contextualization**: mashups

“Mashing-Tree”



Penetration of PCs

- Total number of PCs installed: > 1 billion
- Expected to double in 2014, mainly in emerging markets
- Mature market share decreases:
  - 58 % from the first billion
  - 30 % from the second

Mobile Devices, Wearables

- Various categories: PDAs, netbooks, smartphones, mobile phones
- Smartphones: Symbian (47 %) and BlackBerry (20 %) dominate the market
- Mobile phones: the largest market ever, **4 billions subscribers!**

Size of the Web

1. Number of **websites** or web **pages**
2. Number of **hosts**
3. Number of **users**

Sites and Pages

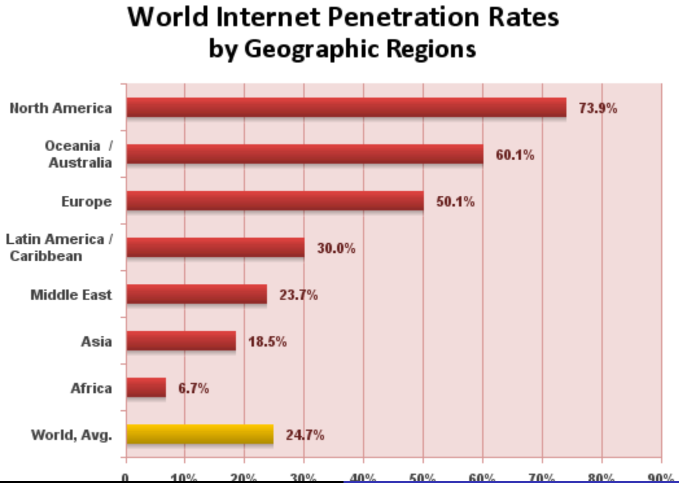
- Google indexes $> 1T$ **pages** (10^{12})
- About 625 M connected **hosts**
- 225 M **web servers**

Users

1.670 B users worldwide: *1/4 of the population!*:

Asia	700+ M
Europe	400+ M
N.Amer.	250+ M

Penetration Worldwide



Dynamics

1. Africa, Middle East: 1300+ %
2. Latin America: 800+ %
3. Asia: 500+ %

By Language

1. Arabic: 1500+ %
2. Portuguese, Russian: 800+ %

Social Networks Users

- Uneven distribution, some services popular in some regions
- Some popular worldwide (Facebook)
- Some locally: MySpace (Latin America), StudiVZ (German-speaking world)

Who’s the #1?

1. Facebook 120+ M users #1 since 2008
2. MySpace

Blogging

- Authors: upto 200 M blogs
- Readers: 384 M
- 133 M “acknowledged blogs” by *Technorati*

Global Market

- Users from multiple regions, under various legislation
- Providers with the infrastructure spread among regions and legislations

Government Infringements

- **Censorship** (extensive in China, 30 k police people constantly watching discussions, tweets, fora)
- Capability to block certain servers “on demand”: *Great Firewall of China*
- Certain control in **democratic countries**
- Some attempts denied by supreme courts
- Some regulations anchored in ToS

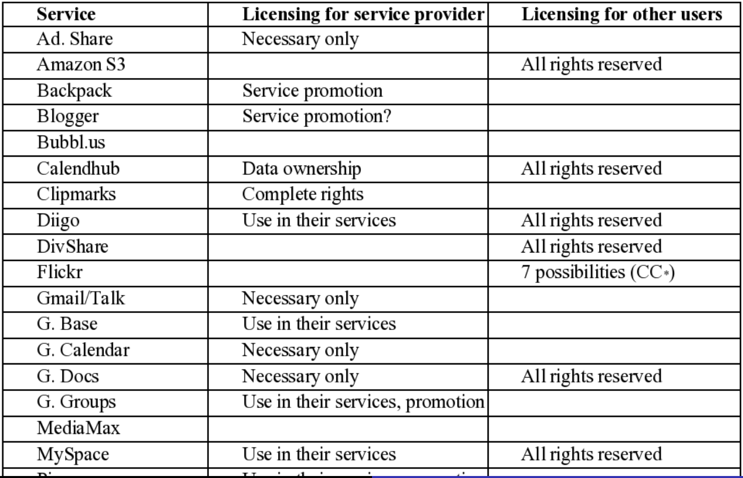
Terms of Service in General

- Though apps are accessible with no costs, still ToS are legally binding documents.
- Screening in 2007 (20 applications)
- All services reviewed except of one provided a ToS document
- Style of licensing documents varies from an itemized list to complex 10+ pages documents.
- Large operators $\Longrightarrow$ longer ToS (Google, Yahoo).
- Sometimes a **simplified version** of ToS is provided
- In very few cases, the ToS in the **local language** is offered.

User-supplied Data Licensing

- Web 2.0 apps are user-centric thus *nothing without user-provided data*
- The apps generally *disclaim* data ownership
- They need specific right to be able to process the data (present it, create thumbnails,. . .)
- Some apps claim further rights: non-private data for advertising or promotion
- License type sometimes left to end-user: selection from predefined licenses
- Often *Creative Commons* licenses are offered

Overview of Licensing (end 2007)



Service	Licensing for service provider	Licensing for other users
Ad. Share	Necessary only	
Amazon S3		All rights reserved
Backpack	Service promotion	
Blogger	Service promotion?	
Bubbl.us		
Calendhub	Data ownership	All rights reserved
Clipmarks	Complete rights	
Diigo	Use in their services	All rights reserved
DivShare		All rights reserved
Flickr		7 possibilities (CC∗)
Gmail/Talk	Necessary only	
G. Base	Use in their services	
G. Calendar	Necessary only	
G. Docs	Necessary only	All rights reserved
G. Groups	Use in their services, promotion	
MediaMax		
MySpace	Use in their services	All rights reserved

End-user Data Privacy

- apps treate personal and “private” data as confidential
- reveal them only under serious conditions

Authentication and Authorization

- rarely an inter-application AAI is used today except of
- **OpenID** enabling to authenticate at one place (identity provider) and register elsewhere (end-app)

Access Mode

1. **private** access (only owners can access the data)
2. **public** access (everybody on the Internet can see or even modify the data)
3. various kinds of **sharing** (explicitly invited people, implicitly “friends”, group members)

Content Limitations

- Some restrict content not to violate copyrights nor contain vulgarity, nudity, racism. . .
- Other impose other limitations due to application intention: Flickr to personally taken photos
- “Nonsenses”: Google Base which allowed textual data to be in English and German (2007)

Incorporating Functionality by 3^{rd} Party

Motto: *Mashups are welcome but not everything is allowed.*

(1) User accounts:

- not to be created in **automated** ways
- **by humans only**
- **forbid account sharing** among multiple people

Incorporating Functionality by 3^{rd} Party

(2) Data:

- any (automated) **harvesting generally prohibited**!
- How many scientific studies have emerged from such a process?

Incorporating Functionality by 3rd Party

(3) Software copyright:

- not allowed to **copy, reproduce, alter, modify, reverse engineer** nor
- create **derivative works.**
- Are mashups derivatives or just dependent applications?
- The only service allowing incorporating GUI is *Zoho*.

Identity Breaches

- **Fake profile** imitating a real person on Facebook or MySpace
- E.g. Alessandro Del Piero case: defamed as a Nazi-fan

Privacy Cases

- Attempts to regulate **file-sharing** services: they compromised FBI files, medical records, and SSNs
- Social networking apps:
  - The providers allow to create mashups getting full access to user profiles.
  - Unresponsible user behavior is often the cause: undesired authorizing 3rd party apps/mashups.
  - However, in some services the users cannot even completely prevent this!

Hi5 Example

Section 6. of hi5 ToS

By posting Content to any area of the Services, you automatically grant, and you represent and warrant that you have the right to grant, to hi5 an irrevocable, perpetual, non-exclusive, royalty-free and fully paid, worldwide license to reproduce, distribute, publicly display and perform (including by means of a digital audio transmission), and otherwise use Content and to prepare derivative works of, or incorporate into other works, such Content, and to grant and authorize sublicenses of the foregoing.

Spamming in Social Networks

- as popular as traditional mail spam
- lower chances to protect yourself
- serious legal cases with high sentences (Adam Guerbuez vs. Facebook $ 800 M)

Social Networks in Investigations

SN systems frequently used in criminal investigations:

- as an evidence of someone activity such as internet use
- Private pictures seem to be a good matter! (see Presley case)

Protocols and Formats

- **Data transfer protocols**: HTTP, SMTP, and XMPP,
- **SOA** architectural style: Web 2.0 machine accessible services (SOAP or REST)
- **Security**: SSL/TLS layer
- **Syndication**: RSS (Really Simple Syndication) and newer Atom.
- **End-user content**: HTML 5 with XML serialization will replace XHTML

Protocols and Formats

- **Object data**: At a lower level, JSON (JavaScript Object Notation) now belongs, beside XML, to the group of dominant serialization formats for object interchange between a Web 2.0 application/service and its client.
- **Plugability**: modern browsers (Firefox, MS IE, Safari) pluggable with various (platform-independent) modules
- **Search interfaces**: OpenSearch enables to locate the engine, its description, query and response formats
- **Access management**: OpenID, Enterprise Sign On Engine or proprietary authentication systems (like Yahoo! Or Googles)

Microformats

- “parasite” formats based on standards like HTML 4
- giving HTML attributes (elements) a specific interpretation
- $\Longrightarrow$ remains *standard-complaining* but is enhanced with *semantically rich data*

Microformats

Microformats cover many areas of general interest:

- People and organizations (**hCard, XFN**)
- Calendars and events (**hCalendar**)
- Ratings and reviews (**VoteLinks, hReview**)
- Licenses (**rel-license**)
- Tags, keywords, categories (**rel-tag**)
- Lists and outlines (**XOXO**).

Personal Identity Formats

Personal identity, profile, and activity data formats:

- **OAuth**
- **FOAF** (Friend-of-a-Friend)
- **Activity Streams**

Metadata

1. HTML **native** metadata (HEAD/META elements)
2. metadata encoded using **microformats** (such as time)
3. fully-fledged **RDF** metadata

Client requirements

- JavaScript-capable browser all apps are AJAX-based (Asynchronous JavaScript and XML)
- JSON replaces XML where overhead is an issue
- browser with Adobe Flash plug-in
- Even some mobile devices satisfy these requirements

Client requirements

Discrepancies between Web 2.0 applications and mobile devices capabilities but

- Smartphones pre-configured for **Google services**
- Cameras ready to upload to **YouTube** or still images to **Flickr**
- Enterprise IS need tailored solutions for BlackBerries, iPhones, Android devices, or Windows Mobile or Java-enhanced mobile phones
- W3C Mobile Web Initiative to enable “Mobile Enterprise 2.0”

Solutions for providers

- Hosting provider, application operator: pre-configured low-maintenance stacks – LAMP or WAMP all-in-one packages (Linux/Windows + Apache + MySQL + PHP)
- Enterprise-strength RedHat Enterprise Linux, Solaris, AIX, or Ubuntu with virtualization tools like Citrix, KVM, Sun xVM, or VMWare
- Market for pre-fabricated tailored virtual machine images for specific purposes (database servers, development machines, messaging servers) into a virtualization platform (VMWare, VirtualBox)

Development Environment for Web 2.0

Typical Web 2.0 application development environment — similar to enterprise development. Example: Java Enterprise Edition stack

- Platform's Software Development Kit (such as Java EE 5 SDK)
- Integrated Development Environment/s often highly customized (Web application development, SOAP web services, JBI, databases)
- Integrated or stand-alone build management system such as make, Ant, or Maven

Development Environment for Web 2.0

- **Source control system**: either a lightweight open-source tools (centralized systems like CVS, Subversion, or distributed Git) or a heavyweight commercial like IBM ClearCase
- Systems supporting **project** documentation, issue tracking, communication in team and with clients, change traceability, accountability, and reporting standalone (JiRA for issue tracking, IM tools for communication in teams) or integrated solutions for development management
- **Free** alternatives, such as Trac.
- **Team management systems**: IBM Rational Team Concert, Microsoft Office Project Server, or Oracle Project Portfolio Management

Frameworks

Web 2.0 application style: high interactivity, complex client-side, JavaScript heavy code

- **Google Web Toolkit** – generates server- and client-side from single-source
- **OpenLaszlo** – open source alternative.
- Numerous JavaScript and widget toolkits like **Yahoo! UI Library**, **Dojo**, or **jQuery**.

Integrated Community Services – GitHub Example

GitHub:

- Complete, secure project/development hosting service (*120 k users*)
- Hosting both *open-source* & *commercial* project
- source code management
- project documentation
- communication
- accounting features

Alternatives

- `Sourceforge.net`: largest, 230000 users, Sourceforge Enterprise suite for commercial projects
- `Codehaus.org`
- `Google Code`
- `Java.net`

Comparison: 3 Case Studies

1. Example A1: Application Hosting
2. Example A2: Application Hosting “NG”
3. Example B: Server Housing
4. Example C: Cloud Computing

Example A1: Application Hosting

Ruby on Rails hosting provider (Prague, CZ):

1. starting from $11/month
2. web domain
3. 1 GB storage
4. 3 Mongrel instances
5. MySQL
6. Git
7. SVN
8. (alternative: full service with unlimited applications, 20 GB storage, dedicated IP and server for $194)

Example A2: Application Hosting “NG”

Google App Engine:

- hosting client applications on Google’s infrastructure
- currently **Python**-based *Django* and **Java**
- persistent storage:
  - queries
  - sorting
  - transactions

Example B: Server Housing

Server housing company (Prague, CZ):

- server housing service from $111/month
- electricity consumption
- air conditioning
- good network connectivity
- non-stop access
- security
- monitoring

Example C: Cloud Computing

Elastic Cloud Computing service (EC2) by Amazon:
What the client/developer does?

1. Create an **Amazon Machine Image** (AMI) containing the applications, libraries, data, and configuration, or use pre-configured, templated images
2. Upload the image into Amazon S3 file hosting service
3. Select, configure, and monitor the operating system, storage, security and network access (such as a fixed IP) via a web interface
4. Pay for resources that are consumed (instance-hours, data transfer)
5. Establish a private cloud in the VPN
6. Select Availability Zone

Cloud Computing Characterized

What advantages does it have?

- **High availability**: EC2 SLA guaranties 99.95%
- **Wide spectrum of OS**: RedHat Enterprise Linux, Windows Server 2003, Oracle Enterprise Linux, Open Solaris
- **Pre-configured images**: DBMS like IBM DB2, Informix, Oracle 11g, MySQL, MS SQL Server
- **Enterprise application environments**: IBM WebSphere AS or JBoss Enterprise Application Platform
- IBMs Web 2.0 development environment **sMash**

Cloud Computing Characterized

Pricing offers several models (example for Windows):

- no basic fee in this model!
- from \$0.11 per hour (on-demand model, low CPU on Linux)
- to \$1.28 for high CPU for one instance (\$80 . . . \$920 / month)

Cloud Computing Characterized

Additional services charged separately:

- data transfer
- monitoring
- payment services
- load balancer

Considerations/1

Powerful computing infrastructure with zero initial capital investment

- More **flexible** than application hosting such as Google App Engine
- **Scalable**
- Range of instant enterprise level operating systems, DBMS, and application servers in EC2

Considerations/2

- Generally more secure & reliable than typical mid-range hosted solution
- Comparison: *more expensive* when the simplest application hosting service satisfies the client needs and *far more expensive* if a (rather limited) free GAE-like hosting is enough
- Simple ordering and management: self-service online

Quality Issues

Cloud computing raises questions concerning the quality of service (QoS)
Service level agreements (SLA) should cover:

- functional specification
- availability
- reliability
- security
- risk elimination in case of disaster
- service should enable data backup management

Risks and Pitfalls

- How our data is protected against security breaches, compromising?
- How is it backed-up?
- Do we have instant access to our data?
- Do we know where it is physically stored?
- Can we select the geographical locations?
- What happens after a disaster?
- What if the provider goes out of business?

What about open-source?

- Free / open source software (OSS) for the **infrastructure**
- Criticized by OSS advocates: lock-in effect danger by relying on proprietary solutions (with potentially growing prices in the future)
- **Transition** to other provider may be cumbersome (different technology, maintenance, processes)
- **Affero GPL**

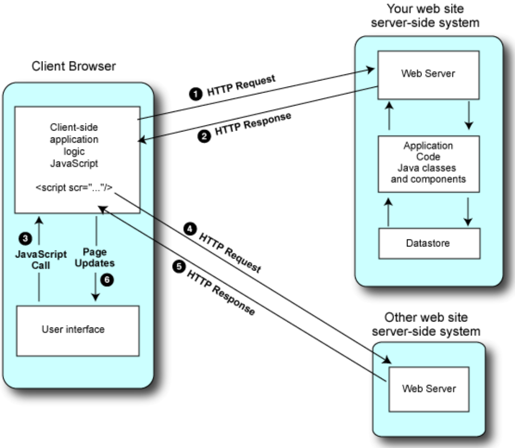
Mashup Development

1. Client-side Mashing by Scripting
2. Client-side Mashing by Widgets
3. Server-side Mashups
4. Content Syndication

Client-side Mashing by Scripting

- Server-side: mostly just serve the client code (JavaScript)
- Example: Google Maps upto 25 % of all mashups
- Adobe Flash can replace JavaScript
- map content is physically mashed in client browser

Client-side Mashing



Example: Script Load

Mashing up . . . (load scripts)

```
 <html xmlns="http://www.w3.org/1999/xhtml">
<head>
<title>Google Maps JavaScript API Example</title>
<script src="http://maps.google.com/maps?
    file=api;v=2;key=abcdefg;sensor=true_or_false"
    type="text/javascript"></script>
```

Example: Script Initialize

Mashing up . . . (initialize objects)

```
 <script type="text/javascript">
     function initialize() {
     if (GBrowserIsCompatible()) {
       var map = new
GMap2(document.getElementById("map_canvas"));
       map.setCenter(new GLatLng(37.4419,
-122.1419), 13);
       map.setUIToDefault();
     }
     }
</script>
</head>
```

Example: Script Use

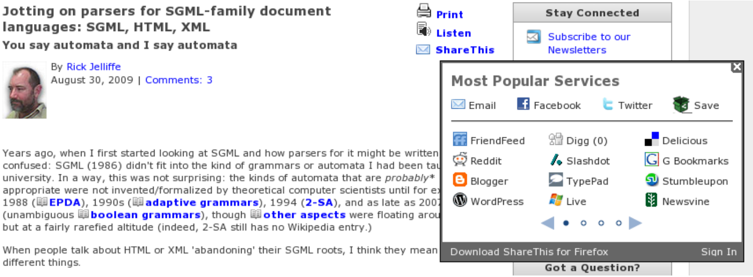
Mashing up . . . (use the map)

```
 <body onload="initialize()"onunload="GUnload()">
<div id="map_canvas"style="width: 500px; height:
300px"> </div>
</body>
</html>
```

Client-side Mashing by Widgets

- simpler than generic scripting: insert provided widget to the page, blog, wiki
- many components like m-m players (YouTube), news displayer, task lists. . .
- components to blogs (Blogger, Google Sites,. . .)
- targeted to end-users, not app developers

Example: Bookmarking Widget



Jotting on parsers for SGML-family document languages: SGML, HTML, XML

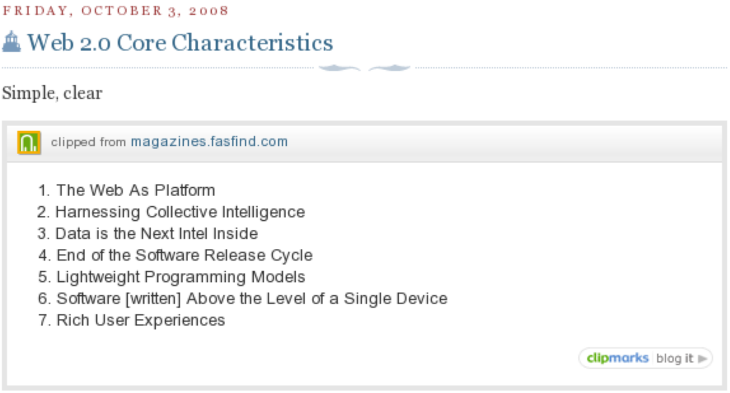
You say automata and I say automata

By Rick Jelliffe
August 30, 2009 | Comments: 3

Years ago, when I first started looking at SGML and how parsers for it might be written
confused: SGML (1986) didn't fit into the kind of grammars or automata I had been tau
university. In a way, this was not surprising: the kinds of automata that are *probably**
appropriate were not invented/formalized by theoretical computer scientists until for ex
1988 (**EPDA**), 1990s (**adaptive grammars**), 1994 (**2-SA**), and as late as 200
(unambiguous **boolean grammars**), though **other aspects** were floating arou
but at a fairly rarefied altitude (indeed, 2-SA still has no Wikipedia entry.)

When people talk about HTML or XML 'abandoning' their SGML roots, I think they mean
different things.

Example: Flickr Widget



FRIDAY, OCTOBER 3, 2008

Web 2.0 Core Characteristics

Simple, clear

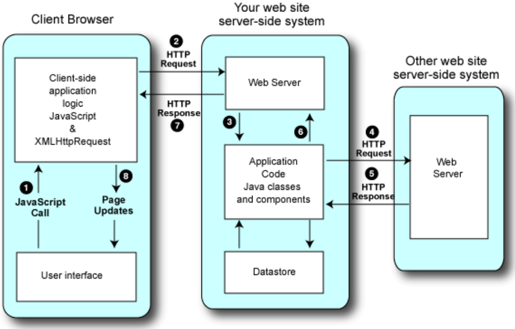
clipped from magazines.fasfind.com

1. The Web As Platform
2. Harnessing Collective Intelligence
3. Data is the Next Intel Inside
4. End of the Software Release Cycle
5. Lightweight Programming Models
6. Software [written] Above the Level of a Single Device
7. Rich User Experiences

Server-side Mashing

- concentrate computational **power and data storage** on server-side
- better combines if there is **unique data** on the server
- potential for hosting on a **cloud**
- content can also be **cached** there

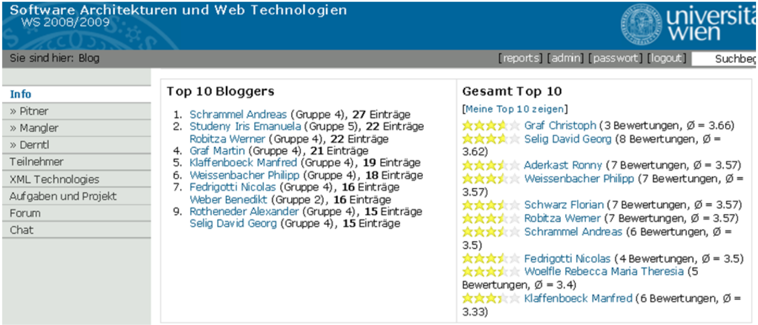
Server-side Mashing


Client Browser
Client-side application logic JavaScript & XMLHttpRequest
1 JavaScript Call
8 Page Updates
User interface
2 HTTP Request
HTTP Response 7
Your web site server-side system
Web Server
3
6
Application Code Java classes and components
Datastore
4 HTTP Request
5 HTTP Response
Other web site server-side system
Web Server

Content Syndication

- RSS/Atom syndication feeds
- covers not just "news"
- **feeds** organized into channels (e.g. one channel per blog)
- **items** (one item per new blog entry)
- basic meta/information about
- XML (even RDF) form
- allows clicking-thru to original resource

Example: Syndicated Content



Software Architekturen und Web Technologien
WS 2008/2009

universitä
wien

Sie sind hier: Blog

[reports] [admin] [passwort] [logout] Suchbeg

Info
» Pitner
» Mangler
» Derntl
Teilnehmer
XML Technologies
Aufgaben und Projekt
Forum
Chat

Top 10 Bloggers

1. Schrammel Andreas (Gruppe 4), **27** Einträge
2. Studeny Iris Emanuela (Gruppe 5), **22** Einträge
   Robitza Werner (Gruppe 4), **22** Einträge
4. Graf Martin (Gruppe 4), **21** Einträge
5. Klaffenboeck Manfred (Gruppe 4), **19** Einträge
6. Weissenbacher Philipp (Gruppe 4), **18** Einträge
7. Fedrigotti Nicolas (Gruppe 4), **16** Einträge
   Weber Benedikt (Gruppe 2), **16** Einträge
9. Rotheneder Alexander (Gruppe 4), **15** Einträge
   Selig David Georg (Gruppe 4), **15** Einträge

Gesamt Top 10

[Meine Top 10 zeigen]

Graf Christoph (3 Bewertungen, Ø = 3.66)
Selig David Georg (8 Bewertungen, Ø = 3.62)
Aderkast Ronny (7 Bewertungen, Ø = 3.57)
Weissenbacher Philipp (7 Bewertungen, Ø = 3.57)
Schwarz Florian (7 Bewertungen, Ø = 3.57)
Robitza Werner (7 Bewertungen, Ø = 3.57)
Schrammel Andreas (6 Bewertungen, Ø = 3.5)
Fedrigotti Nicolas (4 Bewertungen, Ø = 3.5)
Woelfle Rebecca Maria Theresia (5 Bewertungen, Ø = 3.4)
Klaffenboeck Manfred (6 Bewertungen, Ø = 3.33)

Development Tools and Platforms

- as an enterprise system
- specialized tools, e.g. IBM **sMash** (formerly *Mashup Starter Kit*), or **Mapbuilder**

Tools For Simple Workflows

fetch $\rightsquigarrow$ transform $\rightsquigarrow$ present

- **Dapper** system: developing and hosting mashups

Application Programming Interfaces

- bridge between service (application) and its consumer (another programm)
- defines protocols, formats, or provide stubs at the client side $\Longrightarrow$
- low- (e.g. HTTP-based communication protocol, formats) or high-level (OOP class libraries)
- may, but need not, be built upon Web Service standards
- class libraries most frequently for Java, Ruby, Python, and PHP, less C/C++

API based on Web Service standards

- **SOAP**: suite of standards developed since late 90s (industry-driven)
- **REST**: not a standard, but an architectural style (R. Fielding, 2001)

REST principles

- “back to the roots of the web”
- web applications manipulate **resources** (“objects”, “entities”, concrete or abstract)
- resources are identified by **URI**s
- resources can be created, read, updated, or deleted (CRUD operations)
- the operations directly map onto basic HTTP methods (POST, GET, PUT, DELETE)
- resources are manipulated via their **representations** (virtually anything: plaintext, XML, JSON, graphics, . . .)

Web Service Description

- **SOAP**: WSDL always
- **REST**: WSDL rarely, mostly a verbal, informal description

Service Lookup

1. automated service lookup: service directories (almost exclusively for SOAP and rare)
2. manual lookup: the most popular for REST
3. service directories: `ProgrammableWeb.com` (1400+ APIs, 4500+ mashups)

Example: Amazon WS

Both SOAP and REST interface

1. SOAP: less popular, higher overhead, simpler for some development environments (WSDL descriptions easily generate classes)
2. REST: very popular (80. . . 90 % of calls: 5 B/month)

Web 2.0 and Business

1. Generate profit FROM WEB 2.0
2. APPLY WEB 2.0 in traditional business

Monetizing Web 2.0

- interesting apps have potential to **attract users**,
- identify **niche-markets** by connecting people at global scale
- they still need appropriate means to leverage profit
- investors more sensitive after dot-com bubble

Per-unit Fee Model

- fixed per **product**
- content on **pay-per-download** basis like `http://i-legalne.cz`
- **quickly** available products
- more **precise selection** (one song instead of album)

Subscription Model

- fixed amount per **period of time**
- similar to other businesses (leasing), predictable
- combined to *Freemium*

Advertisement-based Model

- **core of Web 2.0 business**
- natural extension from classical media (TV, newspapers)
- dot-com bubble: so high costs that just 1/3 get covered!
- they did not target to **Long Tail**

Advertisement: Conditions

Providers should serve:

- carefully **selected**
- **context-based** (unlike idnes.cz)
- **unobtrusive** advertisements (unlike many)
- preferably in a plain form (cf. *Google Ad*)

Per-transaction Fee Model

- “man in the middle”
- fixed or percent-based from each transaction
- may be profitable for both sides
- either large quantities (`eBay.com`)
- or very specific goods (`Fler.cz`)

Revenue-sharing Model

- business-to-business agreement
- sharing access to users or customers
- slicing the profits

Combined Revenue Stream Models

- often THE MODEL
- Example: **Freemium** (free + premium) model

Web 2.0 in Enterprise

- Web 2.0 is *NOT* just open community matter
- also used to support “normal” business
- support internal processes

Enterprise 2.0

Joe Lennon (Core International):

Enterprise 2.0 is the concept of using tools and services that employ Web 2.0 techniques such as tagging, ratings, networking, RSS, and sharing in the context of the enterprise.

Enterprise 2.0 Patterns

- **Syndication**: syndicated access to information and services
- **Enterprise mashups**: rapid creation, sharing, and evaluation of applications to access and manipulate content and services
- **Marketing as a conversation**: end-user engagement through social networking, transforming marketing from broad-brush communication to thousands of individual conversations

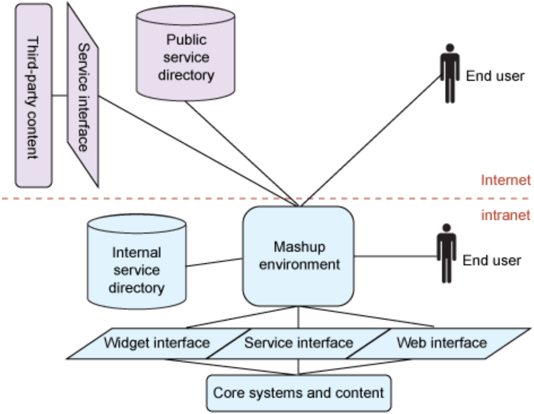
Enterprise 2.0 Patterns

- **Community exploitation**: lowering the communication costs and exploring the social networking capabilities to reach the Long Tail
- **Rich interfaces**: Improvements in real-world metaphors and visualization of complex data through rich-media UI

Enterprise 2.0 Tools

- IBM Lotus Connections
- Microsoft SharePoint

Enterprise Mashups



Enterprise 2.0 Survey

- surveying 14 big companies in various segments
- by Jakob Nielsen

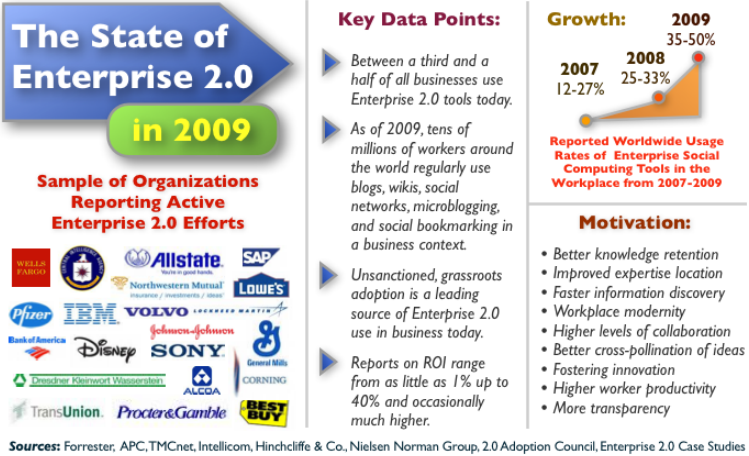
Survey Results

- frontier workers and young people are the driving force
- not tools but the changes in communication: they discover lacks
- but the quality of the tools matters
- enough value (own data or user-provided)
- do not duplicate work

Survey Results

- communication flow is not under control
- freedom must be accompanied with responsibility – no anonymity in enterprise social networks
- self-regulation works in enterprises too
- in average, the companies need about 3-5 years to adopt

State of Enterprise 2.0



The State of Enterprise 2.0 in 2009

Sample of Organizations Reporting Active Enterprise 2.0 Efforts

WELLS FARGO
Allstate
You're in good hands.
SAP
Northwestern Mutual
insurance / investments / ideas
LOWE'S
Pfizer
IBM
VOLVO
LOCKHEED MARTIN
Johnson-Johnson
Bank of America
Disney
SONY
General Mills
Dresdner Kleinwort Wasserstein
ALCOA
CORNING
TransUnion.
Procter&Gamble
BEST BUY

Key Data Points:

- Between a third and a half of all businesses use Enterprise 2.0 tools today.
- As of 2009, tens of millions of workers around the world regularly use blogs, wikis, social networks, microblogging, and social bookmarking in a business context.
- Unsanctioned, grassroots adoption is a leading source of Enterprise 2.0 use in business today.
- Reports on ROI range from as little as 1% up to 40% and occasionally much higher.

Growth:

Reported Worldwide Usage Rates of Enterprise Social Computing Tools in the Workplace from 2007-2009

Motivation:

- Better knowledge retention
- Improved expertise location
- Faster information discovery
- Workplace modernity
- Higher levels of collaboration
- Better cross-pollination of ideas
- Fostering innovation
- Higher worker productivity
- More transparency

Sources: Forrester, APC, TMCnet, Intellicom, Hinchcliffe & Co., Nielsen Norman Group, 2.0 Adoption Council, Enterprise 2.0 Case Studies

From http://blogs.zdnet.com/Hinchcliffe

Reality, not just a buzzword

- driving force to many businesses
- driving force to internet expansion
- new computing concepts
- future enhanced with semantics:
- **See you in Web 3.0!**

That´s all. . .

- Thanks for your attention!
- Questions now or later
- `tomp@fi.muni.cz`