

Web-Oriented Architecture (WOA)



Specially prepared for
QCon 2009 - London

Introduction

Dion Hinchcliffe

- ZDNet's Enterprise Web 2.0
 - <http://blogs.zdnet.com/Hinchcliffe>
- Social Computing Magazine – Editor-in-Chief
 - <http://socialcomputingmagazine.com>
- Enterprise 2.0 TV Show
 - <http://e2tvshow.com>
- **Hinchcliffe & Company**
 - <http://hinchcliffeandco.com>
 - <mailto:dion@hinchcliffeandco.com>
- **Web 2.0 University**
 - <http://web20university.com>
- TWITTER: DHINCHCLIFFE

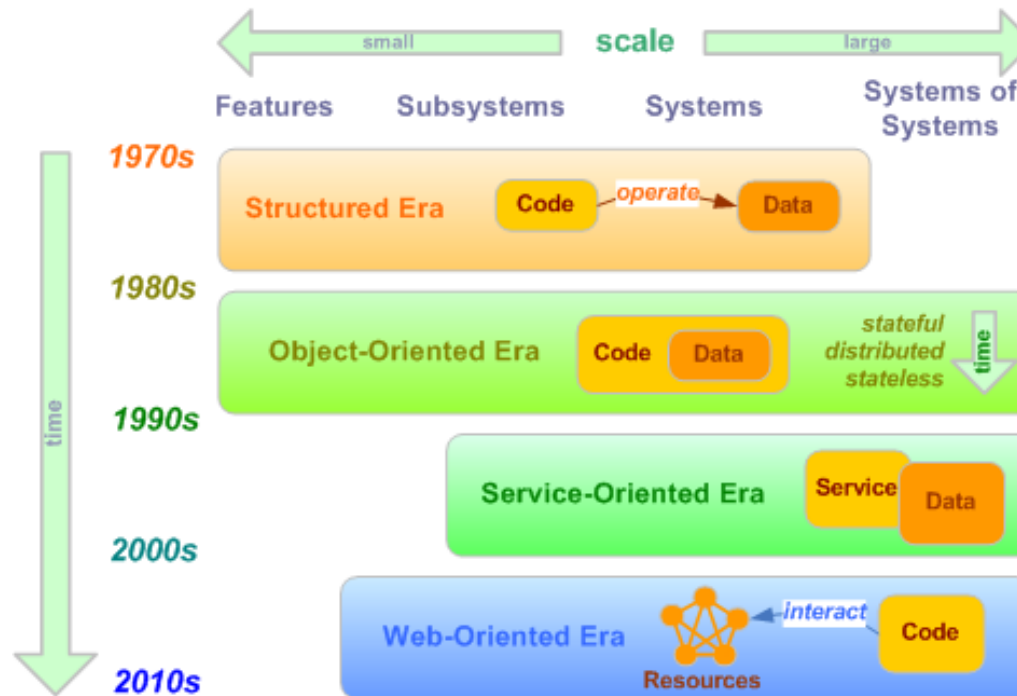


From the **REUTERS** TV Studio in Times Square
The Enterprise 2.0 TV Show



A Short History of Software

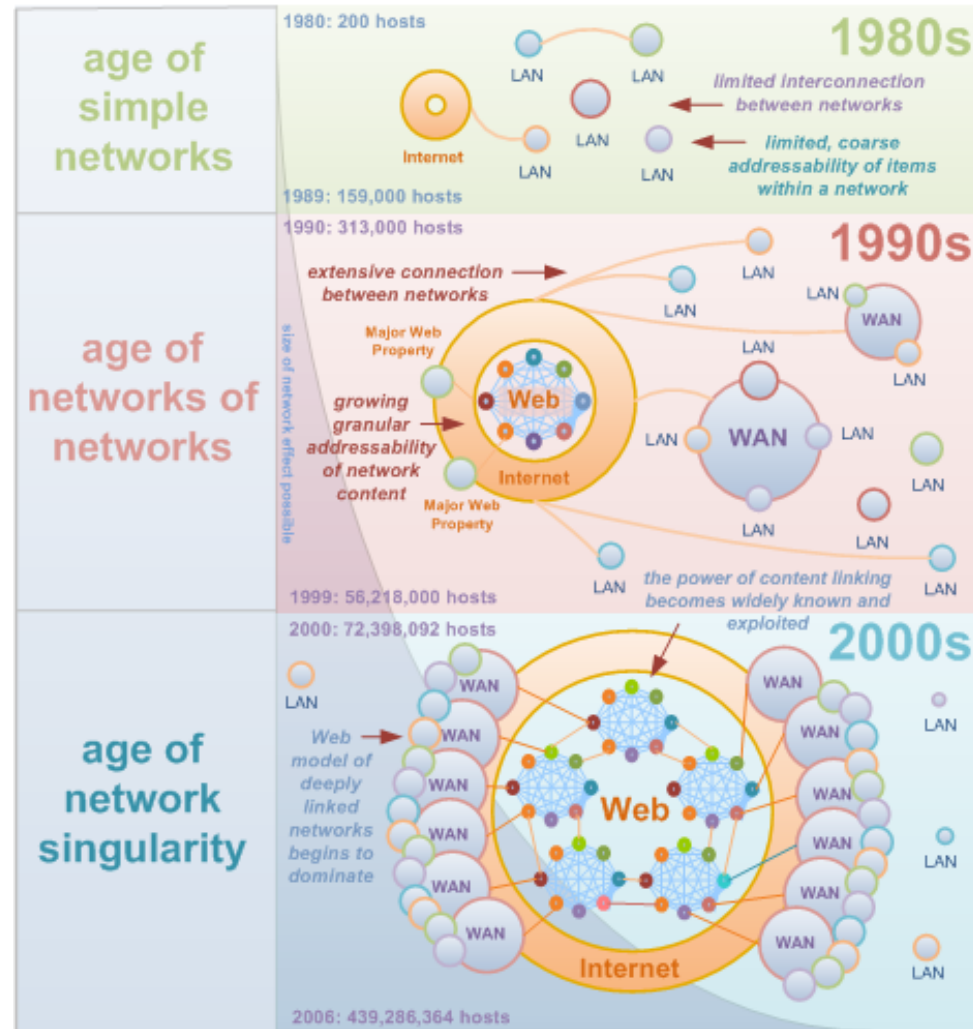
Popular Models for Developing and Integrating Software - 1970s to Now



An issue of complexity

- Developers have resolved most architectural issues historically by:
 - Choosing the right *data structures*;
 - Identifying or creating *algorithms*, and;
 - By applying the concept of *separation of concerns*.

The network began to further complicate the issue



Source: <http://web2.wsj2.com>

potential overall network effect



Integration has become a dominant, top-level activity

- “Research has shown that more than 30 percent of IT spending is on integration” – *Bitpipe*
- “Integration consumes more than 40% of IT resources” – *Aberdeen Group report*
- “Software integration is up to 60% of enterprise software projects.” - *Oracle*

SOA Definition

- SOA is a **modular software architecture**, and the modules are services designed to interact with each other.
 - Important Note: SOA also contains higher order constructs such as composite applications, orchestration, coordination, and more exist.
- SOAs are usually based on **open standards** to encourage automatic **interoperability** of services designed separately.

A Central Goal of SOA: Turning Applications Into Open Platforms

- Openly **exposing the features** of software and data to customers, end-users, partners, and suppliers for reuse and remixing
- This strategy requires documenting, encouraging, and **actively supporting the application as a platform**
 - Has serious governance implications
- Provide **legal, technical, and business** reasons to enable this :
 - Fair licensing, pricing, & support models
 - A vast array of services that provide data that uses need
 - A way to apply these services to business problems rapidly and inexpensively.

But existing SOA models have been challenged

- Most SOA initiatives are delivering low ROI to the business
- The reasons are many but boil down to:
 - SOA technologies have proven to have challenges compared to more successful models.
 - Top-down enterprise architecture moves slower than the environment changes.
 - Important avenues of SOA consumption and production points were often excluded from participation.
- SOA is still, however, the dominant organizing model for enterprise architecture.

Key trends on the Web today

- The growth of networked services with highly valuable open and “portable” data .
- Users by the tens of millions putting modular Web parts on their blogs and user profiles to host the pieces of the Web that they want to share.
- Businesses connecting to each other over the Internet via Web services by the hundreds of thousands.
- The increasing realization that there is limited business value in single applications...



The Openness of 2.0 Apps

- Building open platforms instead of stand-alone applications
- Forming self-distributing ecosystems
- Spreading products far beyond the boundaries of a site
 - APIs, widgets, badges, syndication -> mashups
- In other words: Being everywhere else on the network
- Building on the shoulder of giants
- Offering and consuming widgets, libraries, and APIs
- The automated mass servicing of markets of low demand content and functionality (The Long Tail)
 - Which represents the bulk of the demand

The Global SOA has surpassed our enterprise SOAs

- Some businesses have hundreds of thousands of users of their SOA
- Most are using WOA models for this
- Hundreds of companies have opened their SOA to the Web
 - Mostly startups or established Internet companies that understand the Web
 - But larger companies are beginning to understand this.



Mashup Directory

Total Mashups Listed

3757

Past 7 Days: 23

Past 30 Days: 75

Mashups/Day

3.0

7 Days Avg.: 3.3

30 Days Avg.: 2.5

API Directory

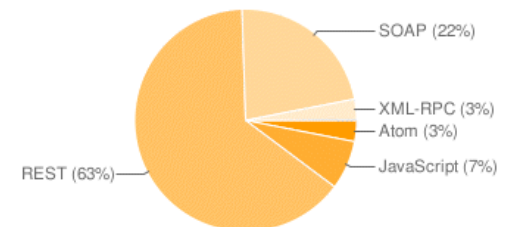
Total APIs

1177

Past 7 Days: 3

Past 30 Days: 38

Protocol Usage by APIs



Source: <http://programmableweb.com>

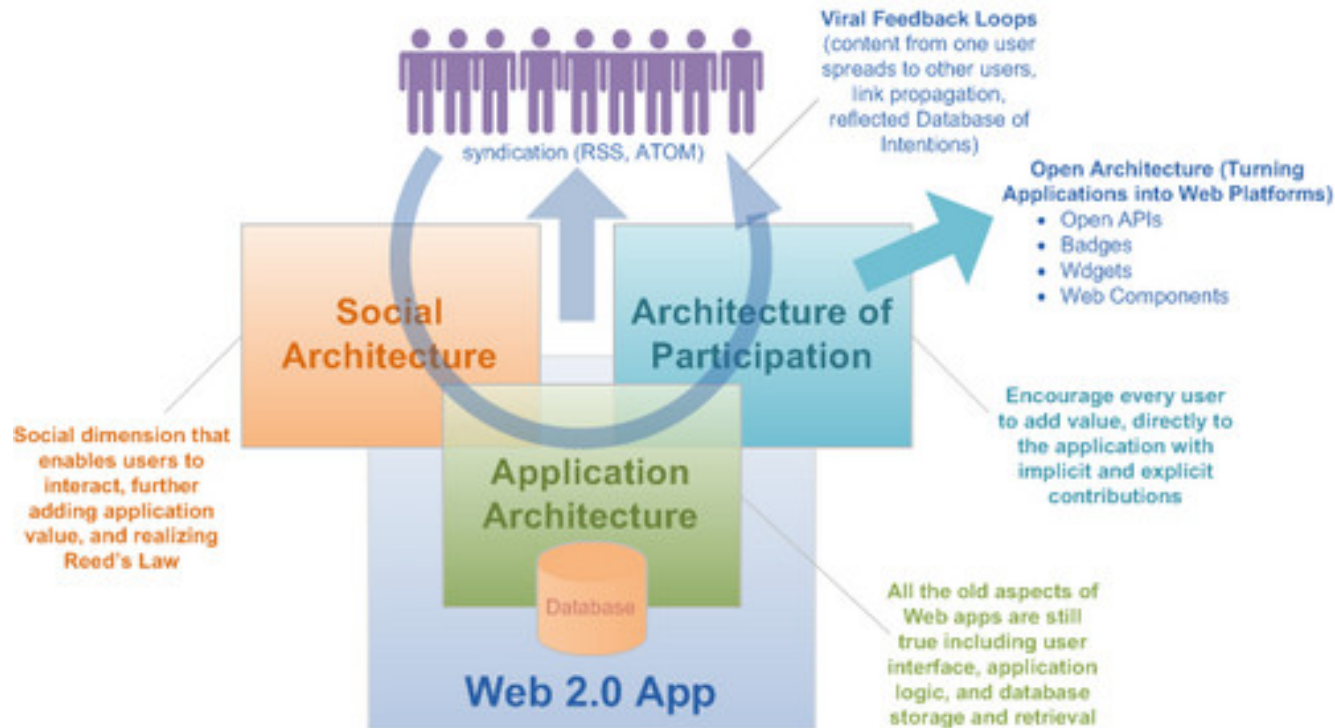
Examples

- Amazon and their highly successful Web Services Division (with hundreds of thousands of business consumers of their global SOA)
 - Over \$300 million in revenue last year
- Google and its numerous and varied open Web APIs from Google Maps to Open Social
- eBay and billions of dollars in listings it generates through its public SOA
- Applications like Twitter.com
 - Gets **10 times the use through its APIs** than from its user interface.
 - A new generation of applications that are primarily used via their Global SOA presence.
- Netflixprize.com, Gold Corp, and many others

WOA is for enterprises too

- Classical SOA is holding it's own but is not growing
- SOAP is in decline, with 54% planned use last year to 42% planned use this year
- Reported enterprise use of REST grew from 14% last year to 24% over the same timeframe.
- *Source:* 2009 Information Week study.

The traditional application model has evolved as well



It Also Means There's A Lot To Master Today

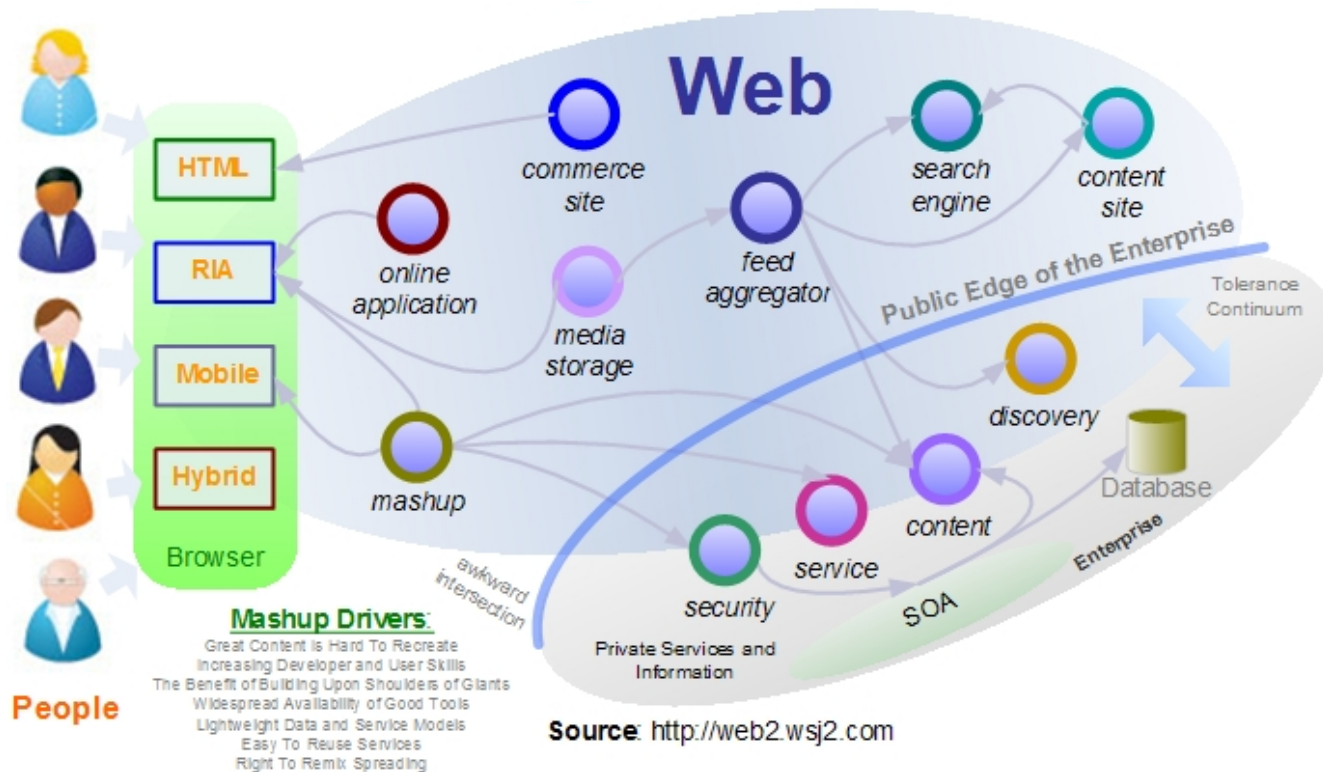
<ul style="list-style-type: none"> • Forums • Blogs • Wikis • Q&A • IM/Chat <p>Community</p>	<ul style="list-style-type: none"> • Friends Lists • Activity Streams • Invitation • Contact Import • Aggregation <p>Social Network</p>	<ul style="list-style-type: none"> • Open Addons • Content Contribution • Gestures • Collective Intelligence <p>Participation</p>
Social		

<ul style="list-style-type: none"> • REST • SOAP • JSON • RSS • Others <p>Open APIs</p>	<ul style="list-style-type: none"> • Badges • Web Widgets • Gadgets • OpenSocial • NetVibes UWA • SNS Apps <p>Widgets</p>	<ul style="list-style-type: none"> • RSS • ATOM • Bookmarking • Galleries • Marketplaces • Ping Services <p>Syndication</p>
Distribution		

<ul style="list-style-type: none"> • MySQL • PostgreSQL • SQL Server • Oracle • Others. <p>Database</p>	<ul style="list-style-type: none"> • Rails/Grails • PHP • Python • Java/J2EE • Others <p>App Tier</p>	<ul style="list-style-type: none"> • HTML • Ajax • Flash • Adobe AIR • Silverlight • JavaFX • Others <p>Client</p>	<ul style="list-style-type: none"> • Mobile • Virtual Worlds • Grid Hosting • PaaS • Content Replication <p>Misc.</p>
Application			

Identity	Computing	Location	Others
SMS	Queuing	Social Graph	
Storage	Storage	Advertising	
3rd Party Sourcing			

Networked applications today are deeply integrated

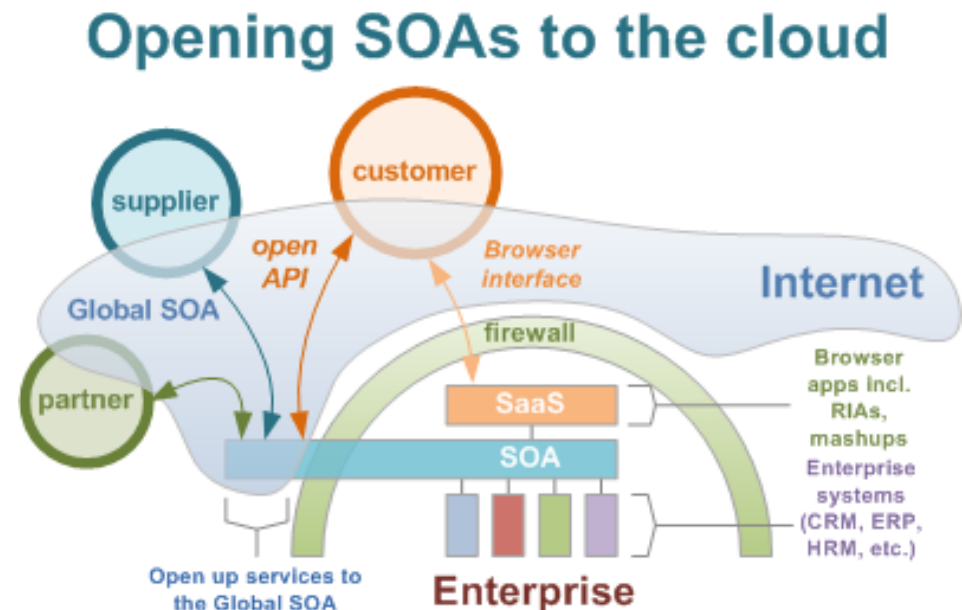


The motive force of 2.0: Harnessing the intrinsic power of networks

- “Networked applications that explicitly leverage network effects.” – Tim O’Reilly
- A network effect is when a good or service has more value the more than other people have it too.
- Two-way participation is the classic litmus test of a Web 2.0 system.

Demand for Widespread Cross-Organization Integration

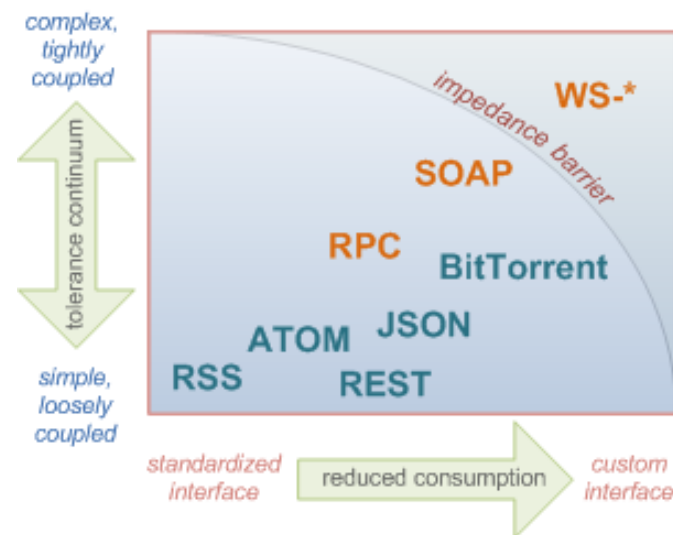
- “48 percent of the CIOs we surveyed said that they plan to implement service-oriented architectures for integration with external trading partners this year.”*
 – McKinsey & Co.



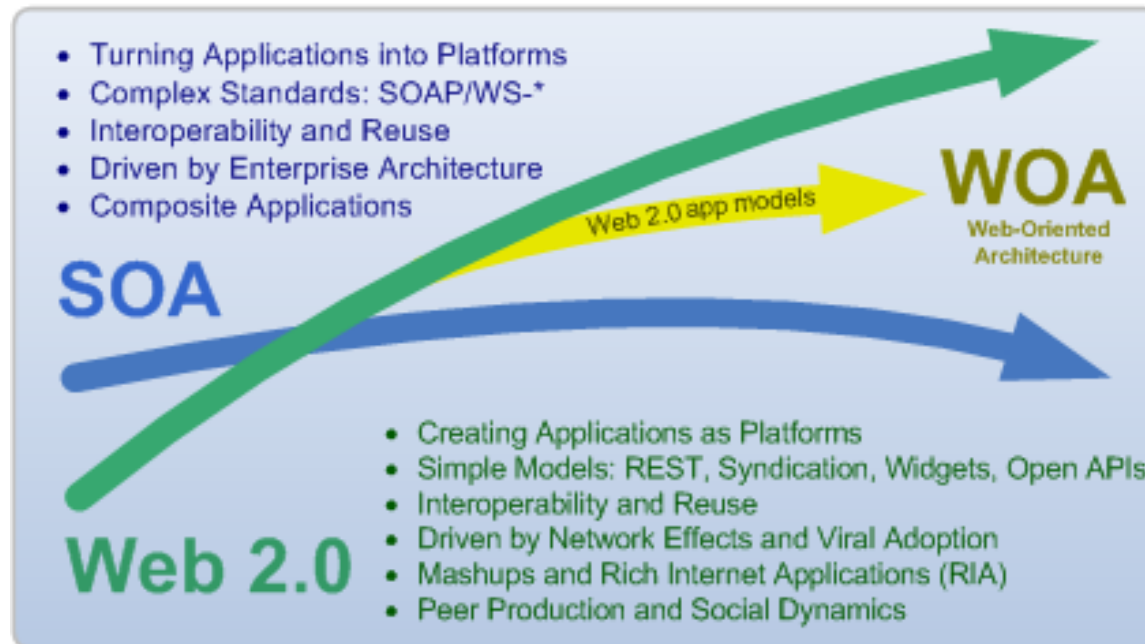
And we now have real-world experience with traditional SOA

- Classical SOA was an excellent first try but has a long list of challenges for the outcomes we desire today.
- The model of the Web has continued to teach us about how to structure information and services.

Options for opening a SOA in the cloud:
Traditional Web services
or **Web 2.0 APIs?**



The High Levels of Success of Web 2.0 Models for Creating Software Ecosystems Helped “Discover” WOA and Inform SOA



Strange Attractors: Similarities between Web 2.0 and SOA

- **Web 2.0**
 - Software as a service
 - Interoperability based on Web principles
 - Applications as platforms
 - Encourages unintended uses
 - Mashups
 - Rich user interfaces
 - Architecture of Participation
- **SOA**
 - Software as services
 - Interoperability based on heavyweight standards
 - Applications as platforms
 - Permits unintended uses
 - Composite Apps
 - Little user interface guidance
 - Little prescription of network participation

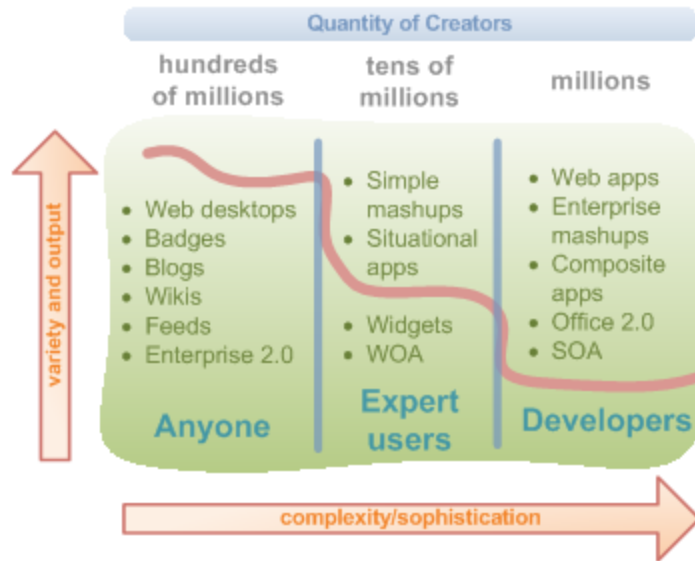
Web 2.0 and SOA Convergence

The Two Top-Level Organizing Principles in Modern Software Continue to Converge



Enabling New Consumption Scenarios

Everyone Assembling the Web: The Difficulty Curve of DIY Platforms, Tools, and Pieces

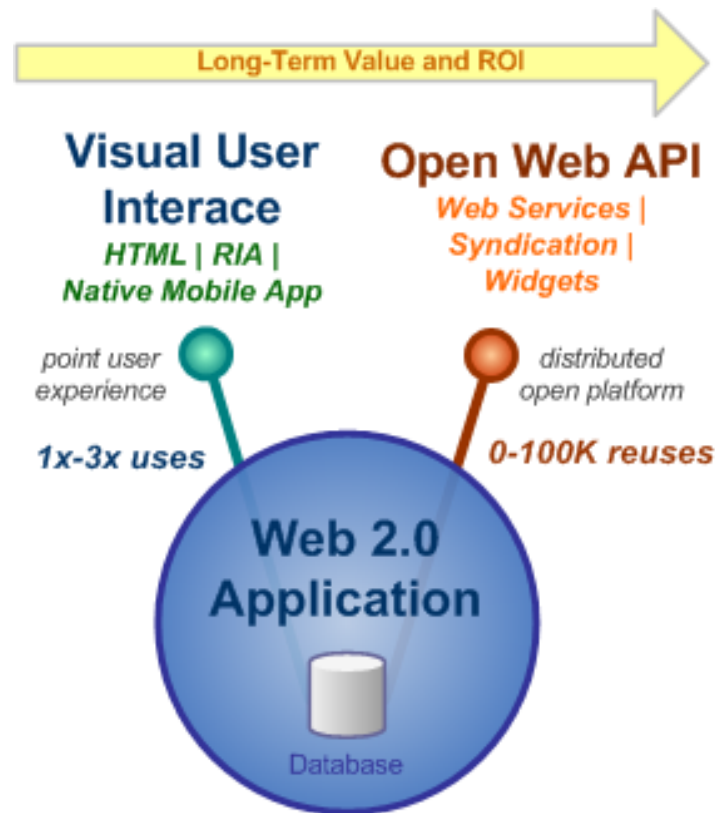


- Cut-and-Paste deployment anywhere on the Intranet
- Consumption of the SOA in any application that can use a URL
- Discovery of data via search
- Integration moves out of the spreadsheet

Next Generation Enterprises Are Leveraging Web 2.0

- A change in the way the Web is being used
- Innovative new customer interactions that shifts most control to users (partners, suppliers)
 - Control over software, data, structure, and processes
- Pervasive, viral, social and technical activities that embrace the intrinsic power of networks
- Driven by a move from push to pull-based systems

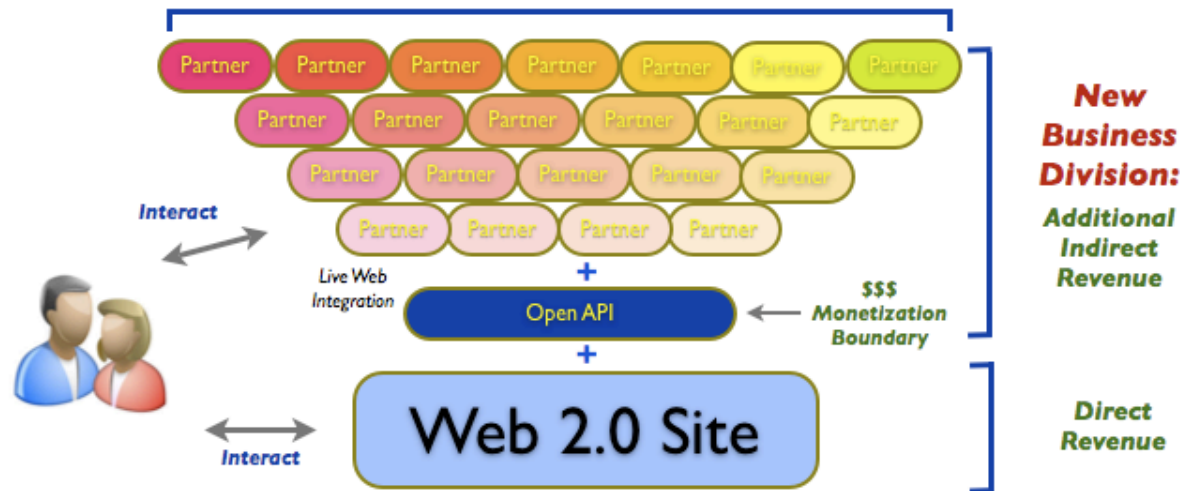
The New Competitive Advantage Online: Creating a Compelling Platform Play




How an Open WOA Creates a Platform

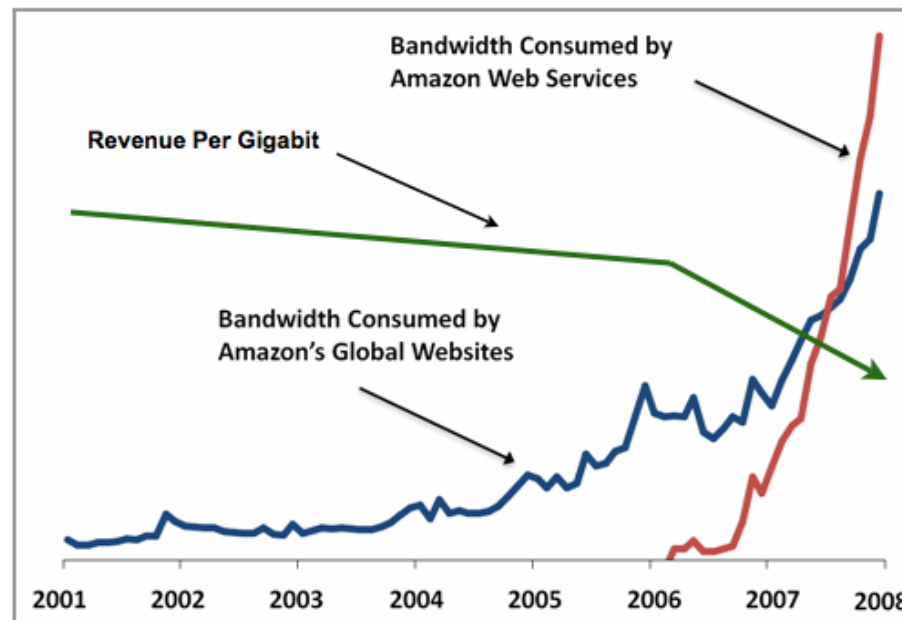
Opportunity: Going To the Customer and Open Web APIs

Tens of Thousands of Dynamic Web Partners



Result: Growth from Innovation

amazon.com. vs.  **amazon** :
webservices™ :
The Platform Overtakes the Web Site



Source: Jeff Bar, <http://aws.typepad.com/aws/2008/05/lots-of-bits.html>. Additions: Dion Hinchcliffe

Example: amazon.com.

- **1st Gen. Product: E-commerce store**
 - No differentiation
 - Scaling of a single site
 - Single site
- **2nd Gen. Product: E-commerce platform**
 - 55,000 partners using their e-commerce APIs live
 - Scaling of the Web
- **3rd Gen. Product: A series of Web platforms**
 - Simple Storage Service (S3)
 - Elastic Compute Cloud (EC2)
 - Mechanical Turk (Mturk)
 - Many others
 - 300K businesses build on top of what they've produced
- **2nd and 3rd generation platforms generate large net revenue**



S3

EC2



The New Philosophy of Product Distribution

- **Jakob's Law** states that "*users spend most of their time using other people's sites.*"
- You must design your products and services to leverage this fact deeply in the core of their design.

What is WOA?

The Structure of the Web

- HTTP is the foundational protocol
- When HTTP is used to transport *hypertext*, it can use links to refer to other parts of the Web
- These links are defined, like HTTP, by a standard and are known formally as URIs:
 - <http://tools.ietf.org/html/rfc3986>

URIs according to w3.org

- Uniform Resource Identifiers (URIs, aka URLs) are short strings that identify resources in the Web:
 - documents, images, downloadable files, services, electronic mailboxes, and other resources.
- They make resources available under a variety of naming schemes and access methods such as HTTP, FTP, and Internet mail addressable in the same simple way. They reduce the tedium of "log in to this server, then issue this magic command ..." *down to a single click.*

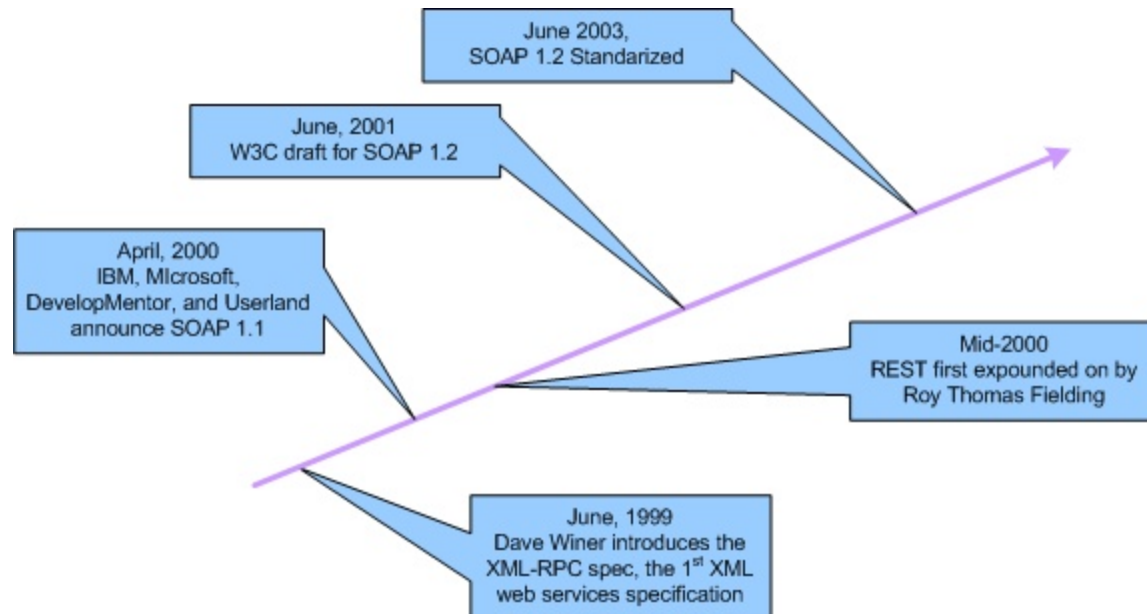
The Short Version

HTTP + URIs = Web

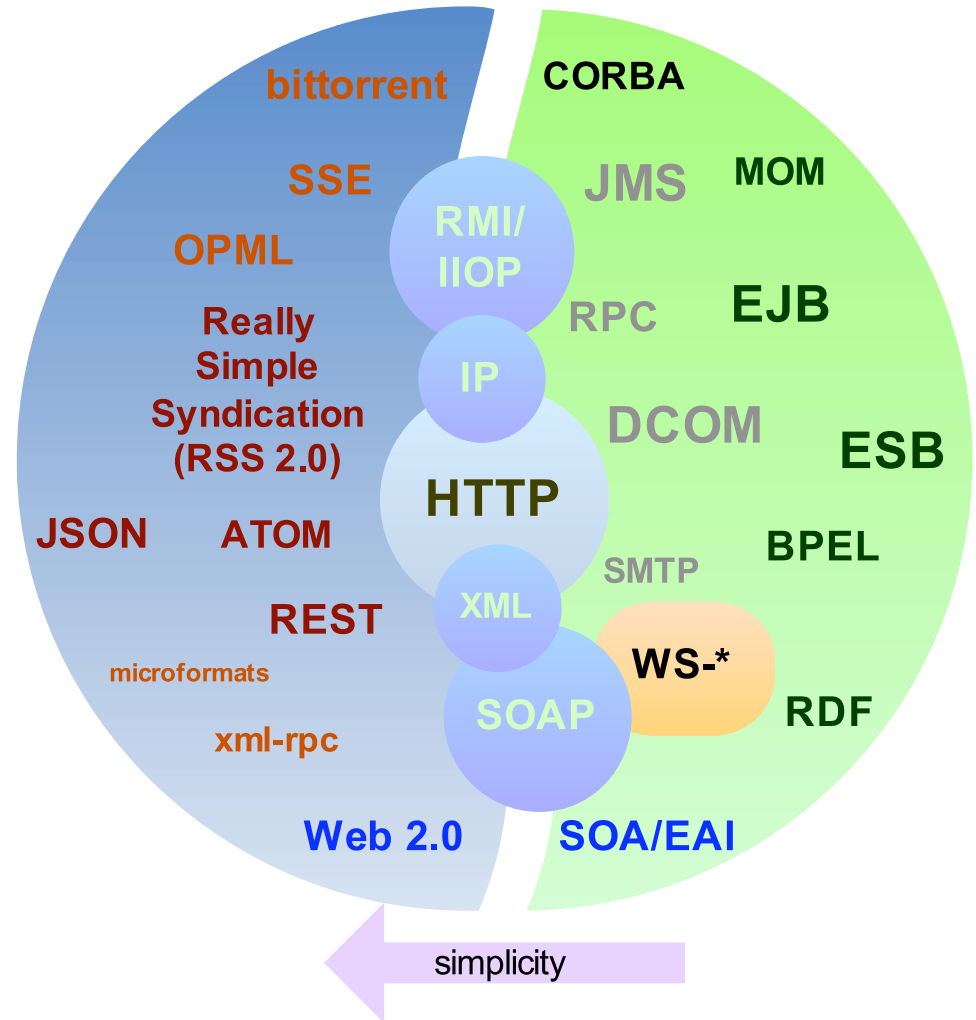
The use of HTTP has evolved extensively...

- There is no limit to the kind of data that can be transferred via HTTP
 - HTTP was originally used for transferring Web pages, images, and other simple resources
 - But now it's used to transfer pages, data, programs, video streams, and virtually everything else
- There is no limit to the amount of data that can be transferred via HTTP
 - Server-side push of HTTP data is now possible by technique
- Has proven to be highly secure and scale to global use

However, we originally thought open data transport needed something else



What service model is best for an WOA?



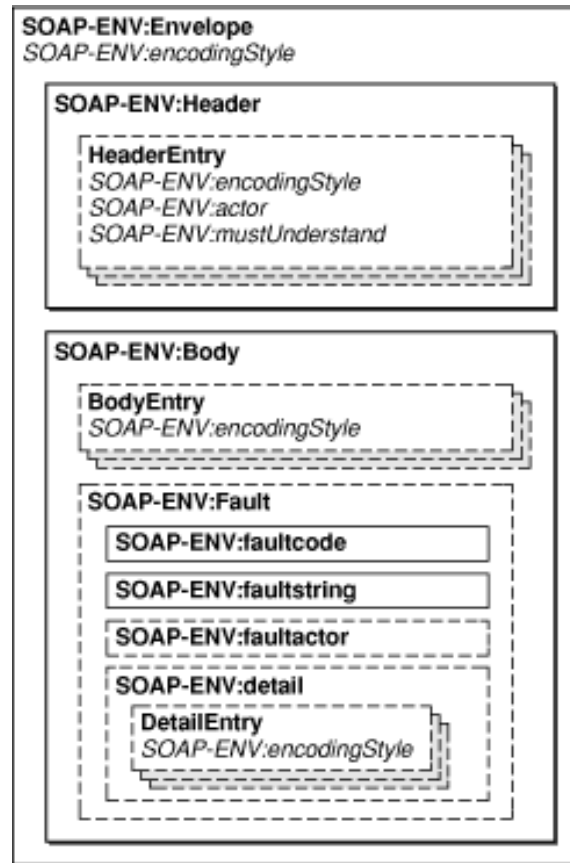
SOAP

- **Message oriented system**
- **Everything is inside the message**
 - Headers
 - Reliability
 - Security information
- **Transport neutral**
- **XML**

SOAP Example

```
<soap:Envelope>  
  <soap:Header>  
    <addressing:To>  
      http://some/service  
    </addressing:To>  
  </soap:Header>  
  <soap:Body>  
    <hello>Hello!</hello>  
  </soap:Body>  
</soap:Envelope>
```

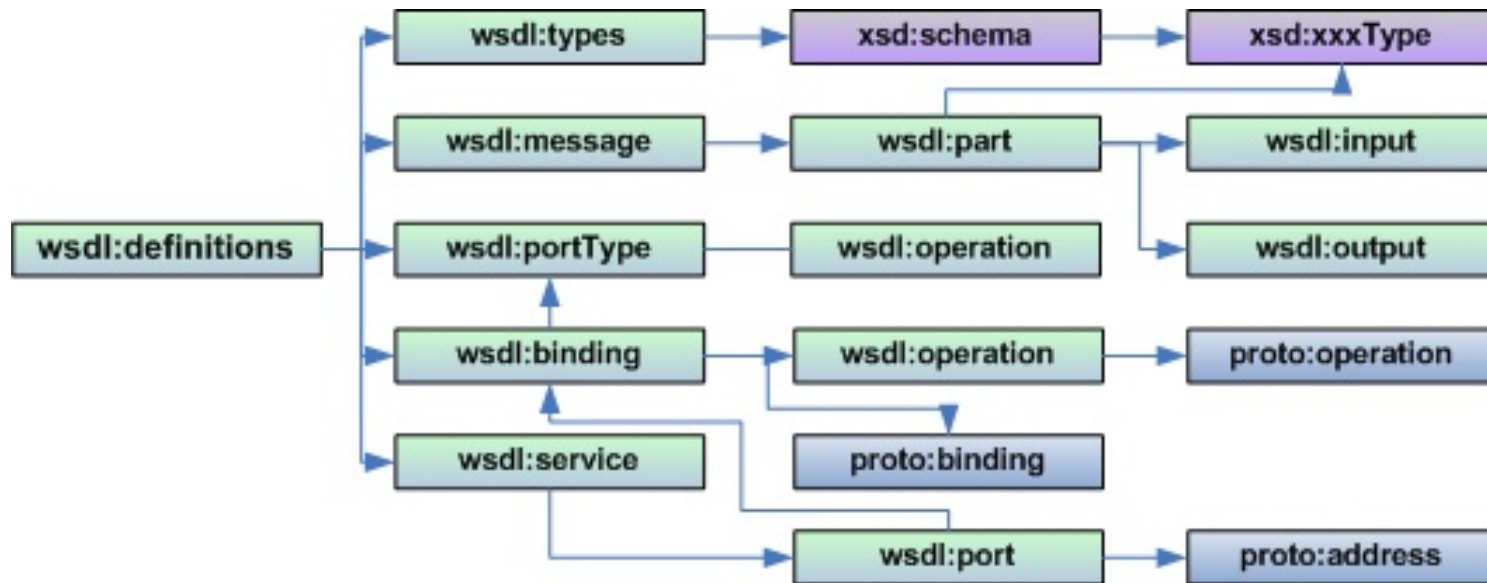
SOAP Message Structure



WSDL

- **Web Service Description Language**
- **Lists operations, services, data types so that other platforms can learn how to interact with the service**
 - Like an interface in Java, but cross platform
- **Data types are described by XML Schemas (XSD)**
- **Forms the service “contract”**

High level structure of WSDL

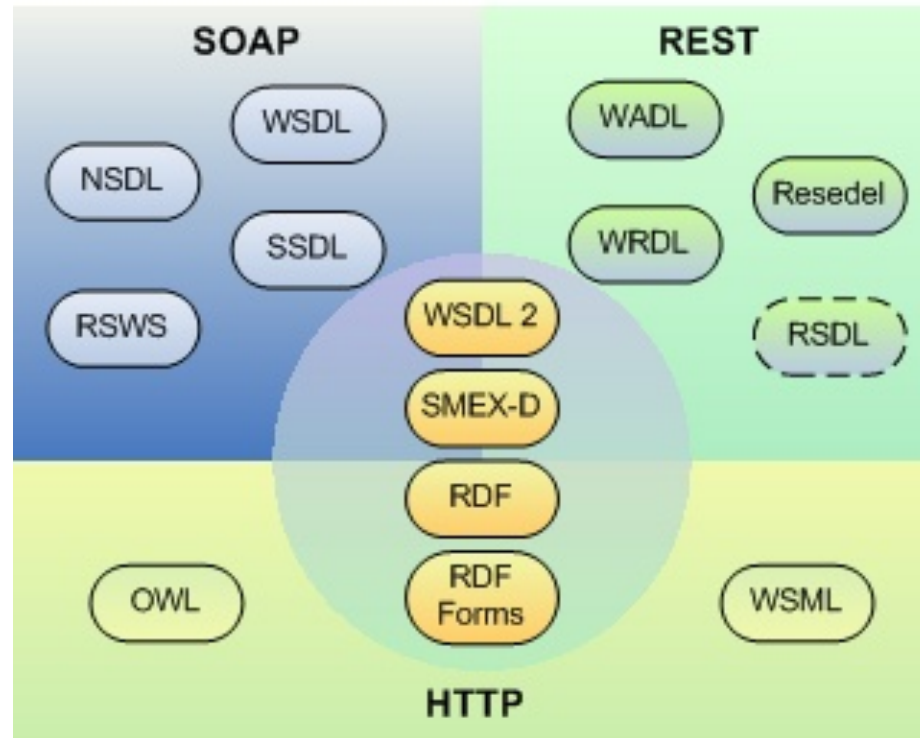


More on WSDL

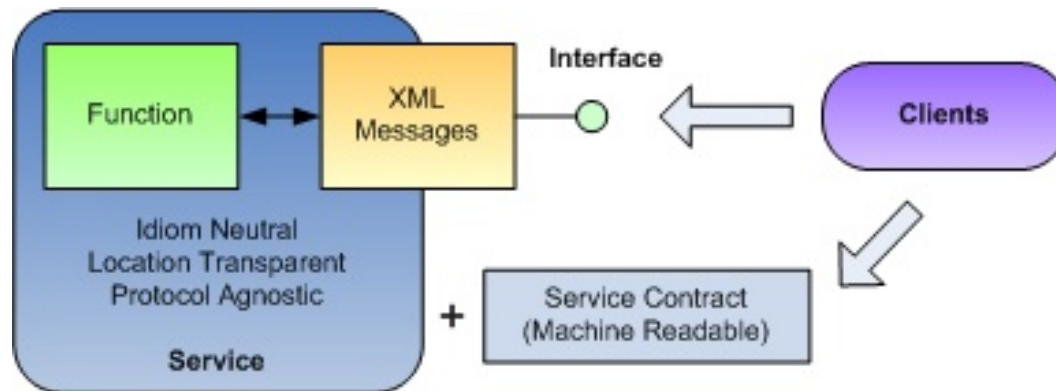
- Can be complex (even huge)
 - WSDL is obtuse
- WS-* specifications are far from pervasive thus services are often not actually interoperable

We tried to fix WSDL

Web Service Description Languages (*Partial*)



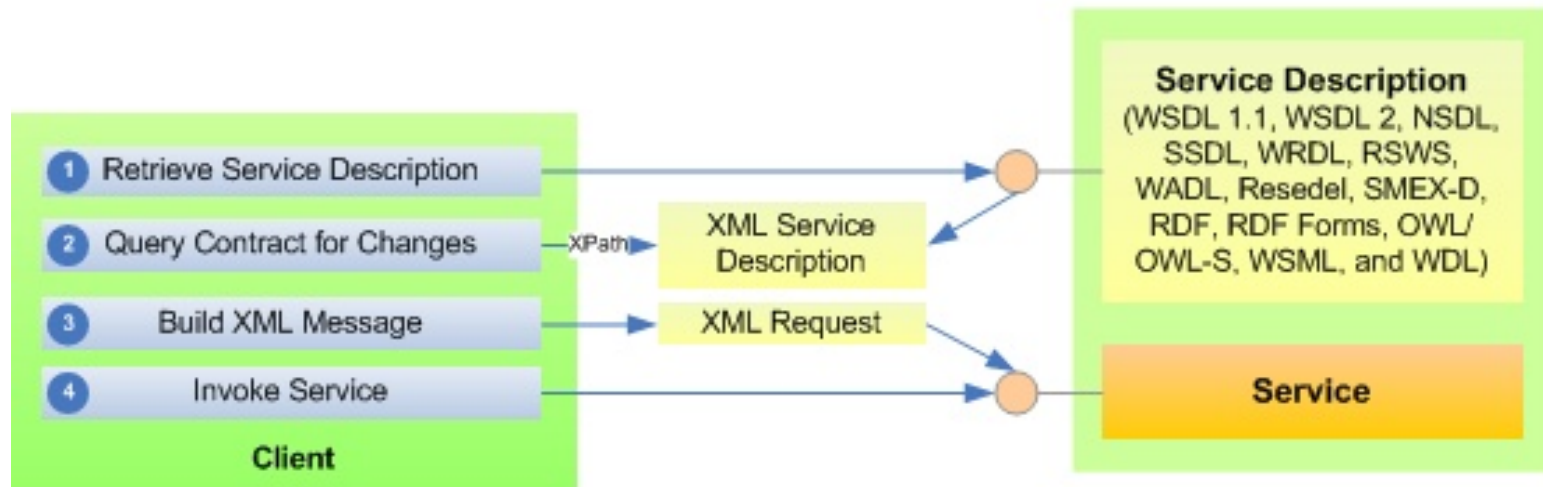
The original vision



Service-Oriented: Focusing on delivering function via open, contract-based message exchange, while being location transparent, protocol agnostic, and free of implementation idioms.

Separate Contract and Service

Service-Oriented and SDL-Neutral Service Interaction Model



Become this vision...



Vs.

RESTful Services

- Representation State Transfer
 - Roy Fielding coined the term for his thesis which sought to provide the architectural basis for HTTP
- But what *is REST exactly?*

It's all about resources

- In REST, everything on the network is a resource
 - Resources are addressable via URIs
 - Resources are self descriptive
 - Typically through content types (“application/xml”) and sometimes the resource body (i.e. an XML QName)
- Resources are *stateless*
- Resources are manipulated via HTTP *verbs* and the uniform interface

The Uniform Interface

Uniform



Get(URI)



Put(URI, Resource)



Delete(URI)

Non Uniform



getCustomer()



updateCustomer(Customer)



delete(customerId);

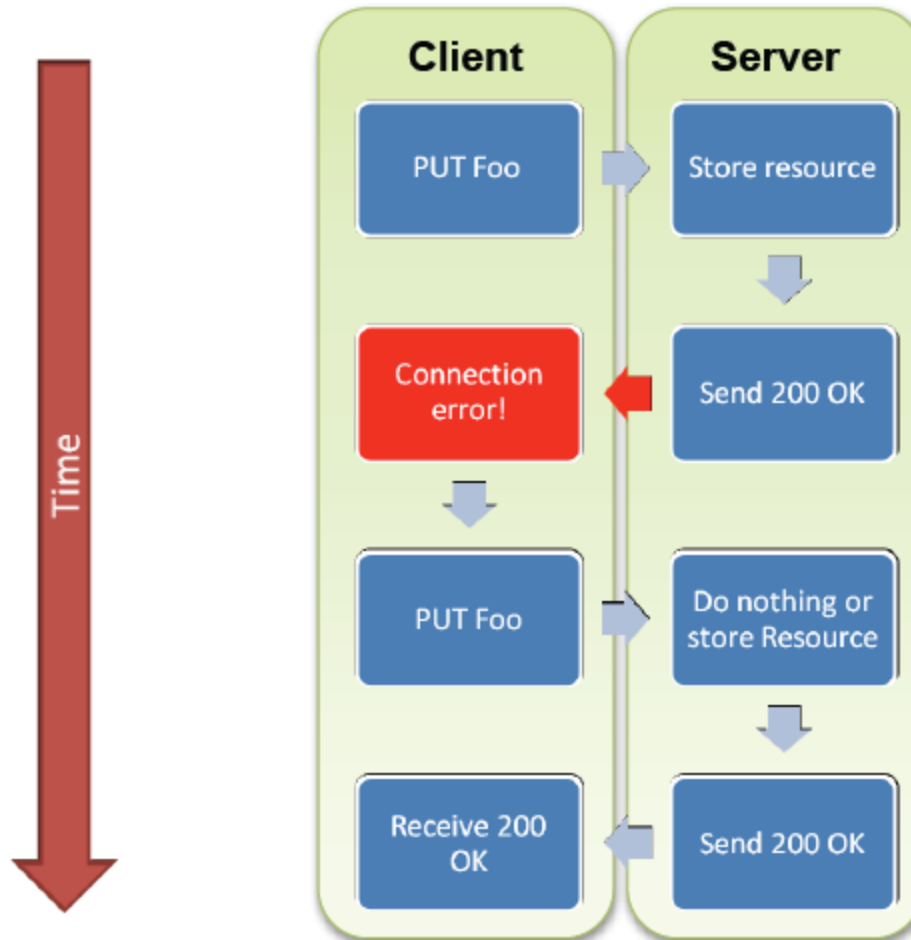
Hypertext and linking

- We don't want "keys", we want links!
- Resources are hypertext
 - Hypertext is just data with links to other resources
- Data model refers to other application states via links
- This is possible because of the uniform interface.
- *Key Point:* There is no need to know different ways to get different types of entities!
- Client can simply navigate to different resources
- The uniform interface allowing different types greatly lowers the barrier to consumption.

Getting a network resource

- GET is safe
 - If original GET fails, just try, try again
- Note that not all HTTP resources follow this rule properly
 - The rule: *Be liberal in what you accept and strict in what you transmit*

Updating a resource



Creating Resources

```
POST /entries
Host: acme.com
...
```

Client

```
HTTP/1.1 201 Created
Date: ...
Content-Length: 0
Location:
  http://acme.com/entries/1
...
```

Server

```
PUT /entries/1
Host: acme.com
Content-Type: ...
Content-Length: ...
```

Some data...

Client

```
HTTP/1.1 200 OK
...
```

Server

HTTP is scalable

- Just look at the size of WWW today
- HTTP has built in mechanisms for caching resources
 - All communication is stateless
 - Session state is kept on the Client
- Client is responsible for transitioning to new states
- States are represented by URIs

Basic Guidelines

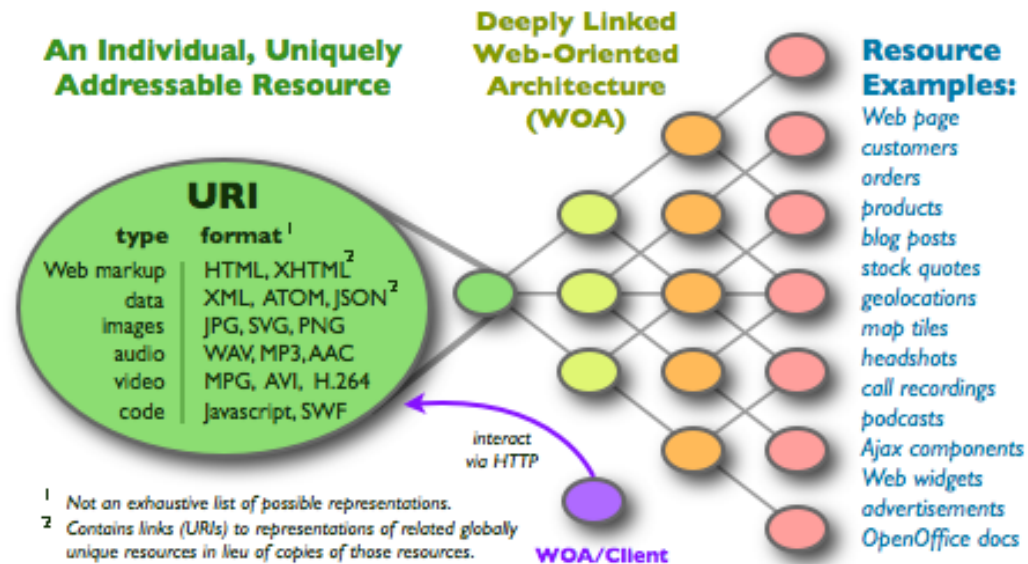
- Information is stored as resources (like a Web page) instead of a traditional RPC Web service.
- Every resource can be located via a globally unique address known as a Universal Resource Identifier.
- Resources are manipulated by HTTP verbs (GET, PUT, POST, DELETE)
- Manipulation of network resources is performed solely by *components* on the network (essentially browsers and other Web servers).
- Access to resources must be layered and not require more than local knowledge of the network.
- Full details at: <http://hinchcliffe.org/archive/2008/02/27/16617.aspx>

Advantages to REST

- **Linkability: Everything is accessible and addressable via a simple URL**
 - Countless benefits from discoverability to consumption to analytics and link ecosystems
- **Simplicity (easy to provide/consume)**
- **Uniform interface**
- **Testable via a browser**
- **No WSDL or traditional service contracts**
 - Allows minimal surface area contract checking (more on this)
- **Is easily consumed by dynamic languages such as Javascript or Ruby**
- **High performance: Works well with caches**

WOA: An organic service fabric

**SOA Reshaped by the Web 2.0 Era:
Granular, Radically Distributed, Web-
Oriented, Open, Highly Consumable**



Source: Dion Hinchcliffe. <http://hinchcliffe.org>. Some Rights Reserved, 2008.

Is WOA just REST?

- REST is the foundational network protocol for WOA
- WOA also includes a set of consumption models including such things as mashups and Web widgets (explored here later)
- WOA also includes a set of widely-used standards and technologies for data representation, data exchange and portability that REST itself does not encompass, but is necessary for a proper and complete architectural definition.
 - <http://dataportability.com> is a partial list of the “other” WOA standards. It is not canonical given that WOA is emergent and not defined by a vendor or standards body.
 - Identity, security, Web applications model, new distribution models are not accounted for by REST alone either.

The WOA “Stack”



Is WOA also SOA?

- Yes, it follows all the basic tenets of SOA including the 8 principles
- So good SOA, and perhaps the best SOA, can be implemented with WOA

Principles of Service-Oriented Architecture

- Service reusability
- Service contract
- Service loose coupling
- Service abstraction
- Service composability
- Service autonomy
- Service statelessness
- Service discoverability

<http://ServiceOrientation.org>

Differences with traditional SOA #1

- SOAs tend to have a small and well-defined set of endpoints through which many types of data and data instances can pass.
- WOAs tend to have a very large and open-ended number of endpoints; one for each individual resource. Not an endpoint for each type of resource, but a URI-identified endpoint for each and every resource instance.

Differences with traditional SOA #2

- Traditional SOA builds a messaging layer above HTTP using SOAP and WS-* that provides unique and sometimes prohibitive constraints to the Web developer.
- WOA finds HTTP and related transfer mechanisms to be the ideal layer of abstraction for most applications.

Differences with traditional SOA #3

- SOA was designed from the top-down by vendors to be tool friendly
- WOA was emerged from the bottom up from the Web developer community naturally. It has the best support in simple procedural code and an XML parser and is widely support by modern Web stacks.

Differences with traditional SOA #4

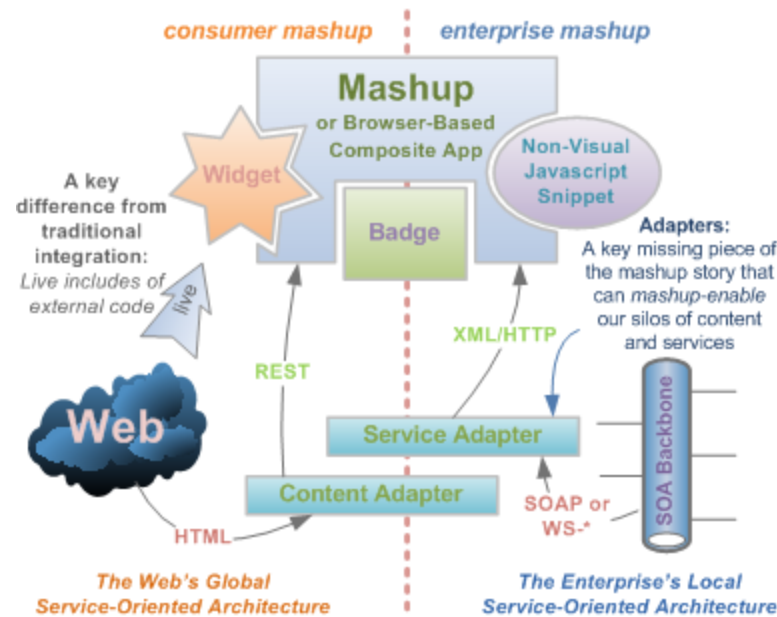
- SOA uses WS-Security and other sophisticated standards for security, while WOA tends to just use HTTPS, OAuth, and HMAC-SHA-1.

Differences with traditional SOA #4

- SOA uses WS-Security and other sophisticated standards for security, while WOA tends to just use HTTPS, OAuth, and HMAC-SHA-1.

Situating WOA in the enterprise:

Low Barrier, High Velocity Integration:
Using the Strengths of the Web to Remix and Connect our
Content and Functionality From *Anywhere* to *Anywhere Else*



State Representation

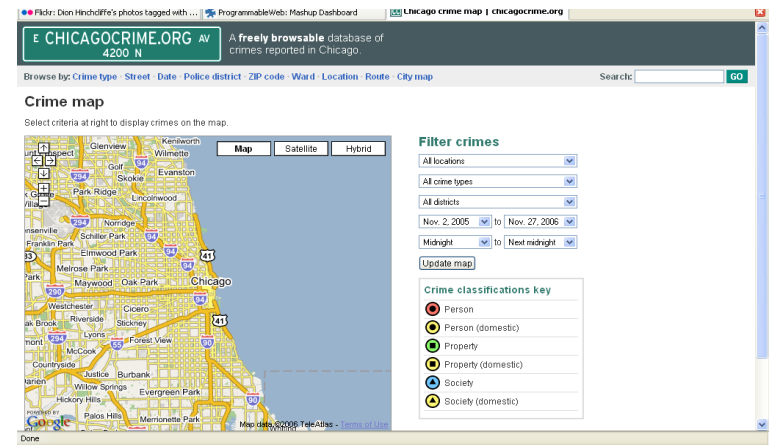
- XML is most common
- JSON is becoming very popular
- YAML - <http://yaml.org>
- Fast Infoset - <http://java.sun.com/developer/technicalArticles/xml/fastinfoset/>
- <http://jsonml.org/>

What is a “mashup”?

- “A **mashup** is a Web site or Web application that seamlessly combines content from more than one source into an integrated experience.” - *Wikipedia*
- Content used in mashups is usually sourced from a 3rd party via a **public interface (API)**
- Other methods of sourcing content for mashups include **Web feeds** (e.g. **RSS** or **Atom**), and **JavaScript/Flash “widgets”**

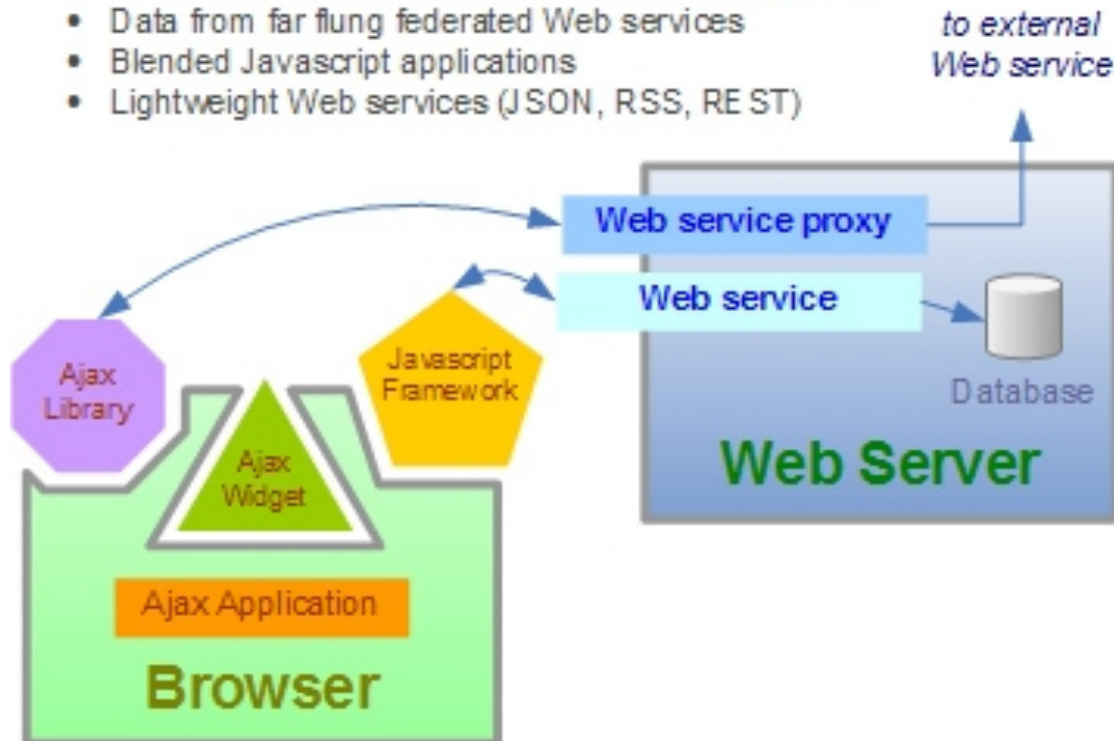
Mashup Examples

- HousingMaps.com
- ChicagoCrime.org
- Flashearth.com
- Zillow.com
- Pickaprof.com
- Tagbulb.com



Online Software with the Next Generation of Web Components

- Browser used for composition
- Informal yet effective integration models
- Zero footprint & low admin, with seamless upgrades
- Data from far flung federated Web services
- Blended Javascript applications
- Lightweight Web services (JSON, RSS, REST)



Web apps that integrate in real-time...

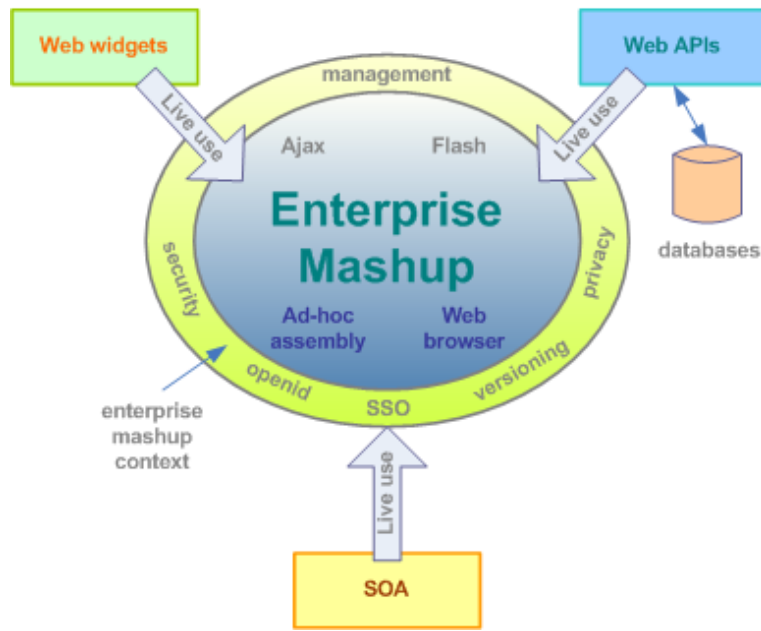
User Generated Software:
Content and functionality dynamically embedded in Web pages



- Example:
<http://pipes.yahoo.com>

The Focus: Rapid Business Solutions

Enterprise Mashups:
Poised for Growth As the Development
Model Matures



- Full resources of the Web and the Intranet
- Enterprise context around management, security, privacy, etc.
- Gives everyone in the organization the ability to leverage the SOA.
- Lightweight, simple model using WOA.

Why is Web-Oriented Architecture Important for Mashups?

- The most commonly used Web service approaches “in the wild” turn out to be the ones based on the “grain” of the Web:
 - Representation State Transfer, or **REST**.
 - Created by Roy Fielding, the co-creator of HTTP, the fundamental protocol of the Web.
 - Designed to fit naturally into Internet architecture
 - Extremely simple, not a standard, just a style of using **HTTP**
 - Fully embraces the workings of HTTP and uses its verbs (GET, PUT, POST, DELETE) on top of a granular, sensical URL structure to indicate what is to happen.
 - **RSS** and **ATOM**
 - ATOM *is* REST

What does an organization with WOA “look” like?

- A rich web of REST resources.
 - Simple tools to weave the Web of resources into new applications.
 - Highly consumable and reusable WOA “parts” including widgets, gadgets, and embedded social apps.
- Open Web APIs exposed on the Internet to ad hoc partners.

Major Opportunities in 2009

- Strategically move IT infrastructure to the cloud.
- Embrace new low-cost economic models for SOA.
- Reduce application development and integration time/ expenditures with new platforms and techniques.
- Open your supply chain to partners on the Web.



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

WOA Conclusions

- WOA is already the dominant model for networked applications today
- But it has largely emerged under the radar (no big vendors behind it)
- WOA can enable the richest possible outcomes on the network
 - And it's probably the shortest route to get there.
 - When in doubt, ask “What would the Web do?”
- Transition of tools, techniques, and architectural styles is required by most practitioners.

Wrap-up: Questions?

For Slides:
dion@hinchcliffeandco.com



<http://e2tvshow.com>