JOURNAL OF OBJECT TECHNOLOGY

Online at www.jot.fm Published by ETH Zurich, Chair of Software Engineering ©**JOT, 2007**

Vol. 6, No. 3, March - April 2007

Oh, What a Tangled Web We Weave!

Mahesh H. Dodani, IBM Software, U.S.A.

1 ENTERPRISE 2.0?

"Situational applications are a way for people with domain expertise to create applications in a very short time. Many IT shops have a backlog of small little projects that their customers want. If it takes 3 weeks to get to a project, the need is gone before the developer even starts coding. We want to give knowledge workers the tools to solve their own problems." – http://devzone.zend.com/node/view/id/678

As I have discussed in several earlier articles focusing on SOA, the key reason for its popularity is as an enabler for enterprises that need to facilitate business innovation by aligning to a flexible and agile IT. As I reported on the SOA state-of-the-art (http://www.jot.fm/issues/issue_2006_11/column5), SOA has crossed the chasm and become the mainstream IT approach for enterprises.

However, change happens! As enterprises have evolved and tapped into flexible and innovative business models as well as their business ecosystems, they are facing the following issues:

- The ability of enterprises to create partnerships quickly to extend their business ecosystem has the side effect of requiring extensive integration work. Each of these integration work requests usually requires many months to complete using the current technologies and supporting products & tools.
- However, many of these business collaborations are short lived, and typically last for a few months. Furthermore, the applications that are needed to support these collaborations are needed just-in-time, have large variability in their requirements for capabilities and data, and can usually be disposed off after the collaborations are completed.
- External business data and applications that could significantly impact results is constantly changing, and need to be integrated into the applications based on the needs of the business collaboration or situation being addressed.

During the same time frame as the SOA evolution, Web 2.0 has been evolving at a very fast pace to leverage the technologies that radically simplify the access to applications and data combined with the ability to harness the collective intelligence of social networks to build rich internet applications. Web 2.0 is best described in

Cite this column as follows: Mahesh Dodani: "Oh, What a Tangled Web We Weave!", in *Journal of Object Technology*, vol. 6. no. 3, March - April 2007, pp. 33 - 38 http://www.jot.fm/issues/issue_2007_03/column3

http://www.oreillynet.com/pub/a/oreilly/tim/news/2005/09/30/what-is-web-20.html?page=1 and is based on the following three broad patterns as shown in Figure 1:

- The first key pattern is software as a service. It enables end users to adopt capabilities directly, and gain immediate benefits from it. Many Web 2.0 offerings have a very low cost of entry – another distinguishing characteristic that levels the playing field for small businesses. And of course they use the public infrastructure, so anyone in the world can walk up and use it with a web browser.
- The second key pattern is the use of community mechanisms. Most Web 2.0 businesses use mechanisms to enable users to play a part in the service, adding value as they use it. For example, eBay uses ratings to measure the reputations of vendors, which leads to more trustworthy transactions. Others include all kinds of social networking features that help people find each other. Tagging further helps people filter information for relevance. There are diverse kinds of community mechanisms – only a few are listed in Figure 1.
- The third pattern is the simplicity of the user experience and the various interfaces by which developers can access data and capabilities. For end users, have seen significant improvements in user interface design and responsiveness based on AJAX (Asynchronous JavaScript and XML) methods. For developers, many simple and highly scalable mechanisms have emerged, such as feeds, simple extension mechanisms, and well-behaved HTTP-based APIs. Developers are building all kinds of situational applications on top of services, remixing them in various ways, without ever having to contact the service providers. The result is a rich, decentralized Web 2.0 business ecosystem.

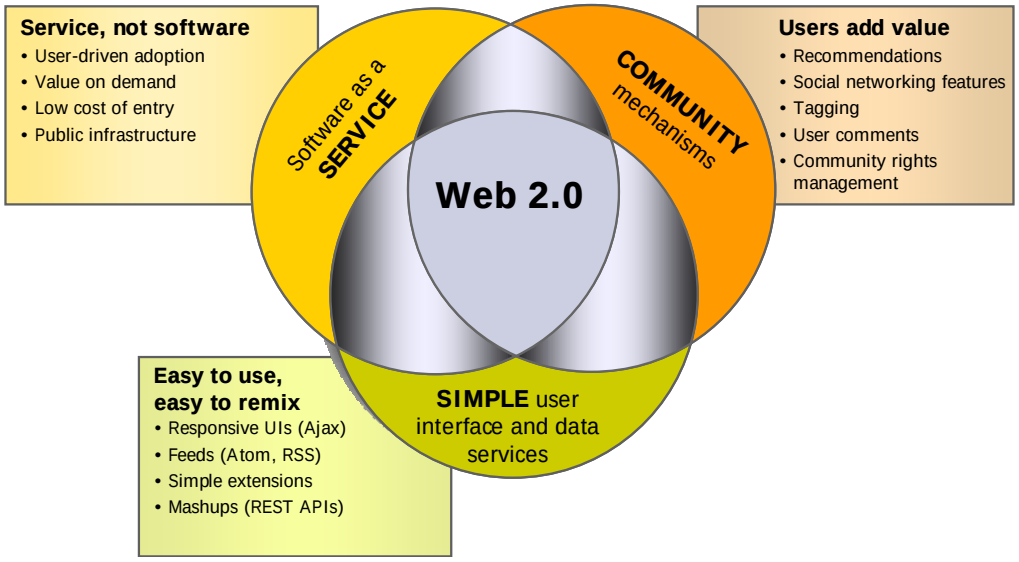
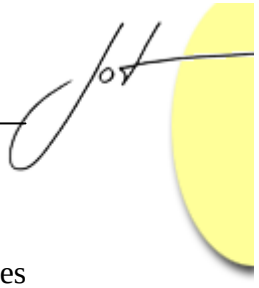


Figure 1: Web 2.0 Principle Patterns



There are several wonderful examples of companies that have used Web 2.0 technologies to provide rich internet applications, including Google Maps (http://www.google.com/maps), Flickr photo sharing (http://www.flickr.com/), and YouOS web operating system (https://www.youos.com/.) Several new businesses have been built purely by integrating content and data using Web 2.0 technologies, including Zillow (http://www.zillow.com/) for real estate sales, HousingMaps for rentals (http://www.housingmaps.com/), and literally thousands of so-called mashups (http://www.programmableweb.com/mashups.) Mashups are websites or Web 2.0 applications that use content from more than one source to create a completely new service. Content used in mashups is typically sourced from a third party via a public interface or API.

As you would expect these mashups and Web 2.0 technologies are converging with the SOA approaches to handle the situational issues faced by enterprises. Note that the Web 2.0 patterns shown in Figure 1 are also applicable to the enterprise. While retaining full control over a private infrastructure, enterprises can still take full advantage of community mechanisms and simple interfaces. Community mechanisms can help people connect within a large organization and across geographic boundaries, just as they help people connect across the public infrastructure. Simple interfaces promise to increase adoption and help spur situational applications within the enterprises.

Enterprise situational applications are built to solve an immediate, specific business problem. These applications are built by blending externally available services, applications and data with enterprise specific content and services. Situational applications are also information centric, focusing on manipulating static and increasingly dynamic content. Situational applications development are accelerated by community-based collaborations.

For the rest of this paper, we present enterprise level support for Web 2.0 technologies and capabilities and how they can be used to build situational applications.

2 ENTERPRISE WEB 2.0 TECHNOLOGIES

Let us start by outlining the main Web 2.0 technology attributes appropriate for the enterprise as shown in Figure 2:

- AJAX, shorthand for Asynchronous JavaScript and XML, to facilitate information and services to be mashed up into new interactive portals. AJAX is a web development technique for creating interactive web applications. The intent is to make web pages feel more responsive by exchanging small amounts of data with the server behind the scenes, so that the entire web page does not have to be reloaded each time the user requests a change. This is meant to increase the web pages' interactivity, speed, and usability. The Ajax technique uses a combination of technologies to achieve this usability including XHTML (or HTML) and CSS

for marking up and styling information, DOM accessed with a client-side scripting language (such as JavaScript and Jscript) to dynamically display and interact with the information presented, XMLHttpRequest object to exchange data asynchronously with the web server, and XML for transferring data between the server and client.

- A lightweight programming model that relies extensively on REST (Representational State Transfer) to transmit domain-specific XML data over HTTP. REST is an alternative to SOAP or other web services based programming models, and can work without an additional messaging layer or session tracking via HTTP cookies
- Changes are handled via RSS/ATOM feeds, which allow someone to link not just to a page, but to subscribe to it and get notified every time that page changes.
- Wikis are an effective tool for mass collaborative authoring of content. They allow users to easily add, remove, and otherwise edit and change available content. This ease of interaction and operation makes a wiki an effective tool to control unique, hard-to-recreate data sources that get richer as more people use them.

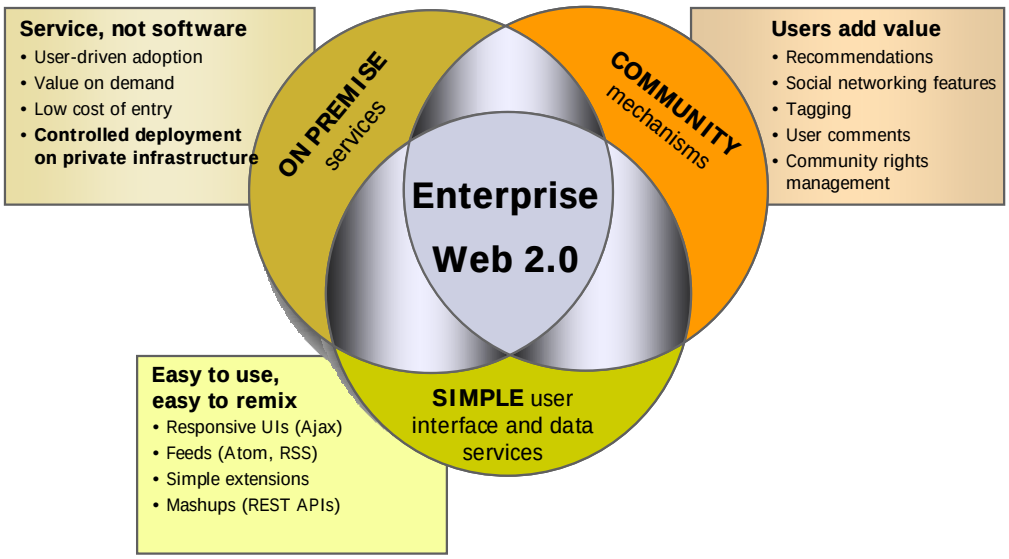
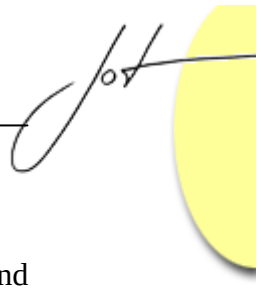


Figure 2: Enterprise Web 2.0 Patterns

For enterprises to take advantage of these technologies, they need to leverage the following capabilities:

- These applications must be able to manipulate content very effectively. The Web is all about content - HTML, forms, images, audio, etc. Enterprise application interfaces and data surface through Web pages and feeds. Mashups are an



additional, personal approach to integration that builds on content and complements services.

- Services, not packaged software, with cost-effective scalability is the primary manner in which content is made available and manipulated. Services must be usable on-demand, and with ease using the technologies highlighted above.
- The power of communities must be harnessed in all aspects of developing and maintaining the enterprise applications. Users must be treated and trusted as co-developers, in a reflection of open source development practices. The open source dictum, "release early and release often" should be adhered to; thus, allowing the applications to be in perpetual beta mode.

Note that enterprise strength products and tools must support all of the above capabilities.

One way to enable such situational enterprise applications is via an application wiki. Application Wikis are environments for collaborative, situational (ad-hoc) dynamic content development with the following capabilities:

- They facilitate web solutions to be developed by non-programmers who are domain experts - i.e. mash-ups, dashboards, etc.
- They use a mark-up based client development strategy, employing plug-in models for easy extensibility.
- Thy allow scripters to easily weave together a "good enough" solutions.
- Combined runtime and lightweight assembly capabilities.

IBM QEDWiki (Quick and Easy Develop Wiki) http://services.alphaworks.ibm.com/qedwiki/ is a browser-based tool used to create simple enterprise mash-ups. A mash-up maker is a development environment for enterprise mash-ups by assembling software components (or services) made available by content providers. QEDWiki is a unique framework that provides both Web users and developers with a single Web application framework for hosting and developing a broad range of Web 2.0 applications. QEDWiki can be used to develop a wide range of Web applications, including situational applications. QEDWiki also provides Web application developers with a flexible and extensible framework for rapid prototyping applications. Business users can quickly prototype and build ad-hoc situational applications. QEDWiki provides mash-up developers with a framework for building reusable, tag-basedwidgets or services. Business users who wish to create their own applications can use these services (or widgets.)

One of the examples that is showcased for QEDWiki is a retail store enterprise mash-up. Hardware store chains are affected by weather for example, rain means they need to make sure their stores have plenty of Retail stores gutter repair supplies, and snow means supplies of snow shovels and snow blowers. The showcased enterprise mash-up, combines a service pulling data from the corporate ERP system to display each stores inventory with a live feed weather service. As each store is highlighted, the mash-up displays the weather for that store along with the relevant inventory. Managers can quickly decide to transfer inventory between stores or place orders with vendors. By

exposing these feeds as web services to their vendors, the enterprise mash-up can be extended to help them predict their needs as well. Services created for one need begin to feed other needs as well. You can watch the video that shows the QEDWiki in action at http://www.youtube.com/watch?v=ckGfhlZW0BY.

3 WHEN FIRST WE PRACTICE TO DECEIVE

Let me end this article on a cautionary note. The astute reader would have noticed that the entire article focused primarily on the technology side of the equation of the Enterprise 2.0. However, if it is to succeed, we must focus on the more difficult sociology/culture side of the equation. How do we change the mindset of the enterprise workers and users to be able to develop and use these situational applications as part of their day-to-day work? Our experience to date is that it is very difficult and time-consuming to get such far-reaching cultural changes within an enterprise implemented in a controlled fashion. However, as people use the same applications outside of their work environment and get comfortable with the approach, we may be able to accelerate the adoption within the enterprise. The key to this success is the realization that these situational applications are constrained to handling a work effort just-in-time, and that they will most likely be disposed after the work has completed. It may therefore be easier to motivate the users to be empowered in defining the best way to support the work they are doing, and take advantage of the short delta in skills needed to enable them. These are exciting times for everyone in business – the ability to actively participate in developing the right tools for the job!

About the author

Mahesh Dodani is a software architect at IBM. His primary interests are in enabling communities of practitioners to design and build complex business solutions. He can be reached at dodani@us.ibm.com.