



July 26, 2007

# Developing Enterprise Web 2.0 Applications

by Jeffrey Hammond

for Application Development & Program Management Professionals



July 26, 2007

## Developing Enterprise Web 2.0 Applications

New Business Opportunities Reward Flexibility And Adjustment

by **Jeffrey Hammond**

with John R. Rymer, Randy Heffner, Carey Schwaber, Erica Driver, and Jacqueline Stone

### EXECUTIVE SUMMARY

The hype about Web 2.0 often obscures the practical benefits of this set of technologies, applications, and concepts. Development teams have begun using Web 2.0 user interface technologies, service-oriented architecture (SOA), and Social Computing features to create sometimes startling consumer applications. But few have yet applied these ideas to enterprise applications in more than trivial ways. When they do, application development professionals will find that Web 2.0 describes the transition of the Internet from flat, static, standalone Web sites into a participative, adaptive, consumer-centric medium. Snappy Web 2.0 user interfaces are the starting point, but capitalizing on Web 2.0 will require Agile development processes and a sharp focus on service creation, application assembly, and community-based development that most IT shops do not employ today. The path to Web 2.0 for enterprise apps won't always be clear, but the technology offers big rewards, including committed customers, more productive employees, and empowered communities that increase the rate of innovation around corporate assets like products and historical data.

### TABLE OF CONTENTS

- 2 **Application Development Professionals Must Pay Attention To Web 2.0**
- 6 **Web 2.0 Development Teams Adjust Technologies And Processes**
- 9 **A Development Manager's Checklist For Web 2.0**

#### RECOMMENDATIONS

- 12 **Brace For Sustained Disruption As Power Shifts To Communities**

#### WHAT IT MEANS

- 12 **Web 2.0 Creates Disruption For All And Opportunity For Some**

### NOTES & RESOURCES

Forrester interviewed six vendor and user companies: Adobe Systems, Amazon.com, Mashery, Microsoft, Optaros, and ThoughtWorks.

#### Related Research Documents

["Topic Overview: Web 2.0"](#)

April 10, 2007

["Rich Internet Applications: Why And How"](#)

September 1, 2006

["Social Computing"](#)

February 13, 2006

["A Taxonomy Of Service Types For SOA"](#)

October 25, 2005

## APPLICATION DEVELOPMENT PROFESSIONALS MUST PAY ATTENTION TO WEB 2.0

In the three years since O'Reilly Media introduced the term *Web 2.0*, application development professionals have seen a surge in irrational exuberance reminiscent of the headiest days of the Internet bubble. While it's easy to dismiss Web 2.0 as nothing more than a fancy term for the next round of Web development technologies, this attitude risks ignoring the tangible benefits that business and development teams are realizing when they successfully apply the set of technologies and applications that Web 2.0 has come to represent.<sup>1</sup>

### Web 2.0 Creates Business Opportunities

Businesses leaders who adopt Web 2.0 technologies and applications aren't just looking for a technology update from development teams; they are also looking for application development professionals to help them drive revenues, improve employee productivity, and take out costs. Web 2.0 does so in three big ways:

- 1. Social Computing creates empowered communities.** Active customer communities increasingly define the value of Web sites and applications. As a result, application development professionals create value when they involve customers directly in the development project. Adobe Systems is doing exactly that with its new "kuler" service.<sup>2</sup> The payback for Adobe? Immediate feedback from the professional design community, increased word-of-mouth marketing, and user-added content all make kuler more valuable to customers. As the kuler community grows, Adobe is incorporating the kuler services into traditional packaged software apps, making them more attractive to the community in the process. Empowering a community isn't risk-free though; self-supporting customers reduce brand control, as they freely communicate their negative experiences about products and services as easily as they share their positive ones.<sup>3</sup>
- 2. Access to content and data spawns innovation networks.** Web 2.0 shops turn the idea of information control on its head by giving external developers access to valuable content and data. Microsoft's Windows Live site, for example, offers developers up to 25,000 search queries a day and access to 100,000 transactions per day on Virtual Earth.<sup>4</sup> Microsoft's strategy is to create an innovation network where firms use its content and data to build mutually profitable businesses. And Microsoft is not alone: Amazon.com's Web services give developers programmatic access to Web site traffic, historical pricing, and complex search data.
- 3. Effective engagement becomes a competitive differentiator.** Consumers are increasingly aware of and strongly prefer Web applications that are easy to use, responsive, and combine rich media with improved visualization techniques.<sup>5</sup> And it's not just consumers: When Symbol Technologies introduced a new rich Internet interface as part of its RF management console, the customer response wasn't just positive — it resulted in a significant increase in customer satisfaction. Improved usability and reduced deployment costs have translated directly to increased customer loyalty and generally higher expectations of what capabilities all RF management consoles should provide.

When development execs adopt Web 2.0 technologies and processes, they allow business leaders to test and refine new business models, get immediate community feedback, and seize unexpected opportunities as they emerge. For example, Mashery uses Amazon's EC2 and S3 services to virtualize its operations' infrastructure. The result? Mashery has lower operational costs than a typical startup but can still scale to meet increased demand without having to order and install new hardware and storage.<sup>6</sup>

### Deconstructing Web 2.0 Applications

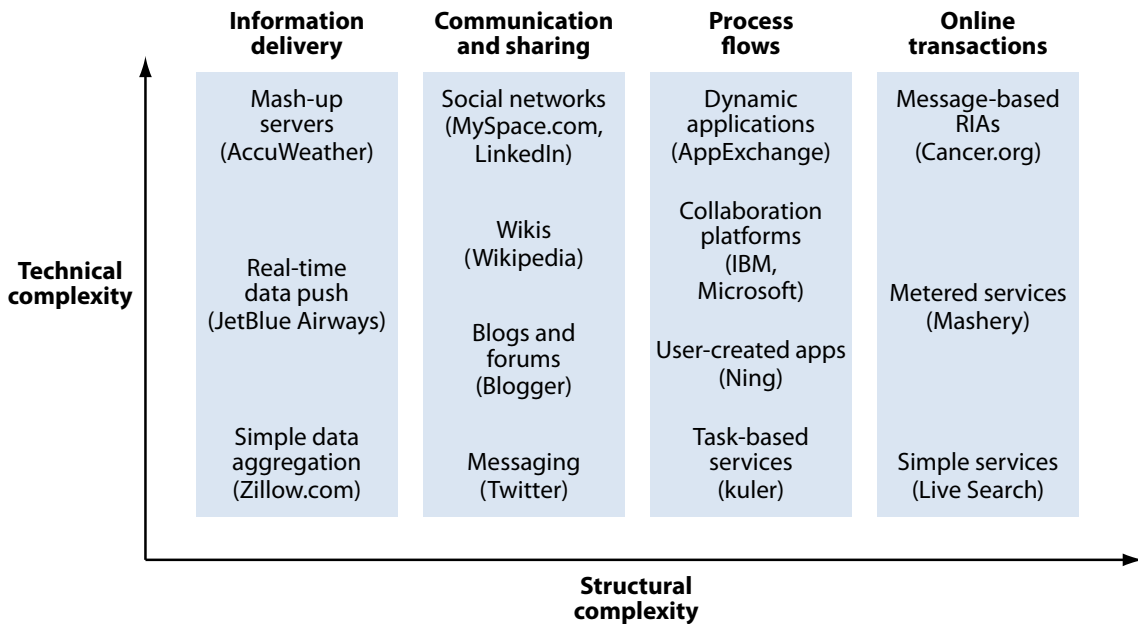
What do Web 2.0 applications look like? The most widely cited examples are so-called "mash-ups" created using services provided by Amazon, eBay, Google, Microsoft, and others.<sup>7</sup> Developers combine these services with other services using scripting languages to create applications that display, for example, delivery routes on online maps. But mash-ups aren't the only Web 2.0 application type (see Figure 1):

- **Mash-ups focus on information delivery.** Most of today's mash-ups focus on information delivery.<sup>8</sup> As a result, many mash-ups are not especially large and can be built by small teams or single developers. Mash-ups like Zillow.com frequently aggregate static information, are stateless, and don't push updates back to aggregation sources. These qualities make them good candidates for early experimentation. But not all mash-ups are simple: JetBlue Airways uses mash-ups to push real-time flight data to passengers, and enterprise mash-up platforms from IBM, Kapow Technologies, and OpenSpan allow development teams to integrate existing apps into complex, bidirectional mash-ups without completely rewriting them.<sup>9</sup> AccuWeather is testing exactly this sort of complex mash-up using IBM's QEDWiki mash-up technologies, including QEDWiki and Mashup Hub, to build a B2B application serving real-time weather data.<sup>10</sup>
- **Social Computing sites enable communication and sharing.** Social networking sites like Blogger, Digg, LinkedIn, MySpace.com, Twitter, and Wikipedia enable self-organizing groups to find each other and share information. The value in these sites is not the wikis, blogs, and tagging technologies that they use but rather the fact that they enable end users to create, edit, and classify content and allow people to connect with others. Over time, user-generated content and semantic links lock users in to a site more tightly than proprietary technology platforms can. Although another site could duplicate LinkedIn's technology and functionality without too much difficulty, prying dedicated LinkedIn users away from their existing contact networks is a far more challenging proposition.
- **Collaboration platforms support process flows.** Social networking sites allow people to connect and share, but collaboration platforms take communication and sharing a step further by adding services like messaging, team workspaces, ad hoc workflow, real-time messaging, presence awareness, and conferencing.<sup>11</sup> Vendors like Microsoft and IBM expose these capabilities and others (e.g., content, portal, and office productivity) as services that application development professionals integrate into Information Workplaces.<sup>12</sup> The result? Traditional

development distinctions, like package integration versus custom development, dissolve as platforms like Microsoft and SAP's Duet and salesforce.com's AppExchange offer more granular integration options to enterprise Web 2.0 development teams.<sup>13</sup>

- **Rich Internet application (RIA) frameworks move toward transaction flows.** While transactional standards for Web services are still maturing, development managers are turning to proprietary RIA frameworks as a stopgap measure.<sup>14</sup> Ajax frameworks like TIBCO Software's General Interface and Nexaweb Technologies' Internet Messaging Bus support reliable message passing over the Internet, and applications like the American Cancer Society's Clinician Portal use Adobe's Flex to support secure data access, conversational application flows, and complex data entry.<sup>15</sup> It would be a stretch to say that RIAs are ready for high transaction loads or long-running business processes, but development teams are increasingly using RIA frameworks as replacement technology for client-server and Web apps because they combine the benefits of rich client user experience with Web application ubiquity.

**Figure 1** Types Of Enterprise Web 2.0 Apps



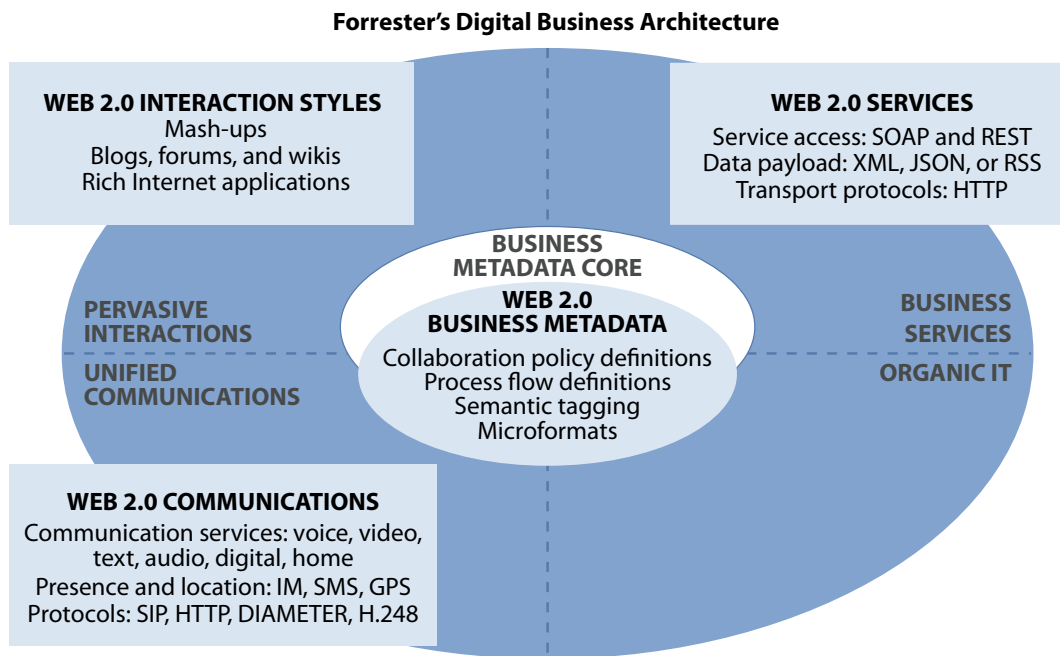
## Elements Of Web 2.0 Architecture

When it comes to building applications and choosing technologies, it's easy to focus on the glitzier aspects of Web 2.0. Grafting on an Ajax tool kit or integrating Google Maps may improve an application's look and feel, but that's just part of a more comprehensive enterprise Web 2.0 architecture. To begin with, Web 2.0 applications will be only part of the overall application portfolio, and Web 2.0 technologies are merely a subset of the technologies from which all applications will be built. By using a holistic model, such as Forrester's Digital Business Architecture, architects can fit new Web 2.0 applications and technologies into an existing set of architectural domains and use much of the same infrastructure as other enterprise IT applications (see Figure 2).<sup>16</sup> As part of this exercise, it's important to understand the implementation patterns that drive Web 2.0. This architecture:

- **Provides extensible services.** Services are key building blocks from which Web 2.0 applications are built. When application development professionals create services that provide access to a firm's content, data, and process, they stimulate innovation outside the firewall. When Amazon provided a service interface to its recommendation engine, a company called liveplasma.com used it to drive a visual recommendation engine to engage users looking to discover new music suited to their tastes. Of course, liveplasma.com provides a direct link back to Amazon to order new music, and both sites profit from an enhanced user experience. It's important to note that multiple alternatives exist for creating and aggregating services, including SOAP and Representational State Transfer (REST).<sup>17</sup> Data formats also vary: JavaScript Object Notation (JSON), Really Simple Syndication (RSS), and plain old XML (POX) are all used to deliver resource payloads with RESTful services.
- **Enables dynamic application assembly.** Web 1.0 applications were independent destinations, and users browsed from page to page and site to site to obtain the data and content they wanted. Web 2.0 applications are increasingly dynamic, and developers assemble them to adapt to a specific user's needs. When Hurricane Katrina struck the US Gulf Coast in August 2005, the storm displaced tens of thousands of people and severely tested the command and control capabilities of relief organizations. But individuals and organizations stepped into the gap with dynamic apps that allowed victims to help themselves by entering and searching for information about housing and relatives.<sup>18</sup>
- **Supports pervasive interactions.** While the desktop and laptop were at the center of Web 1.0, the convergence of next-generation IP networks, rich media, increased bandwidth, and handheld capability has resulted in an explosion of Web-connected devices with a variety of different form factors and performance characteristics. As a result, it is increasingly important that application development professionals allow users to select how and to which device they want content delivered. Podbop takes this approach when it gives listeners the choice of where they get the latest feeds from bands that are playing in their area.<sup>19</sup>

It's important for enterprise architects to plan and design for Web 2.0 within the broad landscape of a firm's technology architecture. Why? Forrester sees an additional class of applications emerging called dynamic business applications — software systems that embody business processes; are built for change; and deliver rich, context-aware information to individual users. Dynamic business apps and Web 2.0 applications share many of the same characteristics and requirements, although there will likely be variations in the patterns by which these two classes of applications are built and delivered. It's critically important for architects to use a holistic model like Digital Business Architecture as the backdrop for planning and designing Web 2.0 applications, technology, and architecture so that the resulting applications interact with, extend, and enhance dynamic applications.

**Figure 2** Web 2.0 Patterns Within Digital Business Architecture



42868

Source: Forrester Research, Inc.

**WEB 2.0 DEVELOPMENT TEAMS ADJUST TECHNOLOGIES AND PROCESSES**

Development managers will find that building enterprise Web 2.0 applications will require adjustments to traditional development technologies and processes. As consumer Web 2.0 apps proliferate, a richer set of development technologies and assembly techniques is gaining strength outside corporate IT firewalls — and it's not just for small applications. Why is this important? As employees grow more accustomed to rich interfaces and improved usability on B2C sites, they will increasingly expect a similar experience from B2B applications and will grow frustrated if development execs can't deliver.

## Application Assemblers Adjust To New Development Technologies

Development managers leading Web 2.0 development teams have increasingly turned toward new application development technologies and tools to assemble Web 2.0 apps. Why? Commonly used programming languages like Java, C#, and C++ are powerful but complex and tend to favor tightly coupled model-view-controller (MVC) architectures, while enterprise development tools take time to acquire, install, and configure. New frameworks, programming languages, and assembly tools make it easier for assemblers to create dynamic applications on top of existing IT investments in J2EE and .NET platforms. In the Web 2.0 technology world:

- **RIA frameworks are table stakes.** Web 2.0 development teams were early users of RIA development frameworks; as a result, a simple set of DHTML or Ajax controls looks just plain boring in comparison to new applications like eBay's San Dimas Client or the *New York Times'* Times Reader.<sup>20</sup> New RIA platforms from Adobe, Google, Microsoft, and Yahoo! are pushing Web 2.0 user interfaces beyond the browser and adding capabilities like offline storage and synchronization.<sup>21</sup> Player-based RIA platforms also deliver rich media content that is at the core of many consumer-focused services like YouTube, MTV Networks, and Fandango.
- **Dynamic scripting languages speed application assembly.** Dynamic scripting languages like Perl, PHP, Python, and Ruby are finding a home at the assembly layer of Web 2.0 applications.<sup>22</sup> ThoughtWorks frequently uses Ruby on Rails because it allows the firm to assemble Web applications faster than it could with traditional languages like Java or C#. Whether it's the built-in support for Web services in the core PHP libraries or the flexibility of Rails, Web 2.0 assemblers are taking full advantage to speed their time-to-market. And larger ISVs have noticed: Both Microsoft and Sun Microsystems are taking rapid steps to integrate dynamic languages into their core virtual machines.<sup>23</sup>
- **Dynamic applications use a variety of architectural patterns.** Most architects are familiar with the MVC pattern, but effective dynamic applications also draw on other architectural patterns like Pipes and Filters.<sup>24</sup> Looser coupling between user interfaces, activity flows, and core services allows extended communities to interact with core content and services with greater flexibility. As a response to this demand, Microsoft provides both user interface (UI) components and headless access for its services on dev.live.com. Another trend is toward the use of simpler REST-style service architectures over SOAP-based Web services because REST services are simpler to create and provide the broadest reach across the widest array of possible clients.
- **New tools increase development and assembly options.** Smaller specialists are moving quickly to meet the needs of Web 2.0 development teams. Examples of new entrants include Ajax providers like Backbase and JackBe. Traditional players in enterprise systems like TIBCO and Software AG are also using Web 2.0 to re-enter the development market with tools that focus on dynamic application development. Another trend is toward browser-based applications assembly environments like those from Bungee Labs, Microsoft Popfly, and Ning.<sup>25</sup>

- **Open source frameworks accelerate delivery cycles.** Web 2.0 development organizations tend to focus on keeping speed up and costs low. Open source frameworks and tools satisfy both needs. Dave Gynn, director of tools and frameworks at Optaros, put it this way: “In the time it takes to complete a normal middleware purchasing cycle, we’ve often completed our first release and are well into our second iteration. I’m not sure how we could achieve the time-to-market our clients demand without using open source frameworks as the basis of our assembly efforts.”

### Development Teams Use Agile Processes And Become More Externally Focused

Application development professionals can adopt any or all of the Web 2.0 technologies listed above and see benefits from their use. But adjusting development technologies alone may not be enough to ensure success. Once a community begins to form around a Web 2.0 application, app dev professionals will find that it becomes important to serve that community and protect it from potential competitors. To do this, development teams need to make applications and services available to the community early and often. Development managers at Web 2.0 shops cite several adjustments to traditional development practices that improve their success in attracting vibrant, extended communities:

- **Agile processes enable “perpetual beta.”** The traditional phased define-design-build-test-debug-deploy-development process does not adapt well to the Web 2.0 world of rapid service creation and dynamic application assembly. Agile development processes like Extreme Programming and Scrum are critical to enabling the pace of delivery that Web 2.0 shops need to stay ahead of their competition.<sup>26</sup> Optaros, a systems integrator specializing in Web 2.0 development, uses Agile because it helps its project team stay focused on rapid delivery. An initial release at Optaros may consist of the work done over several prior month-long sprints; after that point, new functionality is deployed at the end of every subsequent sprint. Some development shops use Agile to work in a “perpetual beta,” where services are upgraded at the end of every iteration instead of after several iterations that comprise a larger release cycle.<sup>27</sup>
- **Community feedback loops drive development.** Creating a continuous feedback loop is critical to building a community around a Web 2.0 application. If community members feel like no one is listening to their needs, they will drift away to other areas of interest. Adobe has adapted a traditional product management role by adding responsibilities for managing Adobe-sponsored forums and monitoring external blogs and wikis to capture community feedback. Enhancement requests become part of the issue list that development teams use to prioritize new features for the next iteration. It’s also important to note that the feedback doesn’t always need to be positive. Knowing that a feature request won’t be supported — and why — is sometimes just as effective in creating a loyal community, and it’s certainly better than no feedback at all.
- **Applications ship with service APIs.** Delivering a cool new application to a user community gets a company noticed, but delivering headless services to access the application’s data at the same time gets the application integrated into the fabric of the Web. Flickr’s photo cataloging application certainly gets its share of standalone users, but the companion API makes Flickr’s

photos, tags, profiles, and groups available to any developer who wants to build Flickr's services into a larger application. As a result, Flickr content is incorporated into hundreds of other Web applications, a network effect kicks in, and an innovation network sends the resulting community into overdrive.<sup>28</sup> The lesson for architects and development managers? Make sure that development projects begin with extensibility as a primary design goal, and ensure that inadequate APIs aren't tacked on at the end of a project as an afterthought.

- **User interface design is without a doubt a top priority.** Effective user interface design plays an important role in creating an emotional connection with an application. And the good looks and interactive nature of Web 2.0 technologies raise the bar for delightful — not just functional — user experiences.<sup>29</sup> Development execs can improve user interface design by adding customer experience professionals to development teams and adopting design best practices like persona development and Scenario Design.<sup>30</sup> Creating a connection to users will become increasingly important as a growing number of Web 2.0 sites, gadgets, and applications compete for the same limited space on a user's desktop or home page.

It's possible to do Web 2.0 development without adopting these practices, but application development professionals will find that the more responsive their development teams are to the extended community that they support, the higher the likelihood will be that they will create a network effect around their project.

## A DEVELOPMENT MANAGER'S CHECKLIST FOR WEB 2.0

Once application development professionals decide to apply Web 2.0 technologies and concepts to enterprise applications, the next question is how to get started. Development executives can start by selecting existing applications that could benefit from user interface improvements or by identifying opportunities to create an extended community of developers around existing applications. Only after the right business case is in place should development managers begin to apply Web 2.0 development technologies.

### Five Steps Forward With Web 2.0 Development

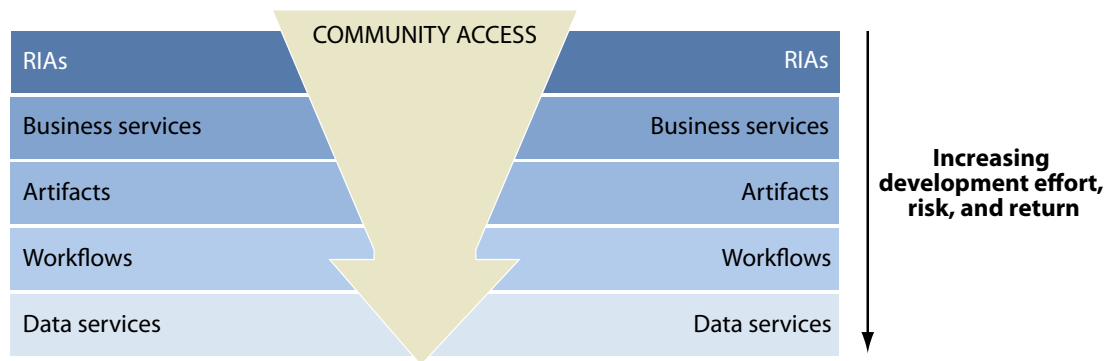
Here's a list of five individual tactics that collectively move an enterprise development shop toward Web 2.0. Each tactic can be applied on its own or in combination with the other tactics on the list.

1. **Improve existing user experiences with RIAs.** As a first step toward Web 2.0, application development professionals can apply RIA development frameworks to existing applications without significant changes to application architectures (see Figure 3). Ajax frameworks are easily integrated into server-side Java and J2EE applications, and Microsoft's Silverlight works well on top of .NET applications. While internal users will see immediate benefits, improving the user experience of existing apps also makes these apps better candidates for exposure to external users.

2. **Expose content and data with appropriate service types.** While snappy graphics get an app noticed, access to content and data through service APIs attracts power users to a community. Query and content business services enable mash-ups, and transactional business services further external integration efforts.<sup>31</sup> This step is more challenging and a bit riskier than simply improving the user experience because it represents the first real transfer of control to the community. Enterprise architects must also define whether APIs will support REST or SOAP (or both) and how the data formats will be specified.
3. **Expose artifacts to external manipulation.** The next step toward Web 2.0 is to give a community the power to create, edit, and manipulate site content and artifacts. This could be as simple as adding a self-help discussion forum or wiki to an application, or it could be something more substantial like user interfaces for creating, organizing, and sharing documents and images, à la Flickr. This step creates an inflection point in Web 2.0 adoption because the applications and services that development teams expose now begin to take a back seat to the value of the content that the community adds to the site, and the community will begin to assume control. This is also the point at which a network effect is most likely to kick in.
4. **Add access to workflow and process.** Adding context to services and artifacts is the next logical step in creating higher-value applications. Development teams should prioritize community feedback as they make decisions about what sorts of task-based functional application services to add to application APIs and how much workflow capability to expose to assemblers. If users can easily accomplish the tasks that are important to them, then they'll become more and more attached to the site. Supporting ad hoc assembly by end users is a more challenging approach than straight custom development because it requires development teams to expose richer collaboration capabilities like messaging and document sharing, as well as assembly tooling that is easy enough for power users to use unassisted.
5. **Directly expose data that creates value.** Exposing application data services makes sense if the data is valuable to others in its own right. Analytical business services that provide information about customer purchase preferences and product inventory levels may be valuable to an extended network of suppliers or resellers, and access to it can become a competitive differentiator. The downside? Exposing data directly to external users requires additional technical safeguards like access keys and usage metering. Nontechnical questions like how to monetize valuable data and what service-level agreements are appropriate also need to be answered before development execs push their enterprise Web 2.0 strategy to this level.

Application development professionals who approach enterprise Web 2.0 development in a stepwise fashion will benefit because they will minimize effort and risk early, while they gain experience in working with an extended community of external developers and content creators. Another benefit is that implementing process changes is less critical during the first two steps when the balance of control is still in favor of the development organization. This allows development teams to get started with Web 2.0 technologies, while development managers gain confidence and document benefits before they fully commit the organization to the process adjustments required to successfully develop an empowered community.

**Figure 3** A Stepwise Approach To Web 2.0 Development



41868

Source: Forrester Research, Inc.

### Where Web 2.0 Development Doesn't Make Sense

Just as IT shops did not rewrite all host-based apps for client server and not all client server apps migrated to the Web, there are limits to what should be attempted with Web 2.0 technologies:

- **Web 2.0 is not ready for the most complex applications.** Applications that require complex event processing or low-latency messaging are still beyond the Web 2.0 technology stack. Standards that support rollback, guaranteed delivery, and multiphase commit are still evolving, so if an application has these types of requirements, it's best to stay away from Web 2.0.
- **Exposing services is an all-or-nothing proposition.** Hidden APIs can be found and quickly used in unintended ways. Organizations that are not prepared for usage spikes can quickly find issues with hardware and provisioning as requests for services go through the roof. Vendors like Mashery are moving to provide assistance, but for now, development managers should plan for worst-case scenarios, and architects should design with the expectation that the masses will find and use any exposed service.
- **It's hard to hide weak processes and apps.** Directly engaging customers exposes weak business processes and poorly designed applications. While internal users have no choice when it comes to using poorly designed applications, customers do. If self-service options are poorly designed, customers will look to see if a competitor offers more control and a better user experience.

## RECOMMENDATIONS

### BRACE FOR SUSTAINED DISRUPTION AS POWER SHIFTS TO COMMUNITIES

The evolutionary path that Web 2.0 will take in the enterprise is not yet certain, but the broad outlines are becoming clear. To adapt, application development pros should follow these recommendations:

- **Assess how extended communities could benefit your business model.** What opportunities do you have to develop an extended community for your business and software assets? Include community outreach as part of your development strategy. Use the results of this assessment to start a discussion with your line-of-business peers. Focus on how to create a development ecosystem around your applications and services. Good documentation and examples become critical to building community knowledge and experience.
- **Update your processes to support service creation and application assembly.** Plan to adjust your development processes to drive service creation and application assembly. Focus on improving agility and creating a continuous feedback loop with an extended, empowered community. Create a gap analysis of your existing development processes and technologies to drive strategic planning, developer training, and tool acquisition.
- **Selectively seed development with RIA, dynamic language, and open source expertise.** Begin to recruit and train architects and developers in rich Internet application and dynamic language concepts. If you don't have a strategy that takes advantage of external services and open source frameworks, then it may be hard to compete with nimbler competitors. Organize smaller teams at the service creation level to gain early expertise and track evolving standards.
- **Don't ignore business terms and licensing models.** How will you expose the services that your company creates around your own corporate assets? Are there usage limits to the services that your company may be using from third parties? Who owns the resulting intellectual property from the mash-up that your developers are creating? It's important for development execs to deal with the business and legal aspects of Web 2.0 in parallel with the technologies that drive it.

## WHAT IT MEANS

### WEB 2.0 CREATES DISRUPTION FOR ALL AND OPPORTUNITY FOR SOME

Change creates opportunity, and the shift toward a Web-centric development model will bring sustained and disruptive business and technical change to the way in which development teams practice their craft. Application development professionals who push their development organizations to adopt a community focus and provide easy-to-assemble services will attract vibrant communities, foster innovation, and create compelling business value. Their firms will be winners in the enterprise Web 2.0 race.

## ENDNOTES

- <sup>1</sup> For more information about what technologies and applications Forrester includes in Web 2.0, see the April 10, 2007, “[Topic Overview: Web 2.0](#)” report.
- <sup>2</sup> The kuler service lets designers create, collaborate, and swap color palettes and import them into other design applications like Adobe’s Creative Studio 3.
- <sup>3</sup> For a detailed treatment of Social Computing’s benefits and risks, see the February 13, 2006, “[Social Computing](#)” report.
- <sup>4</sup> These are the free query limits with commercial terms — higher limits are available for a fee. For more information, see the Windows Live terms of service at <http://dev.live.com/terms/>.
- <sup>5</sup> For more information about how Web 2.0 shops are improving user engagement and what types of user interface design principles users respond to, see the September 1, 2006, “[Rich Internet Applications: Why And How](#)” report.
- <sup>6</sup> Amazon’s Elastic Compute Cloud (EC2) and Simple Storage Service (S3) allow companies to provision servers and store data “in the cloud.”
- <sup>7</sup> A “mash-up” is a Web site or application that uses content from more than one source to create a completely new service. For examples, see <http://www.mashups.com/software.htm>.
- <sup>8</sup> For more information about different types of dynamic applications, see the July 20, 2005, “[How Composite Apps Will Change Enterprise Application Development](#)” report.
- <sup>9</sup> Customers onboard JetBlue flights can track their progress using Google Maps on their personal view screen by turning to channel 13. Web users can track in-air flights by navigating to <http://www.jetblue.com> and selecting the “Track a flight” option.
- <sup>10</sup> For more information about AccuWeather’s efforts to deliver real-time data to customers using mash-ups, see “AccuWeather to Deliver Timely, Localized Weather Information via IBM Emerging Technology,” IBM press release, December 7, 2006 (<http://www-03.ibm.com/press/us/en/pressrelease/20731.wss>).
- <sup>11</sup> For more information about how collaboration platforms play into Web 2.0, see the March 19, 2007, “[Topic Overview: Collaboration Platforms](#)” report.
- <sup>12</sup> Today’s information worker relies on a disjointed set of office productivity, content, collaboration, and portal tools. The Information Workplace (IW) will be much simpler yet richer than today’s tools by incorporating contextual, role-based information from business systems, applications, and processes; delivering voice, documents, rich media, process models, business intelligence, and real-time analytics; integrating just-in-time eLearning; and fostering collaboration. Using a service-oriented architecture, the IW will be rich with presence awareness, information rights, and personalization, and it will provide offline and online support to a plethora of devices. As this unfolds, information work will expand beyond traditional knowledge workers. See the June 1, 2005, “[The Information Workplace Will Redefine The World Of Work At Last](#)” report.

- <sup>13</sup> For detailed information on the convergence of Microsoft's collaboration strategy with Web 2.0, see the May 22, 2006, "Developers, Get Ready: 2007 Microsoft Office Is A Serious Application Platform" report.
- <sup>14</sup> For detailed information on the current state of Web services standards and how Forrester expects their adoption to proceed, see the December 14, 2006, "Your Strategy For Web Services Specifications" report.
- <sup>15</sup> For more information about the ACS Clinician Portal, see <http://labs.cancer.org/files/cpdemo.htm>.
- <sup>16</sup> For more information about Digital Business Architecture, see the November 7, 2005, "Digital Business Architecture: IT Foundation For Business Flexibility" report.
- <sup>17</sup> For more information about the benefits and drawbacks of different Web service architecture options, see the September 13, 2004, "SOAP Versus REST: A Comparison" report.
- <sup>18</sup> For examples of the some of dynamic applications organizations and individual developers created during the immediate aftermath of the storm, see *Google Maps Mania* at <http://googlemapsmania.blogspot.com/2005/09/summary-of-all-known-google-maps.html>.
- <sup>19</sup> Podbop allows users to input a ZIP code and then provides regularly updated feeds that feature music samples of bands that will be playing in that area. Users can choose to get the feeds in a browser or have them delivered to iTunes, where they can also be synchronized to an Apple iPod for later consumption.
- <sup>20</sup> For more information about eBay's San Dimas client, see the June 27, 2007, "eBay San Dimas Marks A New Era For RIAs" report.
- <sup>21</sup> For more information on RIAs, see the June 27, 2007, "Rich Internet Apps Move Beyond The Browser" report.
- <sup>22</sup> Dynamic languages do not supplant Java and .NET components in the service creation process but work at the assembly level, gluing together base information feeds and components. For more information, refer to the following source: John K. Ousterhout, "Scripting: Higher Level Programming for the 21st Century," *IEEE Computer*, March 1998 (<http://home.pacbell.net/ouster/scripting.html>).
- <sup>23</sup> Microsoft will release an update to the .NET Framework later this year that supports Python and Ruby through the Dynamic Language Runtime (DLR). Sun has announced support for Ruby in the Java Virtual Machine through the JRuby project.
- <sup>24</sup> The Model View Controller (MVC) pattern results in separate of the application presentation layer from the data and control layers. For more information on the Pipes and Filters pattern, refer to the following source: Frank Buschman, et al, *Pattern-Oriented Software Architecture, Volume 1: A System of Patterns*, John Wiley & Sons, 1996.
- <sup>25</sup> Ning, Bungee Labs, and Microsoft's Popfly are representative examples of in-browser dynamic application assembly environments; while browser-based IDEs have a checkered track record, assembly environments are targeting a different audience that is more likely to accept a browser-based usage model.
- <sup>26</sup> For more information on Agile adoption and how Agile processes and Agile practices are being adopted in enterprise IT shops, see the November 30, 2005, "Corporate IT Leads The Second Wave Of Agile Adoption" report, and see the June 11, 2007, "Enterprise Agile Adoption In 2006" presentation.

- <sup>27</sup> Perpetual beta refers to a cycle time that is so rapid that the traditional release cycle as we know it disappears. For a list of long-running Web 2.0 beta projects, see <http://www.37signals.com/svn/archives/000972.php>.
- <sup>28</sup> For an example of how Flickr releases its service APIs for developers and a sample of some of the resulting applications that developers have built, see <http://www.flickr.com/services/>.
- <sup>29</sup> The usability benefits that RIAs offer can also have a major impact on the experience that employees have with internal applications — and their productivity. And to encourage new users to become regular readers, blog owners should leverage existing Web design best practices that make content and functionality easy to find and consume and follow emerging blog guidelines that help users feel more comfortable participating in online conversations. See the March 1, 2007, “[Rich Internet Applications: Not Just For Customers](#)” report, and see the November 15, 2006, “[Best And Worst Of Blog Design, 2006](#)” report.
- <sup>30</sup> For a description of these best practices, see the November 9, 2006, “[Top 10 Ways To Improve Your Web Site User Experience](#)” report.
- <sup>31</sup> When deciding what types of services to expose to external development communities, it makes sense to have a taxonomy of which types of services can be exposed and which are the most valuable to the type of community that a business is attempting to create. For an appropriate taxonomy, see the October 25, 2005, “[A Taxonomy Of Service Types For SOA](#)” report.

# FORRESTER®

Making Leaders Successful Every Day

## Headquarters

Forrester Research, Inc.  
400 Technology Square  
Cambridge, MA 02139 USA  
Tel: +1 617.613.6000  
Fax: +1 617.613.5000  
Email: [forrester@forrester.com](mailto:forrester@forrester.com)  
Nasdaq symbol: FORR  
[www.forrester.com](http://www.forrester.com)

## Research and Sales Offices

Australia	Israel
Brazil	Japan
Canada	Korea
Denmark	The Netherlands
France	Switzerland
Germany	United Kingdom
Hong Kong	United States
India	

*For a complete list of worldwide locations,  
visit [www.forrester.com/about](http://www.forrester.com/about).*

For information on hard-copy or electronic reprints, please contact the Client Resource Center at +1 866.367.7378, +1 617.617.5730, or [resourcecenter@forrester.com](mailto:resourcecenter@forrester.com). We offer quantity discounts and special pricing for academic and nonprofit institutions.

Forrester Research, Inc. (Nasdaq: FORR) is an independent technology and market research company that provides pragmatic and forward-thinking advice to global leaders in business and technology. For more than 23 years, Forrester has been making leaders successful every day through its proprietary research, consulting, events, and peer-to-peer executive programs. For more information, visit [www.forrester.com](http://www.forrester.com).