Leveraging Enterprise Applications with Web 2.0

By James Hobart, President, Classic System Solutions

March 22, 2006

Delivering on the vision….

It all started a few years ago with the CIO vision… 'We need a 360 degree view of the customer'. If your company is typical of most out there, you have spent the last few years integrating disparate data sources to transform the executive vision into a reality for your internal users.

How are you doing? Have you delivered on the promise? Are your users more empowered, clients ecstatic and CIO beaming in the ambiance of increased sales and improved customer service? If that reality is still a few years away, no worries, you are not alone. Many companies have found that building the foundation for delivering web enterprise applications is more complex than planned and are just coming online with solid architectures to deliver on the original goal. The good news is Web 2.0 user interface design is poised to provide truly useful methods of delivering complex data in ways that will deliver on the vision.

What is Web 2.0?

We believe Web 2.0 is not a revolution as much as an evolution. Simply put, the web is finally leveraging standards (CSS, JavaScript, etc) to deliver engaging, interactive and integrated content in ways that allow users to focus on the task and the customer rather than the interface. That's right, Web 2.0 is an approach that uses standards-based technology (Ajax, Flash, Java) to deliver a better user experience, period. I know… you have heard this before... The slick sales team from XYZ Enterprise Solutions sold your management the last bill of goods on the 360 customer view implementation model using their 'Integrated XYZ on every desktop' promise. Unfortunately, the promise often was delivered with complex proprietary software based on UI technology decisions made 5-6 years ago. Worse yet, in order to integrate the enterprise solution, a majority of project effort was often spent on building web services to legacy data rather than focusing on custom tailored UI's specific for the key user tasks. Web 2.0 thinking gives us the opportunity to refocus our efforts on usability to design the tailored solutions users have been requesting all along.

Most Enterprise Software Fails in Usability

Simply put, enterprise software vendors have too many masters and a flawed business model when it comes to usability. Consider this; the basic business drivers influencing design decisions in these companies is often contrary to good usability. Want a tough job? Work as a usability practitioner in one of the enterprise software companies.

More features, not less.

Imagine yourself, the head of sales, going into the CEO of XYZ Enterprise Software and recommending elimination of 30% of the low usage features requested by various customers and cutting functionality by another 25% to help make this year's revenue numbers. This just will not happen. Instead, more features will get built, upgrade revenue will be collected, and the poor sap ;) trying to enter his time report into the recently upgraded self-service web-based time sheet system spends additional 15 minutes each week just trying to ensure he gets paid. Multiply this extra 15 minutes a week by the other 20,000 employees and the numbers become staggering.

'Zero Footprint' sells, thick client doesn't.

What about those customer representatives in the call center? Are they really more productive with your new web-based application? Sometimes, the answer is yes, but often, I hear whispers from the users that the 'Old' client-server system was faster and more efficient than the new web-based application. Here in-lies the problem. Client-server and windows applications just don't sell. CTO's want zero-footprint applications to lower their deployment costs. This is a serious issue as companies can spend upwards of a million dollars just to roll out software patches onto thousands of machines. Browser-based applications deliver extremely low deployment costs…but until now at a steep operational cost, due to their point-click-wait behavior. Don't believe me? Let's do a test. Remove all your favorite IDEs (Visual Studio, Eclipse, JBuilder, Dreamweaver) and try building your next major application in a web-based IDE. Change a line a code, press submit, wait, scroll back to that section of code, repeat. Oh, by the way, the 'Back' button doesn't work and don't plan on having any 'Undo' capabilities. My point is we don't eat our own dog food. Now go back and enjoy your favorite IDE.

Design for possibility, not probability.

New sales of enterprise software are dependent on smooth, quick implementations. Essentially the pitch at the executive level is often something like this: 'We've figured out the industry best practices so you need to adapt your business to be in line with how our software works'. When the realities of a true project come into play, software vendors focus on meeting the needs of the client and since each client is usually a bit different the entire design approach becomes based on possibility instead of probability. Essentially, the thinking is 'Let's make this flexible so the next client installation goes more smoothly'. While this can be a good thing, it often complicates the user interface resulting in lower user productivity.

How does Web 2.0 Help?

Designing a Web 2.0 interface does give us a unique opportunity to fix what is broken with most enterprise software today. Regardless of whether you are trying to implement the latest XYZ enterprise web solution or are building a custom web-based application, taking a Web 2.0 approach will likely make for a better user experience.

The Web 2.0 Approach

Since Web 2.0 is an approach, here are four characteristics I would focus on to improve the user experience.

1. **Less Implementation, more content**. Web 1.0 was about bringing data to the desktop, often via portals. This often resulted in a 'My XYZ' page with a hundred links. Great. Now the user has a 1 in 100 chance of making the right choice. These systems were flexible and certainly better than logging into 5 disparate systems, but how much better are they than just creating a static HTML page with those same links or a "Main Menu' in the old mainframe days? The web 2.0 approach will dispense with portlet views and provide integrated content tailored to how the user would actually use it. It will break the political boundaries within the company and provide rich contextual views so that users can make fluid decisions. Long gone will be the days of 'We cannot give you access to the data, but you can link to us…is that ok?' A great example of this approach is www.zillow.com where you can view real estate information from a customer view, not from a realtor view www.realtor.com.

2. **Rich interaction…where needed.** I like Rich, engaging interfaces when they work the way a human works. Unfortunately, many of them to date are loaded with 'surprises' and new learning experiences most users are not willing to tolerate. On the other hand, when an interface works like the real world, most people are happy. Take for instance www.kayak.com . This travel site adjusts the selection of flights when I move the sliders on the left. My stereo works the same way. Move the volume slider and I enjoy more sound. No learning, no waiting, no back button.

3. **Community collaboration**. We're social creatures. Hang out at a mall and watch people for a day. You'll learn a lot. When users look at your data, they often desire validation from others. Web 2.0 applications like www.flikr.com play on this natural human behavior to build decentralized social networks. That's correct; your new Web 2.0 application will actually embrace feedback from everyone in your company with a bottom up approach rather than a top-down approach. This is not a radical as it seems. Most key business knowledge is already stored at the bottom level in spreadsheets and emails far removed from the company's ERP and CRM systems. Just design your application to embrace this phenomenon.

4. **Purpose Built**. Visit a construction site and hang out with a carpenter. You will likely find a selection of saws for specific types of jobs. The web allows us to create a 1-to-1 relationship with the user. Know them, build just what they need and they will be happy. Avoid 'Boil the Ocean' designs that try to meet the needs of every user but rarely meet the needs of any. This approach requires clear vision and strong leadership. Products like www.gmail.com are forging the

path with simple but effective solutions. 'I need to check my email, I don't have much time and I don't want spam.' Done. Thanks Google.

What makes great software designs?

Over the years, I am continuously asked the most basic question: What makes great software? After some reflection my answer is quite simple.

1. A crisp understanding of the problem
2. A deep understanding of your users and their tasks
3. Design strategies that work the way humans work
4. A small team with passion and skills to solve the problem
5. An environment where people are truly productive

Nope. Web 2.0 didn't make the list. Why? In short, long after Web 2.0 has evolved into Web 3.0, etc. items 1-5 will still likely be true and valid. Until then, try using the Web 2.0 approach along with the basics of good design and you may just be able to apply a handy Botox treatment to spruce up the usability of your enterprise applications. After all, giving a 360 degree view of your customer should at least look good ☺

About Us

We spend most of our time training and consulting with Global 2000 clients and software vendors to help bring the world more usable software. We have fixed and mobile usability labs to validate our findings and we help them communicate good design solutions and guidelines with our own enterprise software, GUIguide.

About the author:

James Hobart is an internationally recognized user interface design consultant based in California, USA. He specializes in the design and development of large-scale, high-volume client/server and web applications. He is an expert in GUI design for transaction processing systems and strategies for migration to thin-client graphical user interfaces. He can be reached at jimh@classicsys.com

Find more articles on usability and design at www.classicsys.com