

Design vs. Redesign Pt.2 - What Does It Mean?

Contributed by Dex
Wednesday, 17 October 2007
Last Updated Thursday, 18 October 2007

So, in Part 1 of Design vs. Redesign I told you to pay attention to good design up front. But what does that mean?

A fair question. Basically it means:

- Make sure you have requirements--both business and user.
- Make sure you have at least a skeletal design to work from (wireframes are great for this).
- Make sure the design fulfills the requirements.

Requirements Definition

Most business or product owners are pretty good at stating their high-level requirements, where "high-level requirements" are really their business goals. Obviously these are important, because if you can't meet your goals, you can't make your money. However, they aren't the end of the story, and they aren't going to magically lead you to the successful end-product promised land. Break those goals down into smaller pieces--the data needed, the tasks to be performed, the business processes that need to be supported. Be as specific as you can be. For instance, if you already know that for billing or accounting purposes only one account number can be associated with a given named entity, say that. Keep in mind that your requirements have to get implemented into code at some point, the less ambiguity you can hand a developer--particularly if you're contracting out the development--the better off you, and your software, will be. (Note: defining business requirements for a software selection project is different from defining business requirements for a software development project. I'll address the build vs. buy dilemma another time.)

But even that's not the end, because without identifying your audience and defining their requirements, you'll never build a product that they'll actually use. And if they don't use the product, all of your development work will be for nothing--well, nothing except keeping you away from the promotion or end of year bonus that you crave so much. Get some understanding of your users, their environment, and how they work. Write some quick personas to make them real. And then, most importantly, figure out how to meet their needs and still achieve your business goals. Design to the user, don't make the user conform to your preconceived notions of what is right. Users really don't care what you think is right, they only care about what they know is right...and believe me, they almost always know better than you. Just ask them. A helpful hint: Keep your business logic out of the UI. It will make it far less painful to make changes as you move through the development process and discover new requirements and needs.

Seeding the Design

I know some people will read this and think that I'm advocating a front-end heavy "waterfall" method of development. So, just to be perfectly clear, I'm not. I do believe that it's possible to start the design process up front and still work in an iterative, or even Agile, fashion. You don't have to have a full-blown, pixel-perfect design before your developers get started. You do, however, have to have the beginning of a design so that they have a place to start. (A helpful hint: Keep your business logic out of the UI. It will make it far less painful to make changes as you move through the development process and discover new requirements and needs.)

I've found that one of the best ways to "seed a design" is to provide a set of wireframes to the development team. These wireframes are design agnostic, and are only meant to provide a visual framework around which a finished graphical look-and-feel can be constructed. They're meant to let the developers know what kinds of elements need to be on a screen, whether forms or form elements will need to be constructed, what kind of data is meant to be collected, and how the user will interact with the screens, or how the elements will interact or impact each other. While they don't address things like color, graphics, or style--all of that can come later--they do give the developers enough to work with that they can start

constructing screens and the underlying services and/or data repositories that will be needed.

Better to Know Now

If you've missed the mark, either on the business or user side, wouldn't you want to know before you've spent those hours and hours (and dollars upon dollars) getting every pixel the right shade of pink (kidding...much to my personal annoyance, almost no one makes pink software) and placed perfectly on the screen? If you didn't answer yes to that question, I'd like you to stop reading now and just go turn your computer in to whomever you work for--and if you're self-employed or at the top of the food chain at your organization, you can just send your computer to me, I'll put it to good use--because apparently your momma never taught you the value of not throwing money away, and there's probably nothing I can say that will remedy that situation.

Do a quick check of the business and user requirements against your early design. Do you see any glaring holes? No? Great, get those code monkeys to work. But don't make the mistake of not continually checking your requirements against the design. As the screens start to come together, get the app in front of some users. Put it in front of the business. Get feedback early and often. And refine, refine, refine.

The probability that you're going to find that some of your assumptions about the user were wrong is high. That's nothing to get freaked out about, because you can adjust. And if you listened to me about starting from a skeletal design, you're not going to spend needless hours editing and re-slicing graphics, twiddling around with stylesheets, or re-coding business logic just to alter or expand your design.

Up Next: What happens if you don't pay attention to good design up front.