



Common Challenges to Creating a Positive User Experience

When people try your software for the first time, they'll be comparing it to every other application they've ever used, so it has to deliver a positive user experience (UX). However, when it comes to UX design, developers often drop the ball because they're unable to recognize and avoid some common pitfalls. The purpose of this white paper is to describe some of those pitfalls so you'll be able to screen out unqualified firms during the vendor selection process and choose a technology partner that knows how to build high-quality, usable software.

Introduction

Imagine:

Your new software application is poised to be a great success, but then, out of the blue, disaster strikes. What's gone wrong?

The software developers seemed competent. Not only that, but you've been with them every step of the way to ensure that business requirements were being satisfied. There have been no major hiccups or costly do-overs. In fact, the project officially ended on time and on budget. It was perfect in every way, right up to the very end.

Then you made a horrible discovery: Users hate your application!

Does this scenario sound familiar? Hopefully not. Nobody who manages software projects would want to find themselves in this situation, but it does happen.

Can you prevent such a calamity by hiring a design agency to create a pretty user interface (UI) for your application? No, probably not, because pretty software can be just as hard to use as software with a plain UI (or harder). Instead, your goal should be to build an attractive, well-engineered, and extremely *usable* application—like the software that Apple has become famous for on bestselling devices like the iMac, iPhone, iPad, and iPod.

Graphic design and user experience design are very different disciplines. The former is concerned with looks, the latter with results.

User experience (UX)—usability—will be a deciding factor in the success of almost any software application that you choose to develop. Ignore it, and you risk alienating users and jeopardizing acceptance of your application. Embrace it, and you at least improve your odds of developing an application that users will buy into.

That's not much of a slam dunk, though, is it? Sadly, incorporating user experience design into the software development process just isn't enough to guarantee that the job will get done right. Why not? Because it isn't easy to build usable software. When the echoes of a developer's flashy sales pitch have died away and it's time to execute, a lot can go wrong—but it doesn't have to.

If you know about the UX pitfalls that development firms commonly stumble into, then you can factor that criteria into your vendor selection process and make a better choice.

This white paper identifies common challenges that prevent some development firms from building great software that users will embrace.

Design, in its broadest sense, is the enabler of the digital era—it's a process that creates order out of chaos, that renders technology usable to business. Design means being good, not just looking good.

- Clement Mok
Designing Business¹

Adoption is key to the success of products and services. When clients come to us to evaluate a concept, prototype, or completed product, the evaluation really boils down to one fundamental question: Will people use it?

- Madrigal and McClain
"Barriers to Adoption and How to Uncover Them"²



Defining a Positive User Experience

Unlike so many other aspects of software development, user experience is entirely subjective—and that’s a problem.

What exactly does “good” look like? How would you define “bad”? More to the point, how do you measure it and suggest improvements?

Not many people will dispute that there is such a thing as good software—and bad software—but on what basis can you undertake a substantive evaluation? A simple thumbs-up/thumbs-down rating system is fine for the latest movie, but it clearly has no place in a complex software development project. You need to know specifics before you can attempt to fix what’s broken, add what’s missing, or rearrange elements that seem out of place.

Although user experience is, by definition, subjective, that doesn’t mean it can’t be qualified in a meaningful way. According to usability guru Jakob Nielsen, usable systems have five essential characteristics.

Usable systems are:

1. Easy to learn.
2. Efficient to use.
3. Easy to remember.
4. Difficult to mess up (low error rate).
5. Pleasant to use.⁴

This five-part definition of usability is, of course, subjective, but it’s also clear and sensible—and it can form the basis for user tests that will yield *quantitative* results.

So what kind of software application will deliver a positive user experience?

Two kinds, actually. The first kind is software that’s been optimized through user testing and observation to work the way users do. The second kind is software that’s been developed using libraries with built-in UX best practices and then optimized through user testing/observation to resolve specific issues that the libraries don’t cover.

Google is a rather extreme example of a company that relies on user testing—you might even say crowdsourcing—to optimize its designs. According to Irene Au, Director of User Experience at Google:

More than anything, Google prefers to make design decisions based on what performs well. And as a company, Google cares about being fast, so we want our user experience to be fast. That’s not just in terms of front-end latency—how long it takes the page to download—it’s also about making people use their computers more efficiently.⁵

Apple’s wildly successful UX design activities are in stark contrast to Google’s minimalist, design-by-numbers process. Apple relies on talented graphic designers, UX specialists, and, of course, developers to build popular products that deliver a rich user experience.

The user already has a mental model that describes the task your software is enabling. This model arises from a combination of real-world experiences, experience with other software, and with computers in general...

Before you design your application’s user interface, try to discover your users’ mental model of the task your application helps them perform.

- Apple Human Interface Guidelines: User Experience³

[At Google] engineers and analysts pore over streams of data to assess the impact of experiments with colors, shading, and the position of every element on the page. Even changes at the pixel level can affect revenue.

- “Google’s Irene Au: On Design Challenges”, Bloomberg Newsweek⁶



Common UX Challenges

When you're formulating a risk assessment for your upcoming software project, you should factor in common challenges that sometimes prevent developers from building usable applications.

Here are some of those UX challenges:

Using an Inflexible Waterfall Development Process

Waterfall development is an outdated methodology that many software firms still insist on using. It's a prescriptive development process, meaning developers define all requirements up front and get their customers to sign off on them. Then the developers completely design the application, get the blueprint approved, and move on to writing code, testing, etc. This traditional "big design up front" approach can work well for certain projects with very fixed requirements.

However, in the time it takes to develop a *typical* software application, many of the business requirements will undoubtedly change, and software usability will suffer simply because of the mismatched functionality.

When the application has been completed and it's ready for user testing, the code is, essentially, already frozen because substantial changes would be costly, technically problematic, and potentially disastrous for the project schedule. If user testing is undertaken earlier in development using prototypes or mock-ups, feeding the results back into the process would also be difficult for the same reasons.

To produce a positive user experience, developers must be able to feed user data back into the development process and make appropriate changes *whenever necessary*. A waterfall development approach simply lacks the flexibility that's needed to make a software application truly usable.

Not Focusing on the User

Many software developers fall into the trap of looking at their projects through a set of "code goggles" that filter out everything but the technological challenge in front of them. In other words, their decision making is primarily influenced by programming considerations and they have a tendency to lose sight of other equally important elements—like the user experience.

Such developers have a tendency to define requirements in terms of what the application must do instead of thinking about what the user needs to do and why. These developers may also lack the soft skills necessary for effective requirements gathering, a problem that can also impair usability.

Failing to Appreciate the UX Design Value Add

This challenge is one that can originate on the developer side, the customer side, or both.

*For 50 years, almost all experiences have shown that traditional **waterfall development** methods result in a poor user experience. The reason is simple: **requirement specifications are always wrong.***

- Jakob Nielsen
"Agile Development Projects and Usability"⁷

Recognize that, as an application developer or interface designer, you have a greater wealth of knowledge and a more intricate understanding of your application than your customers are likely to have... remember that you are not designing the program for yourself. It is not your needs or your usage patterns that you are designing for, but those of your (potential) customers.

- Apple Human Interface Guidelines: User Experience⁸



A development firm that has no concept of what proper UX design can do for their customers (and perhaps doesn't care) builds software that's good enough to satisfy the terms of a delivery contract. Such legalistic firms are implicitly saying to their customers, "We'll build an application that meets your requirements, but it's your problem if people don't want to use it."

On the other hand, development firms that do understand the value of UX design can have their efforts frustrated by a client that sees UX design as a mere gimmick or just another up-sell opportunity instead of a necessary part of the development process. Of course, the third scenario, the "blind leading the blind" scenario, where both parties are ignorant of UX design (but might think themselves experts), can be equally, if not more, disastrous from a user's point of view.

Involving Users Too Late in Development

Even a software development firm that's mindful of UX design can make the mistake of waiting too long to test users, observe them, and solicit feedback from them. Users must be involved right from the beginning, at every stage of the development process, if you want your application to really meet their needs.

Replacing User Testing with User Opinions

Although direct user feedback can be helpful, it should not become a substitute for testing and observation. What users say they need and what they actually need can be very different because they're operating under a specific set of assumptions based on how they currently perform tasks. (For example, before Henry Ford popularized the automobile, impatient travellers would have "needed" a faster breed of horse, rather than a car.) Testing and observation can disprove some of these assumptions and point to better ways of doing things that might never occur to most typical users.

Firms that download UX design responsibility to their developers are more likely to fall into the trap of asking users rather than testing them because designing/administering user tests and observing user behaviour requires a specific skill set, while anyone can ask, "What do you think of our application?"

Taking a Stick-on Approach to UX Design

Some developers make the mistake of dissociating back-end software—what they would consider the "real" software—from the user interface, which they regard as a mere façade that they can stick on when development is winding down. (In reality, the front- and back-ends are two sides of the same coin—integrated, indivisible.)

The "stick-on" approach involves building software that satisfies business requirements alone, regardless of how complicated and unintuitive the application may become for users. With the main code set in stone, developers simply add an attractive user interface and thus fall prey to the

...we make a point of not relying entirely on users' self-reported data. A little objective observation of actual user behavior goes a long way and can uncover substantial differences between what users do and what they say they do.

- Madrigal and McClain
"The Dangers of Design by User"⁹



assumption that users won't mind negotiating a labyrinth so long as it has fresh paint and wall paper.

Developers that take this approach have got it all backwards. The onus is on them to develop an application that works the way users do because users are under no obligation to tolerate bad software, even if it is pretty to look at.

Succumbing to Resistance from Software Developers

When UX design considerations increase the complexity of software code or the development workload, developers may decide to push back. When this happens, UX specialists must be prepared to explain their design choices and, if necessary, suggest alternatives that will be acceptable to developers but not compromise the user experience. Depending on the politics involved, there's always a risk that users will take a backseat to other, short-term (cost) considerations.

Failing to Resolve Iterative-Holistic Conflict

UX designers take a holistic approach to software. They identify the most likely pathways that users will take through an application while performing specific tasks, and then they seek to optimize those pathways. In contrast, Agile developers build software incrementally, sprint by sprint, adding one piece at a time. Some UX designers find it difficult to parse their holistic view of the application into workable chunks that align with an iterative development schedule.

Lacking UX Design Competence

Simply understanding the value of user experience design is not enough to get the job done. A software development team must include competent, experienced UX designers with a strong track record.

Leaving UX Out of the Budget

Like every other aspect of a software application, the user experience can be compromised if the project budget does not include the cost of UX design activities, user testing, and the resultant development effort. UX design should be seen as an investment because the total ROI of the software application ultimately depends on the level of user acceptance. Keep this in mind during the vendor selection process because the lowest bidder may not be offering the best value. Factor UX design into your project plan and budget.

...the best tools are those that users are not even aware they are using.

- Apple Human Interface Guidelines: User Experience¹⁰

Conclusion

Developing software isn't easy. Developing great applications that users love is even more difficult because it involves finding user-centric ways of translating business requirements into working code.



If you want to build an application that delivers a positive user experience, you should be aware of the challenges involved before you start the vendor selection process. When you understand the UX pitfalls that ensnare some developers, you'll be more likely to choose a technology partner that knows how to avoid them and deliver great software that works the way users do.

References

1. Gabriel-Petit, Pabini. "Design Is a Process, Not a Methodology", *UXmatters*, July 19, 2010, <http://www.uxmatters.com/mt/archives/2010/07/design-is-a-process-not-a-methodology.php>, accessed on March 25, 2011.
2. Madrigal, Demetrius, and Bryan McClain. "Barriers to Adoption and How to Uncover Them", *UXmatters*, November 8, 2010, <http://www.uxmatters.com/mt/archives/2010/11/barriers-to-adoption-and-how-to-uncover-them.php>, accessed on March 25, 2011.
3. *Apple Human Interface Guidelines: User Experience*, (Cupertino, California: Apple, 2009), 39.
4. Nielsen, Jakob. *Usability Engineering*, (San Diego: Academic Press, 1993), 26.
5. "Google's Irene Au: On Design Challenges", *Bloomberg Businessweek*, March 18, 2009, http://www.businessweek.com/innovate/content/mar2009/id20090318_786470.htm, accessed on March 25, 2011.
6. "Google's Irene Au: On Design Challenges"
7. Nielsen, Jakob. "Agile Development Projects and Usability", *Alertbox*, November 17, 2008, <http://www.useit.com/alertbox/agile-methods.html>, accessed on March 7, 2011.
8. *Apple Human Interface Guidelines*, 25.
9. Madrigal, Demetrius, and Bryan McClain. "The Dangers of Design by User", *UXmatters*, March 7, 2011, <http://www.uxmatters.com/mt/archives/2011/03/the-dangers-of-design-by-user.php>, accessed on March 25, 2011.
10. *Apple Human Interface Guidelines*, 46.





Architech Solutions is a Toronto-based technology consulting and software development firm. We design and build powerful, user-centred systems that work. We're agile, disciplined, and passionate about delivering for our clients.

Architech Solutions
70 Bond Street
Suite 400
Toronto, Ontario
M5B 1X3

Phone: (416) 607-5618
Fax: (416) 352-1768

To book a free on-site Discovery Workshop led by our team of consultants, e-mail info@architech.ca or visit us at www.architech.ca

© 2011 Architech Solutions Consulting Services Inc. All rights reserved.

