Open Source is Ugly: Thoughts on Ehancing GUI

Garth Braithwaite – Interview with Opensource.com

Garth Braithwaite (131), being both a designer and engineer, has made significant contributions to various open source projects, including founding the Open Design Foundation (132). In this short interview with Aleksandar Todorović, on behalf of Opensource.com (133), Braithwaite discusses the lack of professional designers in open source projects as a major reason for poor user experience (UX). He believes, while not all open source projects may require extensive UX or UI design, open source software, when targeting a broader or less experienced demographic such designers unfamiliar with open source software, will logically benefit from good UX design.

Why is UX bad in so many open source projects?

GB

There are a lot of reasons, but one of the largest contributing factors is the lack of professional designers contributing to open source projects. Contributing to the lack of designers, there is also a lack of collaborative and open source design workflows. Secondary to that, there are open source project owners who are unaware of the value of design or are unsure where to start with the design process.

How important is it for an open source project to have a good UI and UX?

GB

Not all open source projects need more UX or UI then they currently have. Often times developers build open source

projects that are aimed at other developers, so they are able to consider the needs of the end user without additional design assistance. The problem occurs when the open source project is being used by an outside demographic, including by developers of a lower experience level. In these cases, good user experience design contributions will help define the target audience—their needs, struggles, and experience—and the recommended solutions for assisting the users.

Good user interface and brand design can also help establish a consistent experience across the project and help attract new contributors.

Is it easy to attract designers to participate in open source projects?

GB

No. Often times it is easier to find open source developers who also have design experience.

What should developers who can't attract designers do?

GB

They shouldn't wait around. If they can hire a designer, great, but in the case of most open source projects where budget is small to non-existent, developers should look to improve their own design skills. Design is a process of identifying and solving problems, something with which developers are familiar. Developers have the ability to aquire at least basic design skills like any other skill: with practice, research, and community support.

Are there notable open source projects with good UI and UX?

GB

There are some great ones out there—particularly ones that overlap somewhat with the design community, like Sass *(134)*, Bower *(135)*, Ember *(136)*, and others. There is a great collection of open source projects with beautiful UI and UX at Beautifulopen.com *(137)*. There are also more mainstream examples like Firefox, VLC, and
others.

The Dutch designer and author Koos Looijesteijn (138), equally discusses the common perception that open source software is poorly designed and difficult to use (139). However he argues that this perspective is unproductive and instead proposes the question: "How can volunteer-driven projects become successful?" He explains that well-designed, while successful open source projects do exist, they often have professional organizations behind them. Looijesteijn suggests that when people criticize open source design, they are usually referring to software that is mainly developed by volunteers.

Balancing Design and Development

The implementation of these designs heavily rely on Free and Open Source projects. That means that much of the economic value added by software companies isn't in software development. They identify problems that people have, design and test solutions. The implementation is often little more (sidenote: My former software engineering coworkers used to joke that they don't really program anymore. That they're more like plumbers piping together stuff that's been made before them.) than bundling a bunch of FOSS framework together with custom UI.

Software developers often divide up work into tiny problems. For each problem, they create a ticket. Often it doesn't matter who picks up the ticket—it could be one of thousands of volunteers who just happens to look for a problem to fix on a Saturday afternoon. They write a test, write the code for it and submit it.

Great design isn't done by picking up a tiny issue and creating a solution for it. It's done by gaining deep under-

standing for the problems people have, the context in which they use products and by testing solutions with people. It's labor-intensive and requires people who are engaged with their users over long periods of time.

And once a designer has designed and tested a solution they believe in, they have to convince the people building it. In a professional environment, they typically report to someone who can decide when a design is well enough tested to be implemented. In volunteer-to-volunteer situation, they just have to hope someone feels like doing it. If the frontend developer is like, "Yeah, but I like it the other way", designers are shit out of luck.

Software developers use automated tests to proof that their code is good enough. Good enough usually means done. One problem, one solution. Designers usually create dozens or hundreds of solutions for the same problem. Except for the simplest of problems, they usually can't automate the evaluation of their solutions. Although they methodologically and iteratively improve and evaluate solutions, they can't proof that their proposed solution is better than the idea of a random bystander.

I've worked with developers who had so little trust in the design team, that they sometimes flat out refused to follow the spec. A designer can't force a developer to implement their solution. All they can do is discuss the situation with the team lead and hope for the best. In most volunteer-driven projects, there isn't even such a structure.

What Were Designers Doing All That Time?

We designers sometimes tell ourselves that our profession is new and unknown. And it is new, compared to architecture from which big branches of design emerged. But before software needed design, the design profession had existed for at least a century—depending. Many of the disciplines that today make up user interface design are old. People were designing dashboards in industrial design, and layouts and communication systems in graphic design. Design was also used a lot in marketing contexts. In that age, design was often done by specialized agencies, billing by the hour. Now this is all before my time, but I imagine that billability became a structural part of design culture: you're my boss and if I work for you, you pay me.

Industrial design usually requires huge investments for a product to go to production, even if it's uncertain if the product will work and be sold. Which is the reason physical designs can be patented. Having patents on your work is definitely a status marker, which too has affected design culture.

Designers typically don't chose the profession to get rich. They usually get paid less than their peers in engineering and often report to someone who's not a designer: someone from marketing product management or business. They chose design, because they feel it's important that someone comes up for the interest of users, for example. But every generation of designers discovers that other people are in charge and that these people don't understand what we do. The struggle of our profession is with the system in which we're employed.

With software it's clear: when you don't have programmers, you don't get a computer program. But when a team is employed to develop a product without designers, a design gets made nonetheless. Accidental design.

Bad design, most likely. But with no designer around and no one listening to users, the client is not going to notice.

When you're struggling to explain why you should even be included in product teams, it's difficult to also argue that the team's work should be made for free for the whole world to use.

So What Do We Do About It?

When I first started to research the topic of open source and design, I was a little frustrated. I wanted to switch from Windows to Linux. But the open source photo editing software Darktable was so unpleasant (sidenote: I'm not saying Darktable is bad or anything—I couldn't enjoy my hobby with it.) to use compared to the closed source Lightroom, that I gave up. If only a few more designers had discovered Darktable, I thought. If only more designers would be in FOSS, we'd all have control over our data and be free from big tech.

But with the historical context in mind, I now accept the situation we're in. It's not a matter of developers acting just a little friendlier or designers acting a little less egoistically. We'd have a very different expectation of UI design today if it weren't for all the money that was poured into software design. FOSS is great, but a well-designed UI can be the difference between a user base of a few hundred programmers and one of hundreds of millions of people.

Well-designed UIs don't just emerge from spontaneous volunteer contributions. A designer could spend most of their working time volunteering and making a great design for a FOSS project. But then still there needs to be a structure and a collaboration process wherein that design is used.