Involving Reflective Users in Design

Paula M. Bach & Michael Twidale
University of Illinois
{pbach, twidale}@uiuc.edu

ABSTRACT
We draw on the idea of the reflective practitioner to consider how end users can directly contribute to user experience design discussions in open source projects. People with expertise in their own use context but without programming or user experience analysis and design skills can provide reflections on personal experiences.

Author Keywords
FLOSS, reflective practitioner, user experience

ACM Classification Keywords
H5.2. User Interfaces: User-centered design

General Terms
Human Factors

INTRODUCTION

Free/Libre Open Source Software (FLOSS) projects can have usability problems, particularly for people with less technical skill than developers [6]. Consequently a number of projects have tried to involve people with user experience (UX) skills. However, for various reasons, UX designers are relatively scarce in FLOSS projects.

HCI research has over many years shown the importance of involving users in the development of successful applications. Could involving more end users help with this scarcity? In this paper we argue for the possibility of involving people in the usability process even when they lack formal UX analysis and design expertise. To do so, we make an analogy with the FLOSS code development process. Although the most valued contribution is code, the submission of suitably formed bug reports is very useful. Similarly end users might contribute their personal user experiences, including personal usability bug or confusion reports, which could be aggregated and used to inform design.

The challenge is to consider how people with fewer technical development skills can make a useful contribution without major commitments of effort, learning and enculturation. We suggest the model of Schön's reflective practitioner as a way to solicit and frame useful personal usability experiences.

CHI 2010, April 10–15, 2010, Atlanta, Georgia, USA.

RELATED WORK

Schön characterizes the expertise of practitioners by their propensity and ability to reflect on what they do, while they are doing it. When everyday practice results in the expected, a practitioner goes about her business, but when something surprises her, she "attends to the peculiarities of the situation at hand" [7]. In situations of surprise, whether good, bad, or indifferent, reflective practitioners engage in a process of framing, hypothesizing, and understanding.

Reflective practitioners *frame* a problem through a process called *problem setting*. Setting a problem involves finding a context and a name for the things surrounding the problem. The practitioner tests a framed problem and experiments on the frame to see if it holds. Schön calls this testing *hypothesizing*. The practitioner tries to *understand* the situation through hypothesizing and continuously tests and reframes the problem to discover the ensuing consequences and implications.

Fischer, in a number of papers (e.g. [2]) has advocated for meta-design to enable end users to be more directly involved in software development. As he notes: "design materials and the externalized representations are essential to design as a reflective conversation". This creates a need to explicitly provide resources and interfaces that can act as effective externalized representations and support reflective conversations. In the case of distributed software development settings such as FLOSS, these conversations are not just those that Schön notes between the practitioner and the materials, but also between the practitioner and the community of other practitioners and the larger software development community. In the case of extreme programming, pair programming strongly encourages reflective practice. It may be that the externalizations necessary for distributed software development afford certain kinds of reflection [3].

Sengers et al. [8] explore the larger issue of reflective design. Although they concentrate on the importance of designers reflecting on their own design activity, they note the importance of involving users, including giving them license to participate, and designing to inspire rich feedback from users, including users' reflection.

In a study of del.icio.us, Hendry [5] found that users discuss features in several ways including feature requests without justification and justified by appeals to personal experience, observed use, and use scenarios. He notes that the discussions reflected much creative work and "sustained reflective conversation" supported and facilitated by the

leader of del.icio.us. Hendry contrasted the ordinary way with which the community sustained reflective conversations to the proposals for systems to be carefully designed to support creative and reflective processes.

THE STUDY

Is there any evidence that there are already reflective user behaviors in a FLOSS project? If so, studying them might gain insights into broadening this kind of participation. We chose to consider UX discussions within OpenOffice.org (OOo) - a productivity suite of applications intended for use by a broad range of end users. OOo is organized into a number of projects, one of which, the UX project, launched in January 2007. The UX project has a wiki page with clear entry points for users interested in contributing. The open invitation for UX contributions from users results in contributions from user/developers who write code, UX designers paid by Sun Microsystems who comment and engage in design activities, active users who ask questions and comment on the ongoing discussions [4] and, the focus of our investigation, reflective users. The openness of the OOo UX project, and its rich mailing list design discussions including design proposals submitted by users make it a good place to study current reflective users.

Users have the option to register as a member of the UX project. As of mid-Sept. 2009, membership totals 335. Registered members can choose to provide information about themselves, and 59 have chosen to do so. The list has nine Sun employees: seven UX engineers, including the UX project lead plus an OOo product manager. Twelve members self-identify as a user, of which only seven actually posted anything. We chose to study in-depth self-identified users who were also active posters (in the top ten) in either of two relevant mailing lists. This yielded 4 people. Frequent posters are involved in discussions in which reflections are likely to emerge. As such, we are focusing on people who are more likely to be skilled at reflection as a means to understand how to encourage more of the productive behaviors identified. We added one more very active poster who did not self-identify as a user, and did not appear to be a UX practitioner, or affiliated with the Sun UX team.

RESULTS AND ANALYSIS OF THE REFLECTIVE USER

The discussions on the mailing lists and related artifacts generated from these five users illustrate three key characteristics of Schön's description of a reflective practitioner: framing problems, hypothesizing, and understanding.

Reflection depends on actively engaging in problem solving and the distributed, collaborative nature of solving problems in open source mailing lists necessitates a cycle of reflective activities. We found examples where reflective users reframed a design problem through prototyping; hypothesized about use; and constructed understandings of use through discussion with other members on the OOo UX project mailing lists.

Having had an established a priori framework, we extracted key concepts from the reflective practitioner framework. With the five users we identified, we read through their posts and tagged examples of framing, hypothesizing, and understanding. We made two passes through the discussions comparing the effectiveness of our tagging and present some of the tagged examples here. Overall from the five users we identified 14 instances of hypothesizing, 26 instances of framing and 11 instances of understanding.

This initial example below begins with a pleasant surprise that launches reflections about use and the testing of a reframed problem. A first user posted mockups that showed a solution for quick navigation between open windows. A second user responded positively to the mockups (shown below), pondered a consequence and then with the mockup open conducted a test of the idea. This element of surprise starts the cycle of reflection.

> Very nice! […]
>
> Side note: expect some performance hit on computers that aren't suitably specified. […]
>
> Incidentally, as I'm now typing to the left, with [user's] illustration to the right, I repeatedly imagine that the preview is rotated a little, in harmony with the 'fan' from which it springs. In fact, it's upright, but I can't shake the imagined rotation from my mind's eye.
>
> Would it be a good or bad thing for fanned previews to match the angles within the fan? Instinct tells me that legibility is *lost* if previews are rotated in any way.

Like reflective practitioners, reflective users take on a particular framing role. This includes setting up a particular knowledge system, structuring particular problems, employing strategies, theorizing in different ways, and owning particular facts [7]. In the example below, the discussion surrounds a proposal about whether or not to include a ruler show/hide button in the OOo word processing application, *Writer*. One user has posted on the mailing list a detailed rationale for including a show/hide button for rulers. Another user responds (shown below) by reframing the problem (1) and presenting a mockup for a solution to the reframed problem (2). The user then hypothesizes about the benefits for the users of OOo in general, and asks for comments (3).

1	I don't really see why we need the rulers hidden by default. That other office suites prefer that behavior doesn't really matter, and that it wastes screen estate under small resolutions could be fixed by hiding the rulers by default with small window sizes only. Actually, what I would suggest is to go the opposite way and improve the rulers so that people DO use them more often.
2	Here's an OOo mockup I did a while ago: http://[removed for anonymity].
3	The rulers would now surround the page (so it is easier to determine where on the page something would be/is), and a "+" button would replace the current "tab-type chooser," offering the user the options to add columns and sections as well tabs. This could make the creation of tabs, columns, and sections more intuitive, faster, and easier. What do you think?

Schön describes hypothesis testing as consisting of "moves that change the phenomena to make the hypothesis fit...The

practitioner makes his hypothesis come true. He acts as though his hypothesis were in the imperative mood. He says, in effect, "Let it be the case that X..." and shapes the situation so that X becomes true" [7]. Reflective users hypothesize as a way to begin exploring how to reframe a problem and postulate how the reframing might affect both their use and also making generalities about the OOo user community. This hypothesizing involves making a commitment to change the behavior of the software and then using imperative language to state, in the form of a hypothesis, what such a move would entail for use.

The reflective user who posted the reflection below hypothesized about an issue that was submitted to the bug tracker. The issue refers to how the save icon behaves in the OOo applications and that the behavior should be changed. Currently the icon is grayed-out until a change to the document is made. The user who submitted the issue proposes to change this so that OOo users can save whenever they want, even if no changes have been made to the document since the last save. In the example below the reflective user hypothesizes several use scenarios that might occur should the behavior change.

> Consider a corporate environment. […]
>
> What does happen when an unnecessary save operation is executed? Well, surely all concurrent users are affected. Even a perfect implementation (and I have serious doubts that the implementation will work perfectly under every scenario) of this process will generate trouble for the other users: will they need to validate again all the changes they just made, because someone on the other end of the line choose to save the document? What is the error rate during this operation?
>
> And please remember that - because of the nature of this process - a lot of clashes are likely and a lot of hidden errors will be lurking around.
>
> Documents might be huge (e.g. spreadsheets - especially when the current size limits are lifted), and saving might take time, blocking the user from productive work.
>
> What if the document is saved over the network. Not all networks are Gigabit networks. It might slow the whole network.
>
> Distress to users: users will tend to hit the save button every few minutes. By stealing their attention, the users are more likely to commit various errors. Don't underestimate this, as it is my experience that users will focus more and more on saving the document. Guides to show the current state of the document usually have failed, because users need to interpret an additional information, so they just tend to push the "Save"-button.
>
> Changes to a document are sometimes necessary to record. Every change needs to be documented in the field I am working. It must be absolutely traceable. This is not corporation policy, this is European law.
>
> So, there are fields, where unnecessary "Saves" just complicate something that is already complex.

Although the hypothesizing is not in the form presented by Schön, the four scenarios above reframe the problem originally proposed by another user, with each showing a negative consequence.

The last example shows a clear meta-cognitive process of reflection. The user refers to previous thinking activity shared with the community and to other thinking as a "usability analysis" and demonstrates such thinking so that other users can also engage in reflective activities that are helpful for improving the user experience of OOo.

> After more thorough thinking, I decided to revisit my writing. I will add sensible comments later on, but wish now to address some general issues using an analytic approach. I hope that this strategy will also help newcomers to perform better usability-analysis in the future.

The usability analysis serves as a template for shared reflective thinking. In the same example (but not shown here), the reflective user asks questions of a proposed feature or change. Such reflective questions include asking oneself the purpose of the proposed change or feature, whether users can accomplish tasks quickly, and alternative ways to approach the task. Each reflection includes a discussion about possible outcomes and experience drawn from one's own use. Although not quite as sophisticated as the examples from this one reflective user, other users on the list have adopted this reflective user's style of reflection including one of the five we identified to study in-depth.

In reflecting about use, even when derived from their own experience, reflective users, oddly, don't talk about their own errors, yet they will talk about how the general user population might make errors. We found this surprising, given that we expected that it would be easier to describe personal experiences, encountered problems with an application, problems with competing applications, personally desired functionality or usability improvements etc., than it is to consider the needs and possible problems of others (precisely the difficulty expert developers have with respect to more naïve end users). Personally desired functionality is indeed discussed (and justified with personal use scenarios). The lack of discussion of personal use-problems may be due to situations where use is tacit and users' analytic capabilities default to paying close attention to the software's behavior and not their own cognitive errors. As noted, we are not looking at UX experts who are sensitized to diagnosing the cognitive origins of usability problems. It may also be a matter of norms – that usability discussions are seen as about designing for others and that personal confusion anecdotes are not seen as legitimate.

Reflecting about use is also demonstrated by the origin of surprise or the situation that stimulates reflection. Such a situation appears not to be stimulated by user errors experienced by the users, but rather by how they expect the software to behave. This is akin to the open source concept of scratching an itch; that developers are motivated by something about the software that irritates them which motivates them to fix it. Sometimes the itch is a functionality absence that they scratch by building the software and sharing it with others. In the same way, an itch stimulates a reflective user to scratch by reflecting on why a problem occurred and what to do about it. Their result is to reflect in the open and work on coming up with a solution for the entire community even though they cannot implement the fix.

The collaborative nature of the reflective practice in the open we observed is related to Fischer's call to move from reflective practitioners to reflective communities [2]. However what we found is not simply a sharing of the results of reflective practices by people with diverse expertise in order to create a collective understanding of a complex issue from multiple perspectives. It is also a social approach to helping an individual to reflect on her own unique experience.

EXTENDING THE IDEA

The findings reported here are on reflective activities undertaken by existing participants in a FLOSS project. The study is small in scope and studying other FLOSS projects to understand what generalizes would help. In some ways it is deliberately unrepresentative. We chose a project with a substantial commitment to discussing and implementing usability and involving several UX experts. Many FLOSS projects have just one or no UX experts [1]. Nevertheless these preliminary findings show the potential of involving reflective uses in UX discussions. As the examples show, although reflections might exploit specialist expertise about software functionality or usability, they can still be useful even when using more common-sense reasoning, or just describing a personal experience engaging with the software. We think this is encouraging as a springboard for widening participation of similar 'reflective–user reports'

For example, FLOSS projects have effective functionality/failure bug reporting, yet a project could encourage usability bug reporting. This might be an analytic report such as that done by a usability practitioner, or a student taking an HCI class. But it could be a simpler matter of 'surprise reporting', where the users reflect on their experience using the software and describe what they wanted to do (and why), what they tried doing with the software (and why), any diagnoses, remediation, or workarounds they tried and attempts to articulate contributory causes of their surprise, whether derived from confusion or not. In our study we did not find examples of such basic confusion reporting. By contrast these more committed individuals seemed to provide more sophisticated reflections that are closely related to desired functionality.

We think this idea has potential. To work it will need careful sociotechnical systems design: appropriate tools and norms for reporting confusions and reflections, ways to collect and aggregate reports so that they can be acted upon, very low-cost, low-effort ways to get involved initially, guidelines and mentoring for reflective users to improve and get more involved, etc. The study revealed examples of collective support for reflective practice, mentoring and modeling of best practice.

Reflective user reports will need to be well understood to be useful and usable. They are a form of data push rather than the more familiar data pull. In traditional user studies, social scientists carefully sample and study aspects of the use context allowing valid and systematic analysis. By contrast, reflective users will be a self-selecting group, of variable quality and reliability and likely to overlook certain issues and bias emphasis on others. However we have at least the inspiration from FLOSS code contributions and bug reports and other settings such as Wikipedia that widening participation can still be highly effective - so long as appropriate quality control and mentoring mechanisms exist.

CONCLUSION

We see explicit support for reflection on use experiences by end users of FLOSS products (or indeed potential adopters) as a way to further open up FLOSS participation. We can gain insights on how this may happen from reflective practices observed by existing FLOSS participants. The reflective practices observed and the wider activities envisioned are not a personal, private improvement of professional practice. They are a public sharing of those practices as a way to provide user centered input into the software development process, particularly of users' experiences of engaging with the software and of trying to adopt, adapt, appropriate or integrate it into their lives. Not only is the sharing of these reflections by definition social, but the elaboration of the reflections can also be social, supporting the learning of the practice of reflective practice.

ACKNOWLEDGMENTS

This work is supported by the National Science Foundation under Grant #0937060 to the Computing Research Association for the CI Fellows.

REFERENCES

1. Bach, P. M., DeLine, R. and Carroll, J. M. Designers Wanted: Participation and the User Experience in Open Source Software Development. In Proceedings of the CHI 09 (Boston, MA USA, 2009). ACM.
2. Fischer, G. From reflective practitioners to reflective communities. In *Proceedings of the HCI Int. Conf.* (HCII), (2005).
3. Hazzan, O. and Tomayko, J. The reflective practitioner perspective in eXtreme Programming, *Proceedings of the XP Agile Universe* (2003), 51-61.
4. Hedberg, H. and Iivari, N. Integrating HCI Specialists into Open Source Software Development Projects. In the *5th Int. Conf. on Open Source Systems*, (2009), 251-263
5. Hendry, D.G. Public participation in proprietary software development through user roles and discourse. *Int. J. Hum.-Comput. Stud*. 66, 7 (Jul.), (2008), 545-557.
6. Nichols, D. M. and Twidale, M. B. The Usability of Open Source Software 2003 Accessed from http://firstmonday.org/issues/issue8_1/nichols/index.html on Jan. 15, 2006.
7. Schön, D. The Reflective Practitioner: How Professionals Think in Action. Basic Books (1983).
8. Sengers, P., Boehner, K., David, S. and Kaye, J. Reflective design, Proc. 4th Decennial Conference on Critical Computing, 49-58, (2005).