Intentional Unusabilty: Supporting deniability through unorthodox design

Xianhang Zhang
University of Washington
Seattle, WA 98115 USA
Xianhang@u.washington.edu

Copyright is held by the author/owner(s).
CHI 2007, April 28 – May 3, 2007, San Jose, USA
ACM 1-xxxxxxxxxxxxxxxxxx.

Abstract
Social software is different from singe user software because when we use it, we care about how our actions affect others' perception of us. The design features of the software interact with this cognitive, social reasoning process or "theory of mind" and affect user behavior. However, this influence can sometimes be counterintuitive to those versed in traditional interaction design. One important set of social protocols that we use in our everyday lives is plausible deniability - white lies that allow us to hide the true motivations for our actions. This paper shows how plausible deniability can be achieved in social software by directly violating established design guidelines and deliberate usability degradation. Such "deliberate unusability" is a common feature of social software constructed with the theory of mind as a guiding principle and show the need for a new set of design guidelines for social software that take this cognitive modeling into account.

Keywords
Social software, plausible deniability, theory of mind, social reasoning, social psychology, usability,

ACM Classification Keywords
H1.2 User/Machine Systems --- Software psychology, H5.3. Group and Organization Interfaces --- Theory and models

Introduction
Traditional HCI and interaction design has focused around usability. An application is usable if it is efficient, effective, easy to use, fun or some other metric pertaining to the subjective experience of the user. One powerful tool in this approach, borrowed from psychology, is the mental model. Mental models are naïve, cognitive schemas about how objects work and how one interacts with them, like "The progress bar measures how much time is remaining". These mental models provide us with predictions and expectations about the results of an interaction and usability is enhanced if the user's mental model is a good fit with the actual behavior of the application.

This mental modeling approach is effective for single user application interaction but needs to be augmented in the case of social software because users not only have a mental model of the application, they also contain "social mental models" or "theories of mind" of the people they are interacting with. We model other people through these theories like "John thinks he's shy" or "Lisa likes John". However, theories of mind differ from the traditional mental models because minds are also capable of possessing theories of mind. This means such theories can be multilayered and recursive like "John thinks I think he's shy" or "Lisa thinks that John doesn't know that I'm aware that Lisa likes John".

We construct and use these theories of mind to guide our social reasoning process and they form a crucial part of how we decide how to act in social situations. When we are interact via social software, the software modulates the range of interactions that are possible. The design of the software affect what theories of mind are constructed and, as a result, what users will choose to do. Thus, it becomes possible to use these theories of mind to construct a model of user behavior and how it will emerge through social software design as well as how to influence and encourage certain group behaviors through this design.

Plausible deniability
Judging motivations forms an important part of our social reasoning process because motivations allow us to predict how people will react in future scenarios. Plausible deniability is the ability to hide the true motivations of our actions by providing others with a plausible, alternate hypothesis or "convenient fiction" that can explain our behavior. Such motivation hiding acts as an incredibly powerful social tool by allowing us to mitigate potentially socially awkward situations ("Sorry I didn't answer your call, my cell phone was on vibrate ") or giving us an advantage in social negotiations (playing hard to get in a relationship).

In order to support such plausible deniability in cell phone example, the social situation has to be set up so:

- I know "my cell phone is on vibrate" is a convenient fiction for me not answering.
- I know I've told you that my cell phone was on vibrate.

- I know that you can't know for sure that my cell phone wasn't on vibrate.

- Therefore, you are forced to accept my convenient fiction.

It is this need to support convenient fictions that often is at odds with conventional HCI. Oftentimes, effective plausible deniability involves deliberately making software harder to use to enhance the ambiguity present in plausible deniability. This paper details several design mechanisms for supporting plausible deniability by deliberate usability degradation.

Omitting information:
Omitting information is the most direct approach to supporting plausible deniability by directly hiding the information necessary to determine motivation. For example, most email systems don't tell you when an email you send has been read by the recipient. Although this information might be useful to the sender, it would also prevent the recipient from plausibly claiming "it must have got caught in the spam filter" when they would rather not have to bother replying to an email.

Error prone UIs:
Making user interfaces deliberately more error prone can allow users to plausibly claim they made an error when they actually did something intentionally. This can allow users to avoid appearing to be rude when attempting socially awkward tasks. For example, if a group event planning tool had a highly sophisticated, foolproof invitation system; it would be hard to plausibly claim that you accidentally forgot to invite somebody. Subtle UI tweaks that introduce room for error into the system would support such plausible deniability and allow users to "forget" to invite certain people to an event.

Default settings:
Default settings allow us to be ambiguous about whether we agree with the defaults of the system or whether we simply don't bother to change them. For example, if the default action on accepting a friend request on a social networking site is that they can only see a limited part of your profile, then you could alter the default for most of your normal friends so that they can see all of your profile but keep it at the default for certain friends. Those friends who can only view the limited profile would not be able to tell if that was a deliberate decision or carelessness on your part. But such ambiguity can only be achieved if the default setting is plausibly difficult to use. Thus, plausibility can be enhanced by deliberately making the setting more unusable by making it harder to understand or placing it in a more obscure location so that users can plausibly claim "Oh, I can't be bothered changing that".

The nature of a default setting also changes the meaning of what changes in the default represent. Any change from the default indicates that not only do you not prefer the default; you dislike it to such an extent that you are willing to expend the effort to change that setting. If the default setting when adding friends was that they could see your full profile, then by setting someone as limited, you're sending the message to them that "you're so awkward/creepy/unpleasant that I was uncomfortable with you seeing all of my profile". Instead, if the setting was limited by default, then the social message you are sending by setting someone as full is "you're so cool and interesting and close to me

that I made a special effort to give you more access to my profile".

Perceived vs actual usability:
Plausible deniability doesn't have to involve actual usability degradation. What is important is that you believe other people think it is difficult for you to use. This means that it could be possible to take advantage of perceptual biases to introduce perceived unusability without significantly degrading actual usability.

The effect of unusability

Social expression
Supporting plausible deniability also tends to make rude actions even ruder. Because a plausible, polite alternative is present, that I chose not to use it sends the message that I want you to know that my motivations are indeed rude. This is not necessarily a bad thing in social software as it allows users to express a larger gamut of social messages.

Plausibly denying plausible deniability
Designing for plausible deniability is only effective if users are unaware that this was your intention. Once users become aware of this, then such actions become much less credible. Thus, designers themselves need a plausible reason for their design decisions to make their software less usable.

If the initial design of the software is usable, then it is very hard to justify design decisions making it less usable. However, if is hard to use to begin with, then designers can simply claim that improving that particular aspect of usability is not a priority. Designers can also claim their design decisions were motivated by other concerns, a concern for privacy or technical limitations for example which can also limit usability. Finally, if all else fails, then it's always possible to pretend to be bad designers who are ignorant of the design flaws and who studiously avoid investigating them.

Conclusions:

Building social software is very different from building conventional software and a new set of design principles and paradigms are needed for effective social software applications. Rather than focusing on usability, the most important aspects of social software is the facilitating of desirable group behaviors.

In this paper, we present a cognitive model called "theory of mind" that allows designers to predict user behavior based on a set of cognitive reasoning principles. We focus on the particular design problem of supporting plausible deniability in social software and shown how software sometimes needs to be in direct violation of traditional notions of usability to effectively support such behavior. Supporting plausible deniability often involves deliberately making the system less transparent and more ambiguous through intentionally poor usability but such design changes allowed a wider range of social expression to be performed.

In the future, gathering empirical data on how theories of mind interact with software design and how users perceive others through the lens of social software behaviors would allow more accurate and powerful prediction models to be built and a better understanding of the design challenges uniquely facing social software.