Open Door

Brandon Burr, Andy Szybalski
CS 377A Mobile Interaction
9 May 2005

IM software and mobile phones currently allow users to prevent unwanted communication in a variety of ways, for example, via away messages. Substantial research has focused on automating the negotiation of communication and enriching it with contextual information. However, less work has been done to try to lower the barriers to interaction by encouraging wanted calls. Open Door is a system that gives users access to the context of friends and colleagues. We are currently investigating what types of context accomplish this goal. Meanwhile, we must respect users' privacy and avoid the pitfall of "telling users how to be social."

Related Work

Calls.calm. A good example of asynchronous and semi-synchronous person-to-person communication, but not especially good for peripheral awareness of many people's statuses. Mostly designed to alleviate phone tag problems. Discussed further below.

Dodgeball. (http://www.dodgeball.com/) Each user explicitly controls what information gets sent out about him and to whom it gets sent. Even with the "crush finder" feature, you basically control what location information gets sent to your crushes, so stalking using Dodgeball isn't as big a problem. However, there's no possibility for nuance: the same information gets sent to all your Dodgeball friends. Dodgeball is also a *push* system rather than a pull system, so when you check in with your location, that location immediately gets pushed to all your friends as a text message. This can be intrusive in some situations and it reduces plausible deniability since all your friends know exactly when you checked in.

Tang, "ConNexus to Awarenex". Seems like a straightforward translation of the IM system to a mobile setting, although it's cool how it aggregates IM and phone availability into a single interface. It'd be cool if we could use someone's IM status to augment their phone status; we'll look into this.

Thefacebook. (http://www.thefacebook.com/) The "Poke Me" action is an interesting example of pull-based person-to-person communication. Plus, it shows that ambiguity can be a feature rather than a bug, especially in subtle social situations like flirting.

Challenge: maintaining plausible deniability

A problem with sharing context is that users don't always want to share the same context with everyone. We have decided to exclude professional contacts from the scope of this project; this partially solves the problem. However, even among friends and social contacts there are bound to be situations where you'd rather not talk to someone. In these cases plausible deniability plays an important role. Different modes of communication afford different methods of refusing unwanted communications; with IM and landline phones, the preferred excuse is "I wasn't at my computer" or "I wasn't at home." With mobile phones, the preferred excuse is "I was in the shower" or "I was in a meeting/class"; in other words, I was in a situation where I couldn't answer the phone. Sharing one's context poses a threat to this kind of plausible deniability.

In particular, Dodgeball is problematic because when a user Alice checks in with his location, for example "@ nut house", this message is immediately relayed to all of Alice's friends. If Alice's friend Bob then calls on the phone and Alice doesn't want to talk to him, she has little plausible deniability, because Bob knows she is at a bar and has her mobile phone with her. Dodgeball users would probably respond that they only put their closest friends on Dodgeball, so they would seldom want to refuse a call; however, this issue would need to be addressed for any larger and more general-purpose system, and it seems likely that as Dodgeball grows more popular, more users will encounter these types of problems.

Systems like PlaceLab and ICAMS focused on automatically inferring contextual information from location and from calendar information. We have elected not to employ these kinds of context for our project in order to limit the scope of the project. Instead, we choose to make use of different kinds of manually-entered context information; for example, IM away messages, since these can be better managed in complex social situations.

Solution: new kinds of context

One way to avoid unwanted communications is to allow users to share different types/amounts of context with different groups of contacts. However, we suspect that few users would have the motivation to maintain different groups. Moreover, sharing different information with different groups leads to the "triangle problem" of IM, when Alice has blocked user Bob, but Carl tells Bob that Alice is online. In the worst case, this can lead to conflict; in many cases it simply makes people reluctant to use grouping/blocking features.

Another solution is to allow users to share a small enough amount of context so that it is acceptable to share it with everyone, and share it in a *pull* rather than a *push* fashion. Making the system less synchronous restores more plausible deniability. Furthermore, we would like to explore types of context which are more ambiguous. For example, a user's status could be represented by an icon: a briefcase, a party hat, a sleepy face, a grinning face, a sad face, etc. While a party hat or briefcase has a fairly concrete meaning, a grinning face is more ambiguous; while it does not explicitly invite contact, it may still have this effect within a small group of close-knit friends, or between significant others. Even more abstractly, users could represent their status by selecting from among ten different colors. While some colors have a fairly definite meaning, for example, red or green, others could mean different things to different people. For example, two significant others could agree that orange means "call me." They wouldn't need to . In general, it seems possible that uses would agree "off-line" upon the meanings of different statuses. One problem with this system is that it does carry a higher cognitive load since users must remember what different colors mean to different people. These simpler statuses could still be augmented by specific text status messages when desired, however, due to the difficulty of text entry on a mobile phone we think that being able to easily select a less specific status is important.

Within our small network of test users it will be difficult to directly observe the effect of plausible deniability on adoption and usage; however, we hope to gauge this effect by interviewing users about their feelings about sharing different kinds of context.

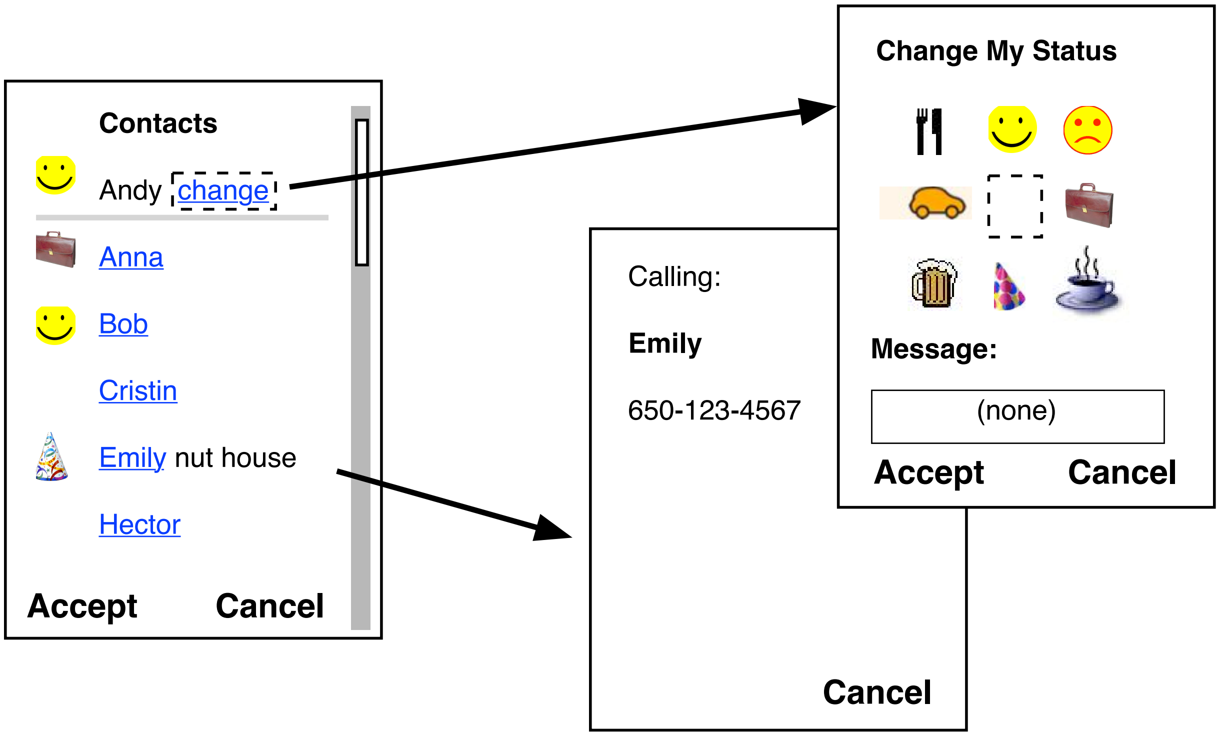
Solution: person-to-person context

The above one-size-fits-all approach does appear somewhat extreme in ruling out sharing different context with different people. We think the approach is warranted since we suspect that most instances where people want more consist of person-to-person communication. For example, telling one's significant other, "call me when you're done with work," or coordinating a planned meeting: "got to the cafe a bit early, cu soon." For these situations, phones are already excellent tools for person to person communication, and when a phone call is too intrusive, a text message will often suffice.

However, this still leaves out the possibility of *pull* communication between two individuals. To fill this hole we are inspired by the "poke me!" feature of Thefacebook. We propose a phone "poke me" feature, where next time a user looks at his contacts he will see a notification, "Alice poked you." For different situations a user could choose to send people a few different types of notifications, for example, "Alice says hi" or "Alice gave you a flower." This type of person-to-person asynchronous communication was well addressed by Calls.calm so we may not actually implement it in our project unless we see a potential for synergy.

Design

The following sketch shows our enhanced contacts list where all users who have registered with Open Door have an accompanying status. In this design the status is represented as an icon with an optional text message. The message could be entered manually or retrieved from IM.



Implementation

We have already implemented the basic functionality of this system as follows:

- A server computer maintains a database of contacts. For each contact we store a name, a phone number, and a *current status.* Every time a user makes a request, a Python script generates a WML contact lists for the user.
- On his/her mobile phone, the user opens the WAP browser. The homepage has been set to his/her personalized contact list, for example:
  http://cgi.stanford.edu/~andyszy/cgi-bin/opendoor/contacts.py?user=Andy
- The page contains *wtai:* links to each user's phone number so the user can make a call with one click.

Plans

We plan to use our system as a platform for investigating how people make use of different kinds of context about each other. Brandon's three roommates, who all have WAP-capable phones, have volunteered to be our testers for this system, and we may add more testers. We plan to do three test periods with different types of context.

- 11–17 May: **Text-only Context:** A user can only select a text message, as in AIM.
- 18–24 May: **Icon Context:** A user can select his status from a set of about 10 icons, as well as entering an optional text status.
- 25–31 May: **Color Context:** A user can select his status from a number of colors, as well as entering an optional text status. We are curious to see if people will welcome these ambiguous messages or be frustrated by them, and if they will feel differently about sharing them with others.

Unfortunately the order of the trials will inevitably affect the outcome as there will be a learning curve, but we're hoping that by putting the most familiar system first and ramping up to increasingly unfamiliar systems, we'll get usage patterns closer to what we'd see in a longer trial that gave people more time to adjust to the system. Also there will be a novelty effect but there's not much we can do about this.

Challenges

WAP is a very slow system for implementing something as interactive as a contact list. (The initial load time is several seconds although subsequent interaction should be possible to do client-side using WMLScript). We're looking into running some sort of polling and caching process in the background to speed this up, since we think a load time of several seconds could be a significant barrier to usage and satisfaction with the system.

In any case we think that network latency will go down eventually to make a system like this more feasible.