

# Talking to Myself through my Alter Ego

Kori Inkpen<sup>1,2</sup>, Mary Czerwinski<sup>1</sup>, Roland Fernandez<sup>1</sup>, Daniel Robbins<sup>1</sup>

<sup>1</sup>Microsoft Research

One Microsoft Way

Redmond, WA 98052, USA

{t-kori, marycz, rfernand}@microsoft.com

<sup>2</sup>Dalhousie University

6050 University Ave.

Halifax, NS B3H 1W5, Canada

inkpen@cs.dal.ca

## ABSTRACT

In this paper we describe two social software applications we are currently working on and explain the challenges we are facing in terms of developing and testing these systems using iterative design. We also describe our current approaches for overcoming these challenges and the tradeoffs inherent in our design choices.

## Keywords

Social software, evaluation, instant messaging.

## INTRODUCTION

Social software is becoming increasingly important and popular in today's society. In May 2007, FastCompany.com reported that Facebook [2] had over 15 million registered users visit the site in March 2007 and that the site is growing 3% week over week [3]. In a June 2006 study of the Microsoft Instant Messenger network, approximately 90 million distinct people logged in each day, producing approximately 1 billion conversations per day [4]. With this widespread popularity of social software, many users are experiencing the benefits of online social spaces and the connections and opportunities that they provide.

Our research group has been exploring several prototypes to enhance social awareness and group communication, advancing today's social software applications. Our initial focus has been on increasing awareness about work and social group patterns of behavior and conversation availability.

This paper first describes GroupBanter and TapGance, two applications we have developed to enhance Instant Messaging. Following this we describe challenges we have faced regarding the development, testing and evaluation of these applications. We then conclude with our goals for the workshop, followed by biographies of the authors.

## GROUPBANTER

GroupBanter is a project that examines ways of extending the traditional one-to-one nature of Instant Messaging (IM) to create new opportunities for serendipitous group interactions. Traditionally, IM applications allow people to view their *friends'* status (e.g., online, busy, offline) and engage in real-time conversations with people on their contact list. Traditionally, IM conversations are private and directed. In contrast, GroupBanter provides people with an awareness of their friends' conversations (i.e., who is

talking to whom) if the conversation has been designated as public.

The GroupBanter prototype is shown in Figure 1. GroupBanter displays all conversations available to a user in the *conversation pane* at the top of the Windows Live Messenger (WLM) window [6]. Users designate a conversation as being public by either clicking on the "group conversation" icon on the main WLM window or by clicking on the "share conversation" icon in the WLM chat window. Conversations that have been labeled public

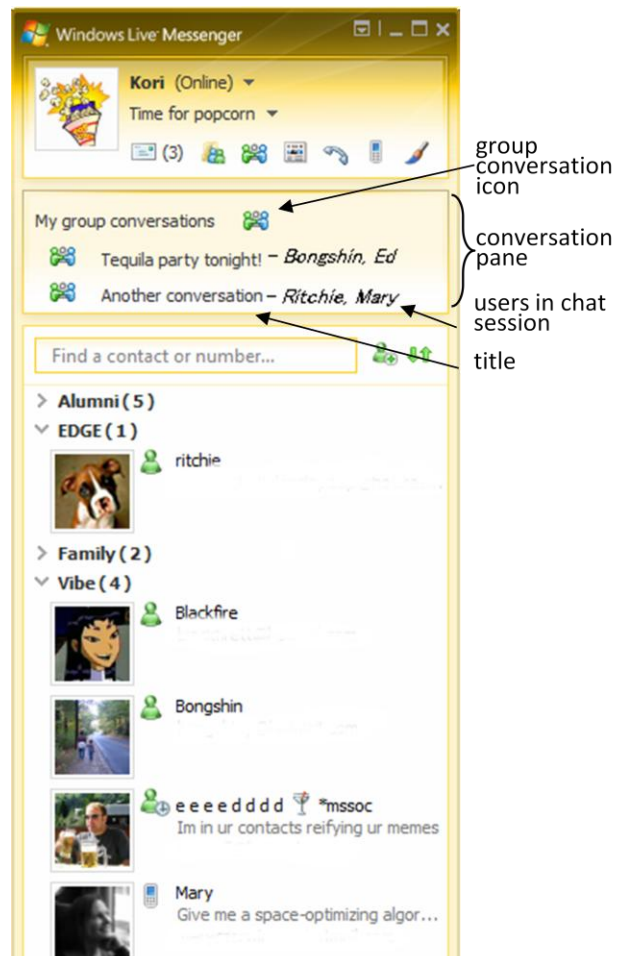


Figure 1. In the GroupBanter prototype conversations are listed with their title (e.g., Tequila party tonight!) and a list of users currently active in the conversation (e.g. Bongshin, Ed).

become visible to others (e.g., people on a user’s buddy list). Anyone interested in participating in a shared conversation can join by clicking on the title in the conversation pane.

The goal of GroupBanter is to provide lightweight support for ad-hoc group conversations, where users don’t need to worry about whom they should or should not invite. Groups can form serendipitously without requiring substantial overhead or pre-configuration. Users can implicitly invite friends to join a conversation just by making the conversation public.

We believe that GroupBanter provides several potential benefits over traditional IM. GroupBanter:

- encourages serendipitous group interactions that might not otherwise occur
- removes the overhead of explicitly inviting everyone who might be interested in the conversation (and managing responses received from each of them)
- makes conversations more inclusive to others
- reduces redundancy between related conversations

**TAPGLANCE**

TapGlance is a project to provide Smartphone users with an easier way to view information and notifications they care most about (see Figure 2). TapGlance recognizes that users in a mobile context often have their attention divided between multiple activities and need their Smartphone operations to be simple and intuitive.

To support the most ephemeral interactions – pulling a phone out of one’s pocket to see if there is anything important to attend to – we have replaced the default Smartphone home screen with a tiled non-interactive display of the most urgent information across several canonical facets. This “glance” view is organized into nine tiles of information. Each tile schematically presents highly salient information in an easily scanned manner. When a

user wants to see details about the information presented in a particular facet, the user “presses-and-holds” a spatially associated key to temporarily zoom in on a particular facet (spatially associated keys for tiles in the TapGlance 3x3 grid are mapped to the corresponding keys on a 3x3 numeric keypad). When the key is released, the screen zooms back out to the tiled home page. This spring-loaded interaction permits users who have a limited amount of attention to quickly glean information from the phone. In contrast, when a user can give their full attention to the phone, “tapping” the spatially associated key switches the phone into that facet’s related application. As an example, in Figure 2a, the top-middle tile of the home-screen is devoted to showing status information about a user’s circle of friends and co-workers. The “glanceable” tile view on the home screen only shows information for one person, possibly the person whose status information was most recently updated. If the user presses-and-holds the key corresponding to this tile (e.g., the “2” key for the example shown in Figure 2), the display temporarily zooms in to show more information about that person and two additional, highly ranked individuals from the user’s social circle. With full-attention, the user can tap the “2” key to enter the mobile version of our social application. In this zoomed in view, the user is again presented with nine tiles where each tile shows high-level information about nine people in the user’s social circle.

TapGlance also enables users to share detailed information related to their current task. This may include information from a users’ calendar (e.g., a meeting) or be related to the current document they are working with (e.g., BracketTracker, or DynaVis as is shown in Figure 2). Additionally, by setting configurations on their PC, users choose what level of abstraction they are willing to share and can manually assign alternative labels to those suggested by the system. In general the application name and window title are used to discern default keywords. There is also an add-in “activity themes” mechanism (with several starter themes provided) that allows users to map



**Figure 2.** The TapGlance prototype shows *glanceable* information on the main screen (a). In the magnify view (b) detailed awareness information is shown for a small group of people. In the focus view (c) users can see high-level status information for several people which they can use to drill down for additional information.

their program names to an activity name, based on themes like “research” or “programming”.

## DEVELOPMENT CHALLENGES

### Building for Windows Live Messenger (WLM)

For the GroupBanter project, we debated whether to use a custom client and service, a custom client with the WLM service, or the WLM client and service. Unfortunately, access to the WLM service from a custom client is not well supported. We investigated using the WLM add-in model, but found it didn’t meet all of our needs. Due to concerns about stability, firewall access, and privacy, we decided to build on the Windows Live Messenger client and service.

We were fortunate to be able to utilize the Windows Live Messenger client and service, particularly since it is a very stable service with well established privacy policies; however, this decision had extra costs associated with it. These costs included learning the WLM build system and codebase, slower build times (1-2 minutes vs. 5-10 seconds for a custom C# client), and learning how to use a custom UI subsystem (without a designer).

For the TapGlance project, we initially used an add-in to WLM on the desktop (to gather the extended status information), a custom client on the phone, and a custom web service on an external server. Due to concerns similar to the GroupBanter project, we decided to switch to the standard WLM service. Finding a way to send extra information that would be accessible to a custom client on the Smartphone was challenging. We eventually were able to access the “friendly name” of each user’s “buddy” and overload this friendly name with the extended status information (our desktop and phone clients are then able to unravel this data and display the normal “friendly name”).

For the TapGlance project, users’ extended status information is exchanged between the users’ desktop PCs (or laptops) and additional user’s who have the TapGlance software installed on their Smartphones. This requires installation of desktop logging software, an add-in to the desktop IM client, a mobile IM client on the Smartphone, and the TapGlance application on the Smartphone. This is a complex process and one that has limited our deployment. Additionally, for any mobile application, it is important to remember that Smartphones are phones first-and-foremost. At every stage of our application design and development we have to make sure that quick dialing and access to incoming calls was not interfered with.

### TESTING CHALLENGES

The multi-user aspects of social software makes testing challenging. Our testing has involved iterating on various design choices, validating that the prototype works as

expected, and testing data logging functionality included for field studies. Additionally, our systems had to be robust enough for many users to be willing to install and work with them over time. We now describe several testing challenges we faced.

## GroupBanter

### *Talking to ourselves*

When testing the GroupBanter application, having users chat with each other was obviously important. In the early stages of development this often meant “chatting” to ourselves. We configured several identities to facilitate this process (e.g., Raven, Blackfire, HarryDogFernandez). Next, our alter-egos needed to have friends. While it was possible to have our various personalities be friends with each other, there were times when it is useful to chat with real live people. Thankfully many of our friends and colleagues don’t mind having several of our alter-egos on their buddy lists.

Once we had multiple personalities, we needed to have them chat with each other; however, most IM clients (and many social software applications) don’t allow multiple people to log in simultaneously on the same machine. Possible solutions to this include running virtual machines that you can quickly switch between, connecting to another computer using a remote connection tool such as VNC, or having additional computers in close proximity. It is also possible to modify the system to enable multiple users to be logged in simultaneously; however, this approach can be problematic because it doesn’t accurately represent how the application will run between multiple computers. We chose to have a separate machine connected via a KVM switch to one of the displays in a multi-display workstation. The second machine had its own mouse and keyboard although an input redirection tool could have been incorporated to re-use the mouse and keyboard from the main computer. This setup allowed us to quickly shift into test mode utilizing our existing desktop configuration.

Voila, we could now talk to ourselves via our alter-egos! (see Figure 3).

### *Real Life Friends*

Some aspects of GroupBanter were difficult to test by only talking with ourselves. For example, we needed to know whether the behavior of our application changed when interfacing with other IM clients (e.g., Adium [1]). Often, in cases such as this, it was easier (and faster) to impose on our real-life friends. Luckily, our friends are very easy going and don’t mind getting strange IM test messages from us (e.g., “ignore this”, “can you tell us who you see in the conversation?”).



Figure 3. Chatting between two of our alter egos.

## TapGlance

### Privacy

Displaying detailed status information about what a user is working on raises serious concerns about privacy. Some users may be comfortable disclosing which applications or documents they are working on while other may not be comfortable sharing this information. In fact, within our own research team, we found that some users thought they would be comfortable sharing the information, but were later surprised when they actually saw their information being shared (or saw how other people used or reacted to it).

While we recognize the importance of protecting users' privacy, we also felt that it would be beneficial to have real status data for early prototypes of the system to determine the accuracy of the status information and the benefit this type of information could provide. As a result, we solicited several people from our research group to install and run a prototype version of TapGlance, despite the fact that we had not fully worked through the privacy concerns. While this deployment was beneficial, we need to address the privacy concerns of our system before we can consider an external deployment of the system.

### EVALUATION CHALLENGES

We also encountered several challenges in preparing to evaluate our social software applications. Given that we have not yet run our user study (it will take place in late July, 2007) this section discusses problems related to the design of the experiment and our preparation to date. We anticipate uncovering more challenges that we will report on once the study has been run.

#### Field Study vs. Lab Study

One of the first decisions we had to make regarding evaluation of our applications was whether to conduct a field study or a laboratory study. Choosing an appropriate

experimental methodology, with reasonable precision, realism and generalizability [5] is difficult when evaluating social software. In particular, the social, serendipitous nature of most social software makes it difficult to examine realistic usage in a laboratory study.

In our GroupBanter project, our main hypothesis was that GroupBanter would encourage more serendipitous group conversations. It is obviously very difficult (and unrealistic) to create serendipity in a lab experiment. Additionally, we felt that real usage was critical for us to understand usage of this software and potential benefits of it. Therefore, we felt strongly that a field study would be the most appropriate methodology.

#### Building Social Software that People will Use

To appropriately test social software, groups of friends need to be willing to install and use the software over a considerable length of time in order for there to be a significant, observable benefit. This can be a huge obstacle to overcome. Often this means building a small, stand-alone application, and trying to lure people to use it or developing an add-in which may be limited in functionality. Both of these approaches can significantly impact evaluation results.

For GroupBanter, we were fortunate to be able to incorporate our prototype into an existing IM client that is used by millions of people. While we were able to instrument most of our desired functionality, we were still limited by the underlying framework of the existing system which made some aspects of the software difficult to implement.

#### Recruiting Field Study Participants (and their friends)

As mentioned previously, in order to evaluate social software in a field study, people must be willing to use the application. Additionally, it is also critical that at least some of the people who use the application are willing to be

social with each other. This is a common challenge when evaluating social software. If the social piece is missing, no one is going to use the system (i.e., if none of my friends sign up for the new IM tool, I won't have anyone to chat with). Therefore, field study participants must be carefully selected.

For the GroupBanter study, we felt strongly that it was important to recruit participants who were already part of a social group that use WLM frequently. However, it can be difficult to recruit groups of friends and this approach causes selection biases. We chose to recruit participants from the summer interns group at Microsoft. Interns at Microsoft are a loosely defined group of people that often want to be social and engage with others in the group. As such, we felt that this population would be receptive to serendipitous conversations. Additionally, the demographics of this group are appropriate for being WLM users and early adopters of social software.

#### *Duration of the Study*

Given that usage of social software tends to evolve over time, it is difficult to determine an appropriate timeframe for a user study. For GroupBanter, since we were selecting participants who already chat using Windows Live Messenger, we felt that there would be less ramp up time needed for our application. However, we were also concerned about the novelty of the application. We therefore chose to first collect baseline IM usage data for one week. Then we plan to introduce GroupBanter and collect two weeks of data. This will allow us to compare across the two GroupBanter weeks to see how usage changes as the participants become more comfortable with the system. Additionally, we can compare each of these weeks to the participants' normal IM usage patterns.

#### *Data Collection*

One of the benefits of social software is that most of the interpersonal interactions are mediated by the application. As such, we can easily instrument the software to collect relevant data. However, we still miss contextual information related to users' interactions with the application such as what is currently happening in the environment and what impact that has on users' interactions with the software.

For GroupBanter we are collecting a variety of metrics such as: the number of conversations a user initiates or is invited to; the number of messages sent; duration of the conversations; whether new people are added to a conversation; who talks to whom; and whether people choose to make a conversation public or join an existing conversation when using GroupBanter. Other data such as whether or not GroupBanter encourages new conversations that would not have occurred otherwise, or changes the people involved in the conversations, will be collected via self reports, such as questionnaires or interviews.

#### *Privacy*

Evaluating social software is particularly problematic in terms of privacy. First, the social nature of these applications makes them very susceptible to privacy concerns. People often do not want to share personal exchanges they have with others. Additionally, gaining informed consent can be problematic. For example, usage data from one participant will often contain data from others who had not agreed to participate (e.g., people the participants chat to). As such we must be very sensitive about the data we collect. For the GroupBanter field study, we have chosen to not record actual chat transcripts. Instead, we are only collecting usage data (as described previously).

Another privacy concern related to the GroupBanter application is the fact that we are proposing to broadcast conversations to others. It is unclear who these conversations should be broadcast to. For example, should they only be broadcast to "friends" on the users' buddy list? Or can they be broadcast to "friends of friends", i.e., everyone who is on a buddy list for one of the participants in the chat session? Given that this is an open research question, we wanted explore it as part of the study. Therefore, we chose a more open privacy model, broadcasting conversations to everyone participating in the study. We will examine the impact of that decision in post-experiment questionnaires and interviews.

#### *Scalability*

In preparing for a user study, understanding and testing the scalability of prototype software is important. For social software that could have millions (or billions) of transactions, scalability is a critical issue. In our GroupBanter study, if we choose to have 500 participants all being part of the same messaging group, each message in a group conversation must be broadcast to 499 other people. If 50 conversations are currently active, this could result in hundreds of thousands of additional messages being sent per hour (which could potentially impact the underlying infrastructure and service).

### **WORKSHOP GOALS**

The first two authors of this submission of have a great deal of experience in evaluation methodology in a variety of other domains including standard HCI evaluation, CSCW, and mobile devices and this experience helps give us insight into solving methodological problems for social software. However, we are still new to this research area and could learn a lot from other attendees at the workshop. Additionally, we are currently preparing to evaluate the systems presented in this paper and will have additional insights on the successes (or failures) of our approaches at the time of the workshop.

### **AUTHORS**

#### **Kori Inkpen**

Kori Inkpen is an Associate Professor in the Faculty of Computer Science at Dalhousie University and a Visiting

Researcher at Microsoft Research. Kori has participated in and organized many workshops at CHI, CSCW, and Ubicomp. Kori's main research area is Computer Support for Collaboration and she has a great deal of experience in the area of experimental methodology, particularly for research areas where traditional methodologies are less effective (e.g., CSCW, mobile interactions).

#### **Mary Czerwinski**

Mary Czerwinski is a Research Area Manager of the Human-Centered Computing (HCC) group at Microsoft Research. Mary is a cognitive psychologist whose primary research areas include spatial cognition, novel information visualization, and task management for information workers. Mary is active in the HCI research community, and has organized and participated in many workshops.

#### **Roland Fernandez**

Roland Fernandez is a Research Developer working in the Human-Centered Computing (HCC) group of Microsoft Research. His current interests include new models of programming, advanced user assistance, and cognitive modeling. Before joining MSR, Roland worked as a senior developer at Microsoft and other companies in the areas of developer tools, databases, graphics, and operating systems.

#### **Daniel Robbins**

Daniel C. Robbins is a 3D User Interface Designer working at Microsoft Research. His current projects include zooming glanceable mobile applications and end-user geometric modeling for video games. Prior to working at Microsoft, Dan helped develop the pioneering 3D UI work

from the Brown University Computer Graphics Group. Dan's degree is in fine art.

#### **ACKNOWLEDGMENTS**

We thank members of the Windows Live Messenger team for their assistance, Jim Wallace for his initial work on the GroupBanter project, and other members of the VIBE group at Microsoft Research. Additionally, we would like to thank all of our IM buddies who put up with random irrelevant chat sessions from our various personalities.

#### **REFERENCES**

1. Adium. <http://www.adiumx.com/>
2. Facebook. <http://www.facebook.com>
3. Fast Company Staff. Facebook by the numbers. Issue 115, May 2007. Retrieved July 6, 2007, from [http://www.fastcompany.com/magazine/115/open\\_features-hacker-dropout-ceo-facebook-numbers.html](http://www.fastcompany.com/magazine/115/open_features-hacker-dropout-ceo-facebook-numbers.html)
4. Leskovec, J. and Horvitz, E. Worldwide Buzz: Planetary-scale views on an instant-messaging network. Microsoft Research Technical Report, MSR-TR-2006-186, June 2007
5. McGrath, J. E. Methodology Matters: Doing Research in the Behavioral and Social Sciences. In Readings in human-computer interaction : toward the year 2000, edited by Baecker , R.M., Grudin, J., Buxton, W.A.S., and Greenberg, S., Morgan Kaufmann, 1995, pp. 152-169.
6. Windows Live Messenger. <http://get.live.com/messenger/overview>