A Game-Based Methodology for Prototype Evaluation of Collaborative Mobile Applications

Michael Massimi
Dynamic Graphics Project
Department of Computer Science
University of Toronto
Toronto, Ontario, Canada
Email: mikem@dgp.toronto.edu
Phone: 1 (647) 998-2406
Fax: 1 (416) 978-5184
Mailing address:
Graduate Student
Department of Computer Science
10 King's College Rd.
Toronto, ON M5S 3G4
Canada

Craig H. Ganoe
Senior Research Associate
The Pennsylvania State University
College of Information Sciences and Technology
330E IST Building
University Park, PA 16802
phone: 1-814-863-8856
fax: 1-814-865-6426
e-mail: cganoe@ist.psu.edu

John M. Carroll
Edward M. Frymoyer Chair Professor of Information Sciences and Technology
307H IST Building
The Pennsylvania State University
University Park, PA 16802-6823
Email: jmcarroll@ist.psu.edu
Phone: 1.814.863.2476; Fax: 1.814.865.6426

A Game-Based Methodology for Prototype Evaluation of Collaborative Mobile Applications

INTRODUCTION

Mobile computing, perhaps more so than traditional desktop computing, requires methods for allowing application designers to try ideas, create prototypes, and explore the problem space. This need can be met with rapid prototyping. Rapid prototyping is a technique that permits members of a design team to iterate through several versions of their low-level designs (Thompson & Wishbow, 1992). During each cycle of each prototype, the design team identifies critical use cases, verifies requirements are being met, and gathers both subjective and objective data regarding usability. Because "shallow" or low-fidelity prototypes can be quickly created, used, and thrown away (Sefelin, Tscheligi, & Giller, 2003), the team can explore many options and designs with less effort than it would take to create "deep" or high-fidelity versions of each prototype (Rudd, Stern, & Isensee, 1996).

Rapid prototyping techniques are especially valuable when the application is intended for a mobile user. This is for three primary reasons. First, the mobile user is likely to be simultaneously attending to a dynamic or unpredictable environment. This environment taxes the user's cognitive abilities. Users must navigate to their destinations, avoiding obstacles and responding to changing conditions. Non-technical aspects can change, like weather or available routes. Many times, the user must "make place" in order to use the system, stopping to seek out an area to use the software (Kristoffersen & Ljunberg, 1999). Technical aspects of the system, such as network availability and power levels, can also be difficult to accurately predict and may require complex adaptation algorithms (Noble et al., 1997; de Lara, Kumar, Wallach, & Zwaenepoel, 2003; Welch, 1995). Compared to a stationary environment, the number of things that can go wrong seems to skyrocket.

Second, interpersonal communication changes when a dimension of mobility is introduced. When working collaboratively on a task, users require awareness of the tasks their collaborators are performing (Ganoe et al., 2003) in order to prevent redundancy and achieve an equitable distribution of work. When users are mobile, however, awareness is no longer simply *what* other people are doing, but also *where* they are doing it. This introduces a need for additional application support for mobile collaborative systems.

Third, heterogeneity of devices results in different interaction styles. Mobile phones provide an excellent example of this problem. Each manufacturer repositions buttons based on hardware and space constraints. Even within a manufacturer's own product line, multiple key configurations occur. This is to say nothing of the variety of mobile devices available – PDAs, tablet PCs, wearable computers, and so on. Some of the large manufacturers, like Palm, provide human interface guidelines to third party developers (Ostrem, 2003). Most do not.

In terms of evaluating systems, Abowd and Mynatt (2000) argue that our current methods are not sufficient. The traditional task-based evaluation methods no longer apply in a world where we cannot always experimentally control the environment and there is not a clear, single indicator of task performance. There are not established tests that can be performed to determine the effectiveness of deployed systems, mainly because there are not many of them in the world yet. Because we do not have a base of knowledge regarding how to design for mobile interaction, early affirmations of whether the application will serve a human need are critical, and Abowd and Mynatt state we should "understand how a new system is used by its intended population before performing more quantitative studies on its impact" (p. 47).

Mobile systems need fast, inexpensive ways of prototyping and gathering usability results. This entry describes previous work in rapid prototyping for mobile systems. We then contribute a novel rapid prototyping methodology for mobile systems, which we call "Scavenger Hunt." It is anticipated that this methodology will be useful not only for those interested in rapid prototyping and design methodologies, but also for design teams with real deadlines to meet. Finally, we identify future trends in prototype evaluation of mobile systems.

BACKGROUND

Games

Our prototype evaluation methodology is based on a game – specifically, a Scavenger Hunt. The basis for this choice stems from success with using games as a tool for design and testing for non-mobile applications.

Twidale and Marty (2005) used a "game show" format during a conference, wherein contestants found usability problems in software, cheered on by an audience. They argue that "it is worth exploring the power of rapid, lightweight methods to catch relatively uncontroversial and easily fixed usability flaws." SH does this as well, although the focus of the participant is not on the actual discovery of the flaw, but on completing a higher-level task.

Spool, Snyder, Ballman and Schroeder (1994) created a game where designers are placed onto teams and are given a time limit to create a UI. Then, test users move from design to design and must complete the same task on each one. The design with the quickest task completion time is the winner. Here, the goal is to teach designers how to create usable software by rewarding them in a game. In this study, the game is used educationally. The goal of the game is to teach the player how to create good designs, or how to use a particular evaluation method (e.g., heuristic evaluation). Instead, we use a game itself to *evaluate* the prototype. This game-based evaluation is designed to compliment other lightweight usability evaluation metrics like heuristic evaluations (Nielsen & Molich, 1994).

Pedersen and Buur (2000) created a board game to help participatory design teams conceptualize their sessions. The board, modeled after the industrial plant where the users worked, was populated with foam pieces representing artifacts and people. The design partners took turns moving the pieces to explain processes in the plant, and this opened the door to discussion about what should and should not occur during a particular process. The notion of turn-taking is especially noteworthy, as it allows design partners to offer their thoughts and obtain equal footing in the design process. We move from a board game to a "real-life" game in the SH process. In addition, we are interested in using a game as an evaluation tool rather than a design tool. Despite these differences, the past successes with games as parts of the design lifecycle are very encouraging.

Mobile Design & Usability

In experiments conducted by Virzi, Sokolov, and Karis (1996), it was found that testing with low-fidelity prototypes found almost as many usability problems as their high-fidelity counterparts. We argue, however, that paper prototypes will not be suitable for mobile interaction, and that low-fidelity computer-based versions of prototypes should be used instead.

SCAVENGER HUNT

Motivation

To gather usability metrics about mobile collaboration systems, we have developed a methodology we call "Scavenger Hunt" (SH). SH emulates the children's game where players are given a list of items that they must collect and bring to a pre-ordained location. In our methodology, the "players" are in fact target users, and each is equipped with the appropriate mobile device and prototype software under scrutiny.

By basing the rapid prototyping technique on a well-known game, the users can quickly be brought up to speed on how to complete the usability test. Further, they are motivated to "win" the game by completing all the tasks to the best of their ability. This combats the ennui that might otherwise set in when a user is simply asked to perform a series of artificial tasks. In fact, a savvy usability tester might pit two teams against one another to see who wins first, and by what methods. Extreme use cases are more likely to emerge when users push the system to its boundaries to win.

Study Details

We conducted a pilot study wherein we used the SH method to evaluate a collaboration tool prototype. The specific details we have used to conduct this SH session follow and are meant to serve as an early model for future applications of this method. These details and parameters can, of course, be tailored to meet the needs of a particular design team, product, or schedule.

Software

In order to pilot the Scavenger Hunt method, we developed a weblog prototype as the software under scrutiny. The weblog (which we call SH Blog) allowed multiple people to add posts to it, edit each others posts, and reorganize the ordering of the posts. We purposefully did not create a "polished" version of the software. The prototype was representative of a first pass through coding the system, and was written in approximately 5 hours.

The prototype was written in PHP and HTML. Clients ran Microsoft Internet Explorer for Pocket PC and rendered pages from an Apache web server running on Linux. Data was stored server-side in a MySQL relational database.

Participants

Eleven participants were recruited from a summer school program for gifted youth. They were divided into groups of three (one participant failed to arrive). Each team was self-selected and worked together on a project during the summer school program, so the participants were comfortable working with each other. Overall they reported high levels of comfort with technology, but did not use mobile computers very often.

Pre-Session Setup

Before the SH session, we distributed 24 clues throughout the building. These clues were evenly divided amongst the three floors of the building. All clues were printed on brightly-colored paper and were hung on walls or placed on tables. We attempted to disperse the clues throughout the building evenly so that a participant would have a chance to find a clue in consistent time intervals (e.g., after about 45 seconds of walking). We ensured that all clues were

in public areas so that participants would have access to them, and would not feel awkward entering private offices.

The SH Blog was engineered to capture data about user interactions before the session. The time of posting and user who posted were logged. A software engineer monitored the MySQL database that stored SH blog posts, and noted the progress of the team. This monitoring was essential to the evaluation of the prototype from the software engineering perspective, as it allowed us to look "under the hood" of the software during the session.

Finally, we ensured IEEE 802.11b wireless networking was available in all areas where there were clues. Some areas, such as stairwells or elevators, could not receive a signal; this is characteristic of most mobile computing environments, however.

Starting a SH Session

We gathered each team individually at the beginning of the session and had them complete a questionnaire that asked questions about their comfort level with mobile computers, their experience with working while mobile, and their preferences for group work. Each team was then given an overview of the game that explained the following:

- There are clues throughout the building. They are all in plain view and are in public spaces.
- You need to collect as many clues as you can in order to answer a riddle.
- You must use the SH Blog to share the clues and to work on solving the riddle. You may not use software on your mobile computer besides the SH Blog.
- You will have 1 hour to complete the task and return here with the answer to the riddle.

Participants were then given the riddle and asked to begin. They immediately began the task and started to walk around the building, entering clues into the SH Blog.

Collecting Data During a SH Session

During the study, participants were video-taped by researchers with camcorders in order to later analyze comments and note salient themes. Participants were asked to think aloud in order to capture the cognition accompanying the interaction and problem solving. At the end of the session, users completed a questionnaire about their experience with the SH Blog prototype. Finally, based on observations during the task and questionnaire responses, we conducted a brief semi-structured interview wherein we asked questions about problems, ideas for changes, and experiential preferences. By using three different research instruments, we are able to collect a wide range of data and make design suggestions based on both explicit and implicit behaviors.

At the system level, we captured information about the number of posts made by each team member, the total number of posts, the movement of posts, the deletion of posts, and so on. By charting these over time and comparing the different groups, we can note differences in usage and system support. For example, one group in our study generated a new post for each clue they discovered; another chose to accumulate all clues in a single post. These varying styles indicate, for example, the need for the SH Blog to accommodate both large numbers of posts and large, monolithic posts.

Evaluation and Results

Some of the salient results of our trial run are presented. It is important to note the different types of problems that the method identified – they run the gamut from usability issues, to systems issues, to social issues. Many of these insights may not have been found by traditional task-based user tests.

We noted that no group collected the entire set of clues. This was for a variety of reasons. A team member might believe that a different team member already collected the clue. A team might miss the clue completely. A team might have found it and subsequently deleted it, reasoning that it was redundant or useless. Without a sufficient subset of the clues collected, teams could not solve the problem.

Different teams approached the game differently, even though they were all given the same starting conditions. One team chose to divide the building into floors and then assigned a floor to each member. Another team chose to have one member act as an analyst back at the "base," while the other two members walked around and focused solely on the collection aspect. This indicates that high-level "gaming" strategies must be supported in the software.

In every trial, the members initially split up and went separate ways to find clues. Again, in every trial, the members eventually met face-to-face once they thought they had collected all of the clues. We noticed a shift in work styles from an individual, mobile worker to a stationary, group worker. This indicates that the software must accommodate individual work and group work separately, and provide a transition between the two work styles.

Of the four trials, only one team actually solved the problem. Even this team had boiled it down to an educated guess. Based on this outcome and the ones given above, we determined that the SH Blog did not allow the users to accomplish the task and needed revisions. The ability to reorder posts more easily and the ability to draw free-form tables were the primary revisions that users identified in the questionnaires and interviews. In one exceptional case, a user accidentally deleted the aggregate post that contained all of their collected clues! It was only then that we realized there was no undo function.

As these four themes suggest, users identified numerous flaws and areas for improvement in the software during interviews, questionnaires, and observations. Because the goal of this study was for feasibility purposes, we have not evaluated our method against other methods on similar tasks. We do, however, feel that the insights gained from using SH were well worth the setup costs. The major contribution of using SH is to show that rapid, low-cost usability & systems testing can be conducted early in the design process. This method may be of use to design teams in both research and industry settings.

FUTURE TRENDS

As mobile applications are developed more frequently in order to suit the needs of the third-wave information worker, we believe that prototyping, iterative design, and usability testing will become more and more important. Cell phones demonstrate that users prefer mobile devices when they are used for interpersonal contact, and methods for evaluating collaboration on-the-go are essential for this task. Techniques like SH are useful for software engineers and usability engineers alike. We believe more tools like it should be developed.

Further, the evaluation of these tools is an open research problem. How do we demonstrate that one mobile system suits its users' requirements more effectively than another? What are the outcomes to be measured? In the absence of a long history of software deployment and use, these questions remain to be answered.

CONCLUSION

As Abowd and Mynatt (2000) observed, it is extremely difficult to conduct evaluations of mobile computing systems because of the always-on, dynamic environment. For this reason, it is imperative that we have tools for early, non-trivial user testing, and we have presented a novel method for doing so. Our method is lightweight and can be applied repeatedly in the design process to ensure that requirements are met before an expensive deployment begins. Although it does not replace actual field trials, it can identify systems-level and interface-level flaws by simulating a representative task in the problem domain. Continued work in identifying the critical components of evaluating mobile systems is important, as is the need for early prototyping and validation.

REFERENCES

Abowd, G. & Mynatt, E. (2000). Charting past, present, and future research in ubiquitous computing. *ACM Transactions on Computer-Human Interaction,* 7(1), 29-58.

de Lara, E., Kumar, R., Wallach, D. S., & Zwaenepoel, W. (2003). Collaboration and multimedia authoring on mobile devices. *Proceedings of the 1st International Conference on Mobile Systems, Applications, and Services (MobiSyS),* 287-301.

Ganoe, C., Somervell, J., Neale, D., Isenhour, P., Carroll, J., Rosson, M. B., et al. (2003). Classroom BRIDGE: using collaborative public and desktop timelines to support activity awareness. *Proceedings of the 16th ACM Symposium on User Interface Software and Technology (UIST),* 21-30.

Kristoffersen, S. & Ljunberg, F. (1999). Making place to make it work: Empirical exploration of HCI for mobile CSCW. *Proceedings of the International ACM SIGGROUP Conference on Supporting Group Work 1999,* 276-285.

Nielsen, J. & Molich, R. (1990). Heuristic evaluation of user interfaces. *Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems 1990*, 373-80.

Noble, B., Satyanarayanan, M., Narayanan, D., Tilton, J. E., Flinn, J., & Walker, K. (1997). Agile application-aware adaptation for mobility. *Proceedings of the 16th ACM Symposium on Operating Systems Principles,* 276-287.

Ostrem, J. (2003). Palm OS user interface guidelines. Retrieved January 6, 2006, from http://www.palmos.com/dev/support/docs/ui/UIGuidelinesTOC.html.

Rudd, J., Stern, K. & Isensee, S. (1996). Low vs. high-fidelity prototyping debate. *ACM Interactions,* 3(1), 76-85.

Pedersen, J. & Buur, J. (2000). Games and movies – towards innovative co-design with users. *CoDesigning Conference,* Coventry, UK.

Sefelin, R., Tscheligi, M., & Giller, V. (2003). Paper prototyping – What is it good for? A comparison of paper- and computer-based low-fidelity prototyping. *CHI 2003 Extended Abstracts,* 778-779.

Spool, J. M., Snyder, C., Ballman, D., & Schroeder, W. (1994). Using a game to teach a design process. *Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems,* 117-118.

Thompson, M. & Wishbow, N. (1992). Prototyping: Tools and techniques: Improving software and documentation quality through rapid prototyping. *Proceedings of the 10th Annual International Conference on Systems Documentation,* 191-199.

Twidale, M. & Marty, P. (2005). Come on down! A game show approach to illustration usability evaluation methods. *IEEE interactions,* 12(6), 24-27.

Virzi, R. A., Sokolov, J. L., & Karis, D. (1996). Usability problem idenitification using both low- and high-fidelity prototypes. *Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems,* 236-243.

Welch, G. (1995). A survey of power management techniques in mobile computing operating systems. *ACM SIGOPS Operating Systems Review,* 29(4), 47-56.

TERMS

1. Scavenger hunt – a lightweight method for evaluating prototypes early in the design of a mobile or ubiquitous computing system wherein participants play a game while the design team identifies systems- and interface-level flaws.
2. Rapid prototyping – the process of creating, evaluating, and refining low-cost, easily fabricated prototypes in order to quickly identify and fix flaws
3. SH Blog – a collaborative mobile weblog that is shared amongst a group of people working on the same task. The people who are involved are also posting from mobile devices.
4. Evaluation methodology – a procedure for determining the quality of a system in relation to how it satisfies user needs. A Scavenger Hunt is an example of an evaluation methodology for rapidly-prototyped mobile collaboration systems.
5. Mobile collaboration – the situation that arises when two or more people must work together on a problem while one or more of them is in the process of changing location or is in the field.
6. Extreme use case – unlike a “critical use case” where a task is identified that is essential to the operation of the system, an “extreme use case” is the situation that arises when users interact with the software under stressful (i.e., timed) conditions and push the software to its limits, both in terms of system-level support and usability.
7. Participatory design – the process of designing *with* users instead of designing *for* users, by actually including end-users on the design team and mutually learning from one another.