# Momento: Early-Stage Prototyping and Evaluation for Mobile Applications (in submission)

*Scott Carter*
Electrical Engineering and Computer Science Department
University of California at Berkeley
Berkeley, CA 94720-1770
sacarter@cs.berkeley.edu

*Jennifer Mankoff*
Human-Computer Interaction Institute
Carnegie Mellon University
Pittsburgh, PA 15213
jmankoff@cs.cmu.edu

## ABSTRACT

While it is important that applications designed for mobile devices be iteratively tested in natural contexts (*e.g.*, by participants in the field), it is especially difficult both to develop mobile applications and to deploy them in the field. We built a system, MObile Messaging and EvalutioN TOol (Momento), to facilitate early stage development and *field testing* of mobile applications. Anyone who regularly uses a mobile device that supports text or media messaging can be a participant in studies using Momento, without having to install any software. Using our tool, developers can simulate and study applications in the field for a wide array of mobile applications *without any device-specific development*. Thus, Momento can support *early-stage, medium to large-scale Wizard-of-Oz field studies* of participants *using their own devices.*

Developers can simulate ("Wizard-of-Oz") an application by sending SMS or MMS (multimedia content) to participants, or responding to participant queries for information (such as "Where is the nearest restaurant?"). Additionally, Momento can be used for experience sampling, diary and probe studies; and logging of "wild" SMS or MMS (multimedia messaging) communication. In addition to detailing the implementation and functionality of Momento, we report on user tests verifying its usefulness for early stage Wizard-of-Oz simulation during mobile application development.

**ACM Classification** H5.2 [Information interfaces and presentation]: User Interfaces. - Graphical user interfaces.

**General Terms** Design, Human Factors, Experimentation

**KEYWORDS:** Mobile, Wizard-of-Oz, rapid prototyping, ESM

## INTRODUCTION

The ability to test mobile applications *in the field* is crucial to developers: True understanding of the value of a mobile application depends on situated use data gathered in natural contexts. Unfortunately, the variability of mobile platforms such as PDAs and cellphones often force developers into device-centric development. As a result, user studies often require participants to carry or switch to an unfamiliar specialty device, rather than using their own preferred hardware. This makes it difficult to run large scale studies because it is hard to find participants, decreases the size of most deployments (since hardware must be provided to each participant), and requires longer time scales, so use of the new, unfamiliar device can stabilize.

Interviews we conducted with three developers of mobile systems included many mentions of these barriers to iterative design. In addition, they highlighted three key resulting difficulties for evaluating mobile systems in the field. The developers most often cited frustration with data sparsity. In particular, they mentioned that the highly situated nature of mobile applications meant that use was irregular and fleeting, mitigating their ability to gather enough data to iteratively develop prototypes. In effect, they felt even when they developed a solution robust enough for field use, the limited *scale of deployment* possible meant that they received only minimal feedback of actual use. To a lesser extent, developers also felt that the lack of familiarity with devices made those devices less *unobtrusive*. This reduced the validity of the resulting data, because participants were not habituated to the devices. Finally, they expressed the importance of *rapidly iterating* high fidelity interfaces based on participants' experiences with them in field experiments. Even once developers chose a particular platform for experimentation, rapid iteration was difficult because of a lack of rapid prototyping tools that could be used for field deployments of mobile applications.

We have created a tool, MObile Messaging and EvalutioN TOol (Momento), that addressess each of the issues raised above. Momento allows developers to make use of devices participants already own. Because of this, our tool can potentially increase the *scale* of studies, as well as helping them remain *unobtrusive* by leveraging existing participant devices. Additionally, because it requires no participant-side installations, no specialty hardware, and can support Wizard-of-Oz (WOz) studies of simulated, unimplemented systems, Momento can enable *rapid iteration* of both low and high fidelity interfaces.

Momento is designed to use technologies that participants are likely to have on their own mobile devices: SMS, which allows people to send short messages between mobile devices, and MMS, which allows people to send multimedia content between mobile devices. Momento provides developers with a desktop application that allows them to send messages to mobile devices and to receive messages from mobile devices. The application also provides facilities for visualizing generated and received events per participant. Using this interface, developers can simulate application behavior in the field for a wide array of mobile applications without any device-specific development. Applications may be tested either at a low fidelity, using text messages sent *via* SMS, or at higher fidelities, using images sent via MMS.

In addition to its support for Wizard-of-Oz style field studies, Momento supports a host of other valuable field techniques. Momento can be used for experience sampling, diary studies, and probe studies. It can also be used to study SMS and MMS use in the field (automatically logging communication).

In summary, it is important that mobile applications be evaluated in the field, but it is difficult to develop cross-device applications for early-stage field testing. With a tool such as Momento, developers can rapidly test application ideas and interfaces, easily recruit participants and reduce deployment costs by making use of devices participants already own, and easily survey participants in the field. Additionally, they can do all of this with devices that participants have already situated into everyday use.

## USE OF Momento

The basis of Momento is the ability for wizards to send and receive mobile messaging events to experiment participants from a desktop application. The application also provides facilities to organize, manage, and export messages for later analysis.

### Example use: WOzing a restaurant finding service

The WOz method involves simulating application behavior with the help of a human "Wizard," usually unbeknownst to the user of the application. The wizard usually simulates aspects of the application that are particularly difficult to build, and other components of the application may actually be implemented. For example, developers of speech-based interfaces have used WOz for human voice recognition in early stage prototypes [10]. In the mobile domain, Li *et al.* used WOz to substitute for location discovery [11]. A developer can use Momento to simulate both an application's back end and user interface. Alternately, a developer could implement parts of the application on a desktop computer, and use Momento to transmit the resulting data or user interface to a participant's mobile device.

For example, suppose a developer wants to test an application for finding the restaurant nearest a participant. The developer might prototype a application that, given an address, shows a map with nearby restaurants. `http://maps.google.com`, among other services, can be used to quickly generate an image showing restaurants near an address. The wizard can then grab this image and send it, using our tool, *via* MMS

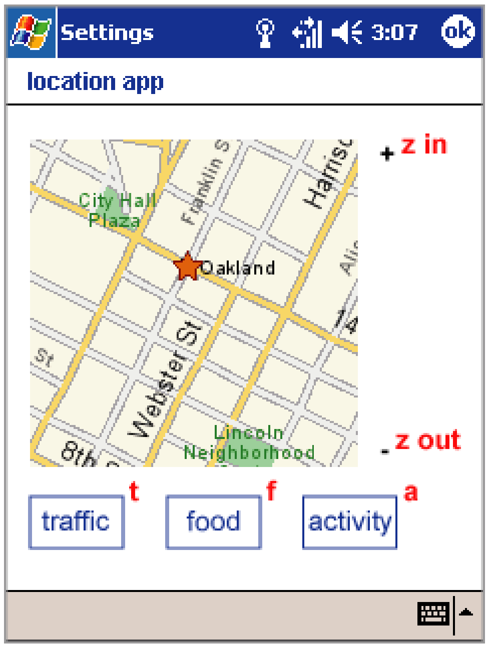


Figure 1: An example interface sent to a participant *via* MMS with tooltips in red. The participant can SMS the tooltip back to the developer/wizard to affect the associated interaction.

(or simply capture the textual list of restaurants and send it *via* SMS) to the participant.

Participants can use predefined codes to rapidly ask questions of the wizard. For example, in an evaluation of a restaurant-finding service, the text 'Q1' in "Q1 Fifth Ave. and Monroe St," might be a macro for the question "Where is the nearest restaurant..." Alternately, text might have a procedural meaning. For example, the text "PQ" could be used to refer to the "previous question."

If a participant needs to interact directly with a picture of an interface sent *via* MMS, this can be facilitated by displaying tooltips near interactive components (Figure 1). A participant can send a message containing the tooltip for the specific widget they wish to interact with. The developer/wizard can then respond with the next appropriate image.

For SMS-based WOz systems, no implementation is required. The developer can draw on any number of sources for information to give in responses. For MMS-based WOz systems, some implementation may be required. The developer must either use pre-created images, or have a program from which images can be captured. The job of implementing a desktop program with an interface appropriate for a mobile device is much easier than that of implementing an actual application to run on the mobile device. The wizard simply interacts with the desktop application, and captures screen images to send to the remote participants. In either case, to run large-scale studies, a developer must simply recruit enough wizards to manage the volume of incoming requests over the time specified for the study.

*Simulating sensor-based applications* While support for simple one-on-one WOz between a wizard and a participant is a major step forward for mobile application developers, many mobile applications also depend on the ability to share information among different mobile users. For example, a messaging application may depend on information about the interruptibility of message recipients. Or a meeting application may depend on information about the location of group members. Momento can also be used to WOz interfaces of this sort.

A developer can use Momento to sample participants periodically, asking them about things such as location or interruptibility (*ESM, or Experience Sampling Method*). Alternatively, a developer can simply ask participants to report a change in state. This data can then be shared with other participants at appropriate times.

While the reader may be concerned about how onerous this method might be for participants, our study (described below) showed that, at least during a single day, participants were accepting of the use of ESM for data gathering. Additionally, the method has been used successfully in the past. For example, in Benford *et al.*'s game "Uncle Roy All Around You" participant's self reported location information that was fed back into the application [1]. While in this case position is integrated into a complete application rather than a WOz study, the application showed that participant self reports could be useful when no infrastructure was in place to sense some important piece of data.

## The Wizard's interface

The Momento desktop application provides facilities to organize, manage, and export for analysis mobile messages (Figure 2). The Momento interface is implemented in Java using standard libraries. To use Momento, a wizard first creates or opens a new project. The project is a directory that will contain all participant and messaging data for a particular study (Momento saves data in tab-separated format for easy export to analysis programs). Once the wizard has specified a project, she can add participants using the "Participant" panel or schedule message events using the "Generated Event" panel. Using the "Receive Event" panel the wizard can view the content of incoming messages. A timeline provides an interactive visualization of when each event was sent (squares) or received (circles) per each participant. Finally, a text panel shows a log of every processed event as well as feedback of wizard actions.

*Configuring Participants* Because every event in Momento is tied to a particular participant, each project must have at least one participant. To create a participant, the wizard selects the first entry in the "Phone" combobox and types the participant's mobile phone number in its place. An assumption is made that this participant's mobile phone has at least SMS capabilities, but the wizard can select whether the phone supports MMS: if the wizard does not specify that the device supports MMS, Momento disallows file attachment to messages. The wizard may then also add a name for the participants and add general notes about that participant.

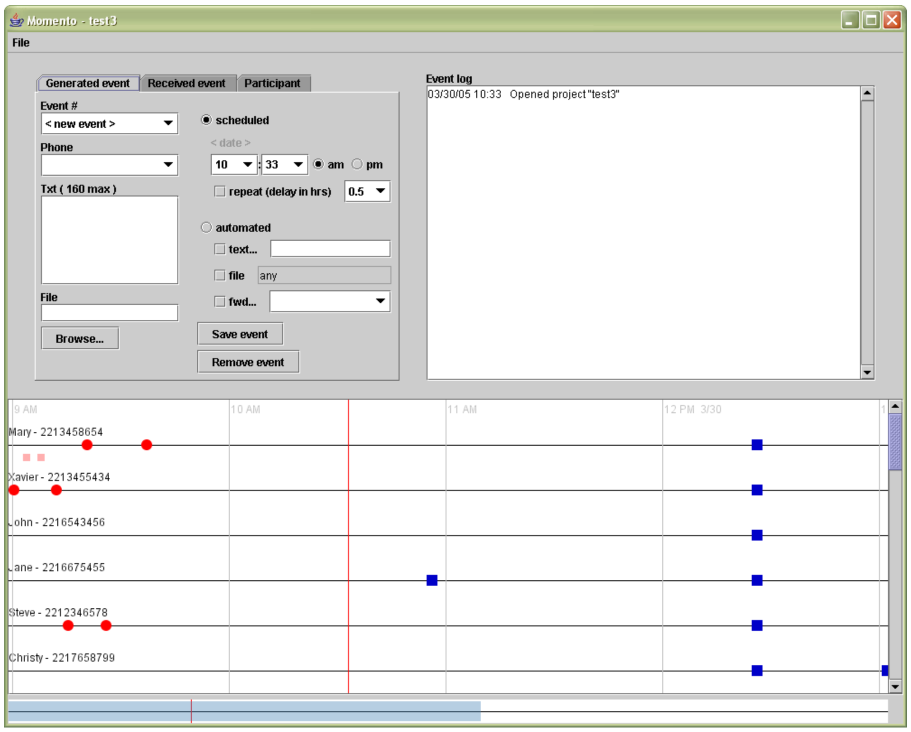
*Configuring Generated Events* A wizard can configure any number of messaging events to be sent to one participant or all participants in the project using the "Generated Event" tab (Figure 2 (b)). For every event, the wizard specifies the participant to whom the event will be sent, the message to be sent, and an optional file to be sent via MMS. Messages can be sent once-only, periodically (scheduled), or can be triggered with some participant input (automated). In the bottom half of Figure 2 (b), the wizard specified that the message "What activity are you doing now?" should be sent to all participants every 1/2 hour starting at 7:40pm

Wizards can specify a variety of triggers for automated messages. A message can be triggered by each arrival of a message from the participant; only when the text of an arriving message matches a particular regular expression; or only when the participant transmits a file (MMS). Wizards can also specify that an automated message forward another message to a participant, by checking the "fwd..." box. For example, in the top half of Figure 2 (b), the wizard specified that if participant (221) 345-8654 sends a message with the prefix 'p1,' the message "Forwarding..." will be sent to the participant, and the original message from the participant will be forwarded to the (221) 654-3456.

*Viewing Received Events* When Momento receives a new message event from a participant, it is visualized in the timeline view (described below). To subtly catch the wizard's eye, the last event received appears in a slightly brighter color than the other events. A wizard can view the details of received events in the received event panel (Figure 3). The panel shows the event number Momento assigned to it, the participant phone number and name with which it is associated, the date and time it arrived, and the text and file content of the message.

*Timeline View* The timeline view presents a graphical interface to message events (Figure 2(a), bottom). Each participant has a corresponding timeline on which are shown received and scheduled message events. Received message events are shown on a timeline as red circles; events scheduled to be generated as blue squares; events generated in the past as gray squares. Additionally, pink squares are used to indicate automated events that can be triggered, but are not scheduled at a particular time. Clicking on any of the message event icons opens the information for that event in its corresponding panel (*e.g.*, clicking on an automated event would open that event's information in the "Generated Events" panel). A light gray grid in the background marks the hours of the day, and a red line advances across the view indicating the current time (the "now" line). Finally, we implemented a slider bar beneath the timeline view that displays the current location of the now line. This is important because the wizard may scroll the timeline view such that the now line no longer appears. The slider allows the wizard to relate the data displayed in the timeline view with the current time regardless of the current time window displayed.

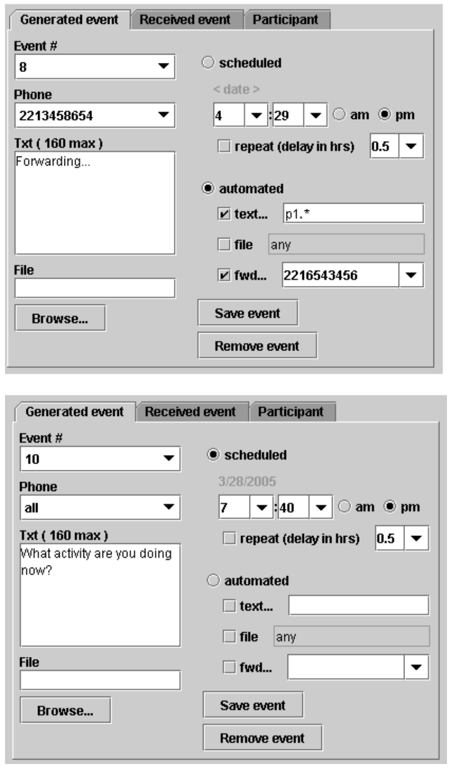
*Mobile interaction* Different mobile phones use different strategies for notification and visualization of messages that participants receive from Momento. In many cases, participants will have configured the phone to match their own pref-



Event # < new event >
Phone
Txt ( 160 max )
File
[✓ scheduled]
< date >
10 : 33 [✓ am] pm
repeat (delay in hrs) 0.5
automated
text...
file any
fwd...

Event log
03/30/05 10:33 Opened project "test3"

(a)

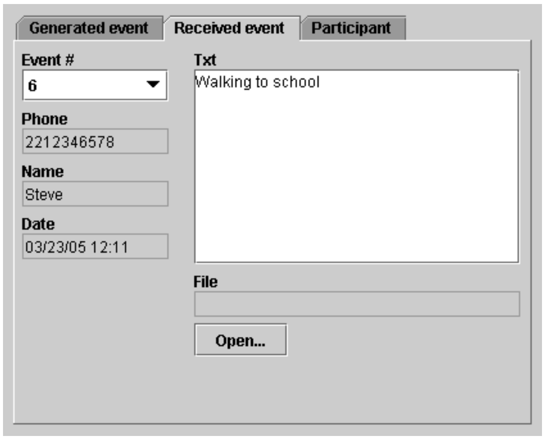


Event # 8
Phone 2213458654
Txt ( 160 max ) Forwarding...
File
scheduled
< date >
4 : 29 am [✓ pm]
repeat (delay in hrs) 0.5
[✓ automated]
[✓ text...] p1.*
file any
[✓ fwd...] 2216543456

Event # 10
Phone all
Txt ( 160 max ) What activity are you doing now?
File
[✓ scheduled]
3/28/2005
7 : 40 am [✓ pm]
repeat (delay in hrs) 0.5
automated
text...
file any
fwd...

(b)

Figure 2: The Momento user interface. **(a)** Main window, showing past (red) and scheduled (blue) messages. Squares indicate events that will/have been generated (messages from the wizard), while circles indicate recieved messages (messages from the participants). **(b)** Detail view of interface for generating a message. (top) An automated message triggered by a regular expression match. (bottom) An automated message that is sent every 1/2 hour.



Event #: 6
Phone: 2212346578
Name: Steve
Date: 03/23/05 12:11
Txt: Walking to school
File:

Figure 3: Received event

Figure 4: A wizard monitoring Momento while focusing on another task.

erences for message event handling. Since Momento uses a standard SIM card to send messages, participants can configure their mobile phone to handle incoming messages in a unique way. For example, a participant could configure their phone to use a specific ring tone when receiving messages from Momento.

## Implementation

Our system uses an SMS/MMS gateway, NowSMS, and a MultiTech GSM/GPRS modem to connect the desktop application to the provider network associated with a currently active SIM card (inserted into the modem). The gateway is a desktop server application that connects directly to the modem and processes received and generated events. The modem has all of the functionality of a mobile phone and connects directly to the provider network associated with the SIM card. The provider network can then forward the message to any participant phone on any network (just as if it had been sent from a cell phone).

## EVALUATION

We ran a pilot study to evaluate the user interface and give us insights into how Momento affects both the developer and participant experience. Thus, we will refer to the subjects in our study as *wizard* or *participant* depending on the role they played in our study.

We asked *wizards* to run a WOz study focusing on two types of information: location and traffic conditions. We provided the the wizards with participants and a pre-defined protocol for the study. Wizards were instructed to use Momento to query participants about their location *via* ESM. Each *participant* was told to query the wizard for location data about other participants (*e.g.*, "Where is Steve?" or "Who is in the lab?", *etc.*) or traffic information (*e.g.*, "How is the traffic on the bridge?").

Because this was the first time we observed this technique in use we also wanted to give participants and wizards a chance to brainstorm actively about other possible uses for the system. To encourage this open-ended use, we instructed wizards that they could come up with other experience sampling questions and participants that they could query the system for any information they need throughout the day.

To run the study, we recruited total of four subjects with at least an upper-level undergraduate understanding of user studies. For one working day, two of the subjects played the role of participants, while two switched off between being participants and wizards. Momento is designed to be used as much as possible in the *context* of other work. Thus, wizards were asked to use Momento from their personal desktop machines (Figure 4). The job of wizard was traded off during the study, with each wizard's turn lasting at most 2 hours, followed by a break of 1 to 2 hours.

## Results

We found that the system was intuitive to use and were inspired by the range of uses for the system in the open-ended section of the study. Overall, during the one day study, wizards generated 55 messages and participants sent 58 messages. However, only a small number of those were related to the main questions of the study. There were four traffic requests from participants and four responses to those requests. Also, there were five location requests and five responses to those requests and total of seven questions generated querying participants' location (five scheduled events and two automated events).

The rest of the messages were related to uses of the system that wizards innovated. For example, one wizard setup questions that participants could respond to with scalar results. One of these questions was related to a message that occurred during the course of the study:

> Is the current [lab] tour distracting your work? Please reply on a scale of 1 for not distracting, 5 for impossible to work

Other questions were more general, such as:

> How busy are you right now? 1 not busy at all, 5 is really busy.

Also, wizards used the system to compensate for their sometimes not being able to gather information as rapidly as an interactive system might. For example, the wizard used the automatic reply feature in Momento to respond with the following text whenever a participant sent a request with text that matched the regular expression ".*traffic.*":

> I'll check the traffic for you as soon as I can and get back to you

There were also more playful uses of the system. For example, one wizard configured the following two messages: one a question and one a response if a participant sent the correct answer.

> What is the airspeed of swallow?

> African, I mean American, aaahhhhhhh.

One wizard extended the idea of prototyping games using Momento and created a short text adventure game during the study. The first message the wizard sent was:

> You're in a forest. There is a trail to the N and a cabin to the W. Which way would you like to go? N/W

The wizard then set up two automated messages that would fire based on participant responses. If a participant sent back a string that matched the regular expression "W.*", Momento responded with:

> You open the cabin but there was a troll and he squashed you with his fist. Game over. Bad luck

If a participant sent back a string that matched the regular expression "N.*", Momento responded with:

> You followed the trail and there was a big pot of gold and a beautiful maiden. Congratulations, you win!

*Interviews* In interviews, wizards reported that using Momento was not too distracting and "felt like e-mail or instant messaging." One wizard suggested modeling the interface after instant messaging, adding clickable "faces" for each participant. Also, there were a set of features that both wizards thought should be added to the interface. The most pressing need reported was a means of visually linking questions and answers. Currently, the timeline visualization shows generated and received messages but does not indicate the relationship between messages (*e.g.*, whether a message was a response to a previous message, or which message it was a response to). A similar issue was that wizards wanted to schedule a message to repeat only if a question had not been answered. Also, both wizards wanted the ability to create groups of participants. While Momento supports sending a message to all participants, wizards wanted to send messages only to subsets of participants. Wizards also thought there should be a "Send now" button on generated messages. Finally, wizards sometimes had difficulty selecting overlapping message icons in the timeline visualization.

### Discussion

While our evaluation was small and made use of only a subset of Momento's capabilities (*e.g.*, we did not use MMS), the interface seems to fill a need for mobile application developers. Our experiment showed that the system works well for standard applications, such as traffic and location information, but also that there is a potentially large set of additional applications that lend themselves to prototyping using Momento (*e.g.*, games). The success of the open-ended section of our study also suggests that it may be valuable for developers to take advantage of "active brainstorming" using Momento to generate ideas for applications.

## BROADER APPLICATIONS

Momento supports a variety of techniques to support rapid prototyping of mobile applications, requirements gathering, and studies of mobile messaging. In addition to it's Wizard-of-Oz support, researcher and developers can also use Momento to support formative methods such as experience-sampling, diary, and probe studies; and to study the use of SMS and MMS through logging. Below, we describe how Momento can support each technique, and illustrate the benefits of Momento by describing how that technique was executed in the past.

### Formative requirements gathering with ESM, diary, and probe studies

We described how Wizard-of-Oz studies run with Momento can use ESM to provide wizards with richer data when crafting responses. But developers can also use Momento to support any variety of ESM-, diary study-, or probe-based formative study to gather requirements for applications. In ESM, developers query participants about their activities either periodically or when an event occurs. In the diary study and probe methods, participants write about activities important to them as they happen, and may capture other media such as images or sounds [2]. Using Momento, developers can rapidly create ESM and diary studies that make use of technologies that participants already use regularly.

*Benefits of Momento over past approaches* While other developers have used SMS and MMS in ESM and diary studies [7, 5], they have handled messaging in an *ad hoc* fashion [8]. Developers have also conducted ESM studies using tools such as pagers or phones [3] or developed custom tools for ESM [9]. The participant experience is not significantly different from this past work when using Momento. However, Momento streamlines the experience of developers, allowing them to organize, save, and automate message events. For example, developers can use scheduled generated events for periodic ESM queries, automatic generated events for event-based ESM queries, and records of received entries for diary and probe studies. Additionally, because all message configuration is done on a desktop machine accessible to the developer (as opposed to on the participant's device [3]), questions can be modified or added at any time. Also, studies using Momento have the capability of being more scalable because they do not rely upon a particular operating system. Finally, Momento can be used to support other related methods, such as probe methods, that require participants to respond to questions with media other than text.

### Logging communication data

To support studies of communication patterns, Momento can act as a gateway between participants communicating *via* SMS or MMS messages. In the SMS case, participants send all messages to the gateway, but preface messages with a participant-specific tag. Developers use the forwarding feature of the automatic generated event that sends the content of a message to another participant's phone. When it receives a message, the gateway logs the message and forwards it on to the appropriate participant. This results not only in a log of communication but also allows the developer to produce visualizations of aggregate data for test interfaces. In the MMS case, because the MMS protocol is more sophisticated, the gateway can be configured to automatically log events, removing the burden of prefacing a message for participants.

*Benefits of Momento over past approaches* While studies of messaging use have thus far relied on paper-based diary study methods, direct observation, interviews, and questionnaires [6, 17], with this approach, messaging events can be captured automatically. While other methods can elicit some data not available to Momento, using Momento makes studies of communication patterns nearly seamless for participants. Also, using Momento developers can begin analyzing data immediately without the extra step of translating data from paper to digital.

## CONCLUSION AND FUTURE WORK

In this paper we presented a tool, Momento, that allows developers to make use of devices participants already own to test mobile interfaces in the field, at extremely early stages of development. Our tool can potentially increase the *scale* of early-stage studies of mobile applications, as well as helping them remain *unobtrusive* by leveraging familiar devices. Additionally, because it requires no participant-side installations and no specialty hardware, Momento can enable *rapid iteration* of both low and high fidelity interfaces.

Other work has focused on supporting early-stage design and testing of mobile devices [11, 12]. However, this work does not support field studies of the scale or unobtrusiveness enabled by Momento. More broadly, various creative WOz studies have been used to test a variety of ubiquitous computing applications in the past [4, 15, 19, 16]. The most commonly used method to rapidly iterate concepts and designs for mobile applications is paper prototyping [13, 18]. While neither Wizard-of-Oz nor paper prototyping require any additional coding, Wizard-of-Oz studies using Momento can be used in the field because participants do not need to remain in the presence of wizards. Our study of Momento validates its ability to support field studies and its value as a tool for exploring new design ideas.

Momento can also be used for ESM and diary studies, and to study current SMS and MMS use in the field (automatically logging communication). As described in the previous section, in both cases, it presents significant advantages over past approaches.

One particular method we are interested in exploring in the future is the use of Momento for sensing information that participants *cannot tell us*. Developers could add contextual cues to an Momento project by configuring third-party applications to send a message to Momento with a specific preamble. For example, a developer evaluating a public peripheral display could setup a third-party Bluetooth discovery application to send found phone numbers to Momento. This message could then trigger Momento to send an automatic message querying the appropriate participant about her use of the information on the display.

There are also many opportunities to improve the desktop system. Improvements to the timeline view we intend to implement include zooming and selection and manipulation of subsets of messages. Also, currently automated events can be triggered by a regular expression or file. In future work we will implement support for more sophisticated triggers that encompass trends across messages and participants. Finally, mobile applications often require interaction with public displays, especially in ubiquitous computing environments [14]. In future work we will implement methods for such applications to interface with the Momento desktop system directly.

## REFERENCES

1. Steve Benford, Will Seager, Martin Flintham, Rob Anastasi, Duncan Rowland, Jan Humble, Danae Stanton, John Bowers, Nick Tandavanitj, Matt Adams, Ju Row-Farr, Amanda Oldroyd, and Jon Sutton. The error of our ways: The experience of self-reported position in a location-based game. In *Ubicomp*, volume 3205 of *Lecture Notes in Computer Science*, pages 70–87, 2004.

2. Scott Carter and Jennifer Mankoff. When participants do the capturing: The role of media in diary studies. In *Proceedings of the Conference on Human Factors in Computing Systems (CHI)*. To appear, 2005.

3. S. Consolvo and M. Walker. Using the experience sampling method to evaluate ubicomp applications. *IEEE Pervasive Computing Mobile and Ubiquitous Systems: The Human Experience*, 2(2):24–31, 2003.

4. Sunny Consolvo, Peter Roessler, and Brett E. Shelton. The carenet display: Lessons learned from an in home evaluation of an ambient display. In *Ubicomp*, volume 3205 of *Lecture Notes in Computer Science*, pages 1–17, 2004.

5. Helen J. Forgasz and Gilah C. Leder. Academics: How do they spend their time? In *Proceedings of AARE Conference*, 2003.

6. R. E. Grinter and M. Eldridge. y do tngrs luv 2 txt msg? In *Proceedings of the Seventh European Conference on Computer- Supported Cooperative Work ECSCW*, pages 219–238, 2001.

7. Robin M. Hogarth. Is confidence in decisions related to feedback? evidence — and lack of evidence —

from random samples of real-world behavior. In Klaus Fiedler and Peter Juslin, editors, *In the beginning there is a sample: Information sampling as a key to understand adaptive cognition*. Cambridge University Press, 2004.

8. Sami Hulkko, Tuuli Mattelmki, Katja Virtanen, and Turkka Keinonen. Mobile probes. In *Proceedings of the Nordic conference on Human-computer interaction*, pages 43–51, 2004.

9. Stephen S. Intille, Emmanuel Munguia Tapia, John Rondoni, Jennifer Beaudin, Chuck Kukla, Sitij Agarwal, Ling Bao, and Kent Larson. Tools for studying behavior and technology in natural settings. In *Ubicomp*, volume 2864 of *Lecture Notes in Computer Science*, pages 157–174, 2003.

10. Scott R. Klemmer, Anoop K. Sinha, Jack Chen, James A. Landay, Nadeem Aboobaker, and Annie Wang. A wizard of oz prototyping tool for speech user interfaces. In *Proceedings of the ACM Symposium on User Interface Software and Technology (UIST)*, pages 1–10. ACM Press, 2004.

11. Yang Li, Jason I. Hong, and James A. Landay. Topiary: A tool for prototyping location-enhanced applications. In *Proceedings of the ACM Symposium on User Interface Software and Technology (UIST)*, pages 217–226. ACM Press, 2004.

12. James Lin. Damask: A tool for early-stage design and prototyping of cross-device user interfaces. In *Supplement of the ACM Symposium on User Interface Software and Technology (UIST)*, pages 13–16. ACM Press, 2003.

13. Christian Lindholm and Turkka Keinonen. *Mobile Usability: How Nokia Changed the Face of the Mobile Phone*. McGraw-Hill, 2003.

14. Paul Luff and Christian Heath. Mobility in collaboration. In *Proceedings of ACM CSCW'98 Conference on Computer-Supported Cooperative Work*, From Single-Display Groupware to Mobility, pages 305–314, 1998.

15. E.D. Mynatt, J. Rowan, S. Craighill, and A. Jacobs. Digital family portraits: Providing peace of mind for extended family members. In *Proceedings of the Conference on Human Factors in Computing Systems (CHI)*, pages 333–340. ACM Press, 2001.

16. Sharon Oviatt. Multimodal interfaces for dynamic interactive maps. In *Proceedings of the Conference on Human Factors in Computing Systems (CHI)*, volume 1 of *PAPERS: Multi-Modal Applications*, pages 95–102, 1996.

17. Huatong Sun. *Expanding the Scope of Localization: A Cultural Usability Perspective on Mobile Text Messaging Use in American and Chinese Contexts*. PhD thesis, Rensselaer Polytechnic Institute, 2004.

18. Scott Weiss. *Handheld Usability*. John Wiley and Sons, 2002.

19. Leila A. Takayama Xiadong Jiang, Jason I. Hong and James A. Landay. Ubiquitous computing for firefighters: Field studies and prototypes of large displays for incident command. In *Proceedings of the Conference on Human Factors in Computing Systems (CHI)*, pages 679–686. ACM Press, 2004.