Into the Wild: Low-Cost Ubicomp Prototype Testing

Yang Li and James A. Landay
University of Washington and Intel Research

ActivityStudio lets designers easily manage moderate-scale in situ user tests of ubicomp applications.

Ubiquitous computing vastly expands the spectrum of human activities that computing technologies can support beyond traditional office activities (M. Weiser, "The Computer for the 21st Century," *Scientific American*, vol. 265, no. 3, 1991, pp. 94-104). In particular, ubicomp applications that address long-term, everyday activities such as keeping fit and aging in place can simplify our lives and positively influence our behaviors.

These applications, however, raise new challenges for interaction design. It's difficult to determine how an application design might affect users' activities solely based on qualitative or quantitative data collected from participants performing a set of well-defined tasks. Consequently, it's often crucial to test a design in situ—in an ecologically valid environment for an extended period.

But testing a design in situ is time-consuming and prohibitively expensive. It often takes a team of skilled researchers and engineers months to create and deploy an application prototype and set up necessary infrastructures for an in situ test. Moreover, existing UI design tools primarily target laboratory-oriented testing environments.

Recent prototyping tools such as Topiary (Y. Li, J.I. Hong, and J.A. Landay, "Topiary: A Tool for Prototyping Location-Enhanced Applications," *Proc. 17th Ann. ACM Symp. User Interface Software and Technology,* ACM Press, 2004, pp. 217-226) and the Designer's Augmented Reality Toolkit (B. MacIntyre et al., "DART: A Toolkit for Rapid Design Exploration of Augmented Reality Experiences," *Proc. UIST 2004*, pp. 197-206) for location-enhanced applications have pushed the boundary of testing to the field. However, these tools are still limited to controlled, short-term design testing.

Rapid prototyping tools enable designers to quickly iterate on ideas by creating and testing application prototypes at a low cost. These tools afford a fluid, low-threshold design experience that is important for the early stages of ubicomp design. This advantage, however, often comes at the cost of ignoring technical details that are needed for long-term, in-situ testing. There is thus a tension between the nature of low-cost prototypes and the high degrees of realism and robustness required for in situ testing.

INTEGRATED SUPPORT FOR IN SITU TESTING

To address this challenge, we have developed ActivityStudio (http://activitystudio.sourceforge.net), an open source suite of tools for prototyping and in situ testing of low-cost ubicomp application prototypes.

The tool suite's design environment, ActivityDesigner, lets designers create rich ubicomp prototypes in a few hours, using high-level abstractions and techniques such as activity models, storyboards, and programming by demonstration (Y. Li and J. Landay, "Activity-Based Prototyping of Ubicomp Applications for Long-Lived, Everyday Human Activities," *Proc. 26th Ann. SIGCHI Conf. Human Factors in Computing Systems,* ACM Press, 2008, pp. 1303-1312). A designer can test multiple prototypes in ActivityDesigner anytime during a design process, with simulated user activity data or data recorded from previous in situ tests.

Here we focus on the integrated support of ActivityStudio for deploying and testing ubicomp prototypes in situ.

Automatically generating testing infrastructures

To save effort in deploying a design, ActivityDesigner automatically generates computational constructs from a design, such as database structures and testing profiles, and deploys them to ActivityServer, a key component in ActivityStudio's runtime architecture as shown in Figure 1.

ActivityServer maintains users' activity event streams on which prototypes act. Each event is described

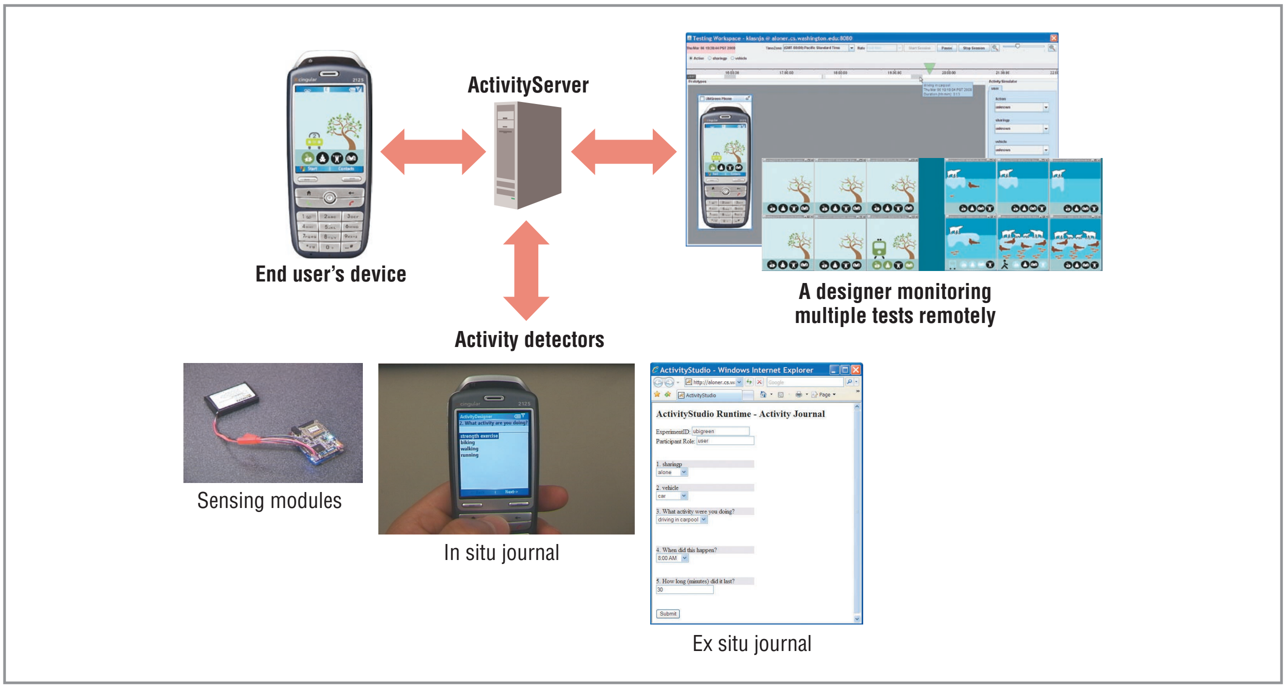

Figure 1. Runtime architecture of ActivityStudio for in-situ testing. The end user's device (shown displaying an environmental awareness application), the workspace for monitoring multiple tests, and the activity detectors communicate with one another through ActivityServer.

by a set of properties—for example, what the user is doing, where the user is, whom the user is with, and the event's time and duration. ActivityServer also includes an optional component for automatically extracting the current screens of multiple prototypes being tested as images. A thin client running on a low-end device like a mobile phone can retrieve and display such images.

Supporting multiple options for running prototypes

To support a wide range of target scenarios, ActivityStudio provides numerous options for running low-cost prototypes created by designers.

For high-end target devices such as a Tablet PC, ActivityStudio provides a virtual machine that can schedule and run multiple prototypes and periodically retrieve users' activity data from ActivityServer.

For low-end devices not able to run the VM, ActivityStudio provides two options. If a JavaScript-compliant Web browser such as Opera Mobile runs on the device, ActivityStudio can automatically convert a prototype to an HTML/JavaScript-based Web application.

Otherwise, an extra-thin client that runs on the phone must periodically retrieve interface screens from ActivityServer, which carries out all interaction computation. This option requires the least support from a target device but offers the least interactivity. For example, a prototype might not function well when the device is disconnected from ActivityServer.

Allowing combined activity sensing

To accommodate the rapidly changing development of sensing technologies, as well as the varying nature of target activities, ActivityStudio employs a flexible architecture that can easily incorporate numerous activity-sensing components. Activity detectors simply send users' activity events to ActivityServer via HTTP Post.

ActivityStudio supports two types of sensing components: those based on "human sensors"—user participants who self-report their activities throughout the day—and those that automatically detect users' activities based on real sensors and inference. The former approach eliminates the effort of incorporating real sensors and allows quicker deployment, but it introduces inconvenient interruptions in users' activities, especially for spontaneous, frequently occurring activities such as making a phone call.

Designers can combine these two types of activity-detection techniques in ActivityStudio. They can choose to let a user report the activity information that's difficult to detect with available sensors and to use sensors to automatically detect other information. For example, a designer might specify that a location-tracking system detects the "where are you?" property and the "who are you with?" property is to be self-reported.

ActivityStudio will fuse events from multiple sources and store them in ActivityServer as an activity stream for each user. It will generate a sensor wrapper that describes how sensing results should be mapped and reformatted to a form ActivityServer understands.

In addition, ActivityStudio provides both a mobile-phone-based in situ journal and a Web-based ex situ journaling UI for user reporting. It automatically composes both interfaces from the testing profiles generated from a design.

The former approach is appropriate for reporting frequently occurring events that involve rich situational factors such as walking up stairs (user action) for a meeting (situation). Users might forget these events or their details if they don't report them right away. The latter approach is suitable for reporting less frequently occurring events throughout the day such as going to the gym.

Monitoring and analyzing multiple test sessions

ActivityStudio makes it possible to test a design with multiple participants at the same time. Designers can monitor these test sessions, including users' activities and the state of the interface screens on the target devices. They can analyze participants' behavior patterns from various dimensions, such as "what the user is doing" or "where the user is," and generate a graph that shows how the number of instances of different user activities changes over time.

CASE STUDIES

Here we discuss two design projects that used ActivityStudio for creating prototypes and subsequent in-situ testing. These projects employed various testing options provided by ActivityStudio for two different target domains: keeping fit and environmental protection.

Improving fitness awareness

We used ActivityStudio to design an ambient display called Fitness Fish Display. Partially inspired by an interactive computer game called *Fish'n'Steps* (J.J. Lin et al., "*Fish'n'Steps*: Encouraging Physical Activity with an Interactive Computer Game," *UbiComp 2006: Ubiquitous Computing*, LNCS 4206, Springer, 2006, pp. 261-278), our display represents different physical activities that users perform to keep fit as fish. Users can also explicitly interact with the display to look at detailed information about their workouts. A swimming gold fish appears on the display after a user achieves a weekly exercise goal.

To understand how users would interact with Fitness Fish Display in their day-to-day environment, we deployed the system in the homes of two participants for one week using Tablet PCs as target devices. We copied the design files to these devices and, using ActivityDesigner's Testing Workspace, created a test session for each participant on ActivityServer. These tasks took only a few minutes to complete.

ActivityStudio provides both a mobile-phone-based in situ journal and a Web-based ex situ journaling UI for user reporting.

In this study, we experimented with both Web-based ex situ and mobile-phone-based in situ journaling approaches for simulating activity sensing. Each participant used a different method to report workout activities, durations, locations, and colocated people. ActivityStudio automatically generated both journaling UIs from testing profiles exported from the ambient display design. The participant who used the in situ method reported that it took him less than a minute to report an event by answering four brief questions on a mobile phone.

Based on the one-week study, we designed a second version of the application that addressed one participant's feedback that a single user display is boring. In the Collaborative Fish Display, two separate fish represent the status of both participants' activities. A fish swims higher and grows bigger when its owner is more physically active. The new design addressed privacy concerns by making participants' workout details, such as location, only visible to themselves.

We deployed Collaborative Fish Display with the two participants for an additional two weeks. From this study, we observed that the number of physical activities of both participants significantly increased. Although it's impossible to make claims about behavior change given the short study period and small sample size, these encouraging results did suggest that participants could successfully integrate this new technology into their everyday lives.

Encouraging environmental protection

Intel, University of Washington, and Carnegie Mellon University researchers used ActivityStudio to design UbiGreen, a mobile-phone-based transportation monitor that helps people make personal changes to improve our global environment. UbiGreen gives users positive feedback when their transportation patterns have less impact on the environment—for example, by walking, biking, or taking public transit.

By using ActivityStudio in this project, the designers were able to easily explore two distinct metaphors: a changing polar ecosystem and a tree that grows leaves to visualize users' transportation activity information. The ambient display appears in the background of an individual's mobile phone.

To determine which design option is more effective, the researchers conducted a two-week in situ study with 12 participants divided into two groups that each used one version of UbiGreen. The study employed a thin-client approach in which a mobile phone periodically retrieved the latest interface screen generated on ActivityServer for a particular participant. The researchers could easily start the required 12 test sessions simultaneously on ActivityServer.

ActivityStudio detected participants' transportation activities using both Intel's Mobile Sensing Platform (T. Choudhury et al., "The Mobile Sensing Platform: An Embedded System for Capturing and Recognizing Human Activities," *IEEE Pervasive Computing*, vol. 7, no. 2, 2008, pp. 32-41) and the mobile-phone-based in situ journaling method for discriminating between different types of vehicles—for example, car pooling versus driving alone.

The thin-client solution made it possible to easily update and redeploy a design during the study through ActivityServer, without having to make the participants return to the lab to update their phones.

In situ testing of ubicomp applications is challenging. ActivityStudio provides integrated support for testing low-cost ubicomp prototypes in ecologically valid environments over extended periods. It offers a range of options for running these prototypes on target devices and detecting users' activities. This variety is necessary to meet the testing requirements of a diverse set of applications. Several case studies show that ActivityStudio significantly lowers the cost of in situ testing and lets designers easily manage moderate-scale in situ user tests.

***Yangi Li** is a research associate in the Computer Science and Engineering (CSE) Department at the University of Washington. Contact him at yangli@cs.washington.edu.*

***James A. Landay** is an associate professor in the CSE Department at the University of Washington and a strategic consultant for Intel Research. Contact him at landay@cs.washington.edu.*

Editor: Bill N. Schilit, Google; schilit@computer.org