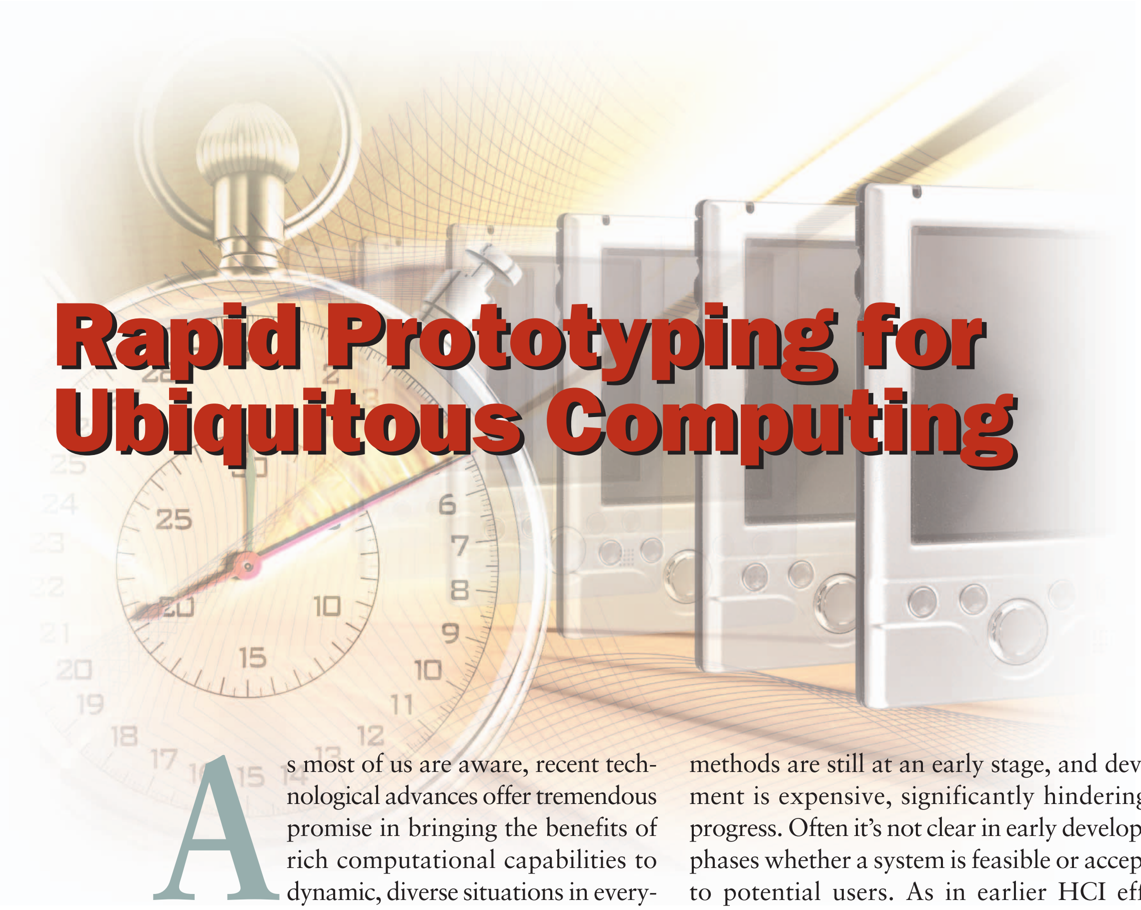


Rapid Prototyping for Ubiquitous Computing

Nigel Davies
Lancaster University

James Landay
University of Washington

Scott Hudson
Carnegie Mellon University

Albrecht Schmidt
University of Munich, Germany

As most of us are aware, recent technological advances offer tremendous promise in bringing the benefits of rich computational capabilities to dynamic, diverse situations in everyday life. However, much of this promise remains unfulfilled. We can't yet use this enormous computing power to facilitate our day-to-day lives and regularly deliver compelling ubiquitous computing applications. Our situation is analogous to the early 1980s and the then-new concept of GUIs. We could see that GUIs had potential, but building any new application with them was a major undertaking, and building good interfaces seemed extremely difficult. To overcome these barriers, researchers and developers created a series of tools and methodologies over time to explore many alternatives quickly with minimal effort. Those prototyping tools and techniques had a much-needed enabling factor that contributed greatly to the success of what we now consider traditional interfaces.

Today, building ubicomp systems for realistic scenarios is difficult. Compared to simpler, more mature interface domains, development tools and methods are still at an early stage, and development is expensive, significantly hindering our progress. Often it's not clear in early development phases whether a system is feasible or acceptable to potential users. As in earlier HCI efforts, progress in prototyping methods and tools will be central to overcoming the barriers to widespread development and deployment of ubicomp. This need is particularly strong owing to our envisioned systems' high-level complexity, the implementation challenges of using many small and distributed devices, the multidisciplinary questions involved, and the need to understand and evaluate the full impact of the systems we build.

Implementing ubicomp scenarios

Because ubicomp scenarios often consist of different undeveloped components, a complete implementation might be impractical, and a partial implementation can't show the full potential. This presents a dilemma, particularly in research and early development, because researchers and developers must concentrate on their specific area to advance technology rather than expend effort on broad system-implementation issues.

This dilemma can be partially overcome by rapidly prototyping the whole system while focusing most of the engineering, design, and evaluation effort on the specific area of interest. It's then possible to have a complete system

1536-1268/05/$20.00 © 2005 IEEE ■ *Published by the IEEE CS and IEEE ComSoc*

working, letting us clearly assess a new development. Tools for automating development and evaluation activities have always played an important role in this type of activity. Two articles in this issue illustrate new advances in this area. The first is "Wizard of Oz Support throughout an Iterative Design Process," by Steven Dow, Blair MacIntyre, Jaemin Lee, Christopher Oezbek, Jay David Bolter, and Maribeth Gandy. The second is "Multipurpose Prototypes for Assessing User Interfaces in Pervasive Computing Systems," by Bonnie E. John and Dario D. Salvucci.

Researchers have also appropriated traditional user interfaces' lessons and methods for use in pervasive computing. In particular, researchers have demonstrated the Wizard of Oz technique's potential in creating working ubicomp systems. In addition to Dow and his colleagues' article, other articles in this issue discuss this topic. The first is "The Webkit Tangible User Interface: A Case Study of Iterative Prototyping," by Mark Stringer, Jennifer A. Rode, Eleanor F. Toye, Alan F. Blackwell, and Amanda R. Simpson. The second is "Evaluating Early Prototypes in Context: Trade-offs, Challenges, and Successes," by Derek Reilly, David Dearman, Michael Welsman-Dinelle, and Kori Inkpen. Additionally, John and Salvucci demonstrate how to adapt sophisticated cognitive modeling approaches, previously applied only in traditional interfaces, to pervasive contexts.

Overcoming multidisciplinary boundaries

Because researchers embed successful ubicomp projects in rich real-world contexts that can touch many aspects of life, they've made multidisciplinary teams the norm rather than the exception. However, overcoming boundaries between various disciplines is a significant challenge and in many cases represents a key factor for successful development. Problem-solving approaches differ radically, and finding common ground for assessing results can be difficult.

Means for developing and sharing sketches for solutions early on can help solve these problems. Prototyping, especially low-fidelity prototyping, is an interesting, important starting point for joined research and development processes. Several articles in this issue showcase examples of paper prototypes or other noninteractive approaches. In addition to Stringer and his colleagues' article, one article is "Prototypes in the Wild: Lessons from Three Ubicomp Systems," by Scott Carter and Jennifer Mankoff. The second article is "Prototypes and Paratypes: Designing Mobile and Ubiquitous Computing Applications," by Gregory D. Abowd, Gillian R. Hayes, Giovanni Iachello, Julie A. Kientz, Shwetak N. Patel, Molly M. Stevens, and Khai N. Truong. The third is "Rapid Prototyping and User-Centered Design of Interactive Display-Based Systems," by Dan Fitton, Keith Cheverst, Chris Kray, Alan Dix, Mark Rouncefield, and George Saslis-Lagoudakis.

This experience clearly shows that we can adapt such techniques to pervasive computing requirements. The techniques serve well as versatile tools in an overall suite of techniques moving from lower-fidelity prototyping approaches at early development stages to higher-fidelity and more technological approaches as a design matures.

Understanding envisioned systems

Developing complex systems isn't a new problem. However, when looking at ubicomp systems, understanding the full complexity is often different and more difficult than in areas of more bounded scope. In realistic ubicomp systems, this involves interactions between system components and users. Furthermore, set-

IEEE Computer Society Publications Office

10662 Los Vaqueros Circle, PO Box 3014
Los Alamitos, CA 90720-1314

STAFF

Lead Editor
Shani Murray
smurray@computer.org

Group Managing Editor
Crystal R. Shif
cshif@computer.org

Senior Editors
Dale Strok, Dennis Taylor, and Linda World

Staff Editor
Rita Scanlan

Editorial Assistants
Brooke Miner and Molly Mraz

Magazine Assistant
Hilda Hosillos
pervasive@computer.org

Contributing Editor
Joan Taylor

Design Director
Toni Van Buskirk

Layout & Technical Illustrations
Carmen Flores-Garvey and Alex Torres

Publisher
Angela Burgess
aburgess@computer.org

Associate Publisher
Dick Price

Membership & Circulation Marketing Manager
Georgann Carter

Business Development Manager
Sandra Brown

Advertising Coordinator
Marian Anderson

Submissions: Access the IEEE Computer Society's Web-based system, Manuscript Central, at http://cs-ieee.manuscriptcentral.com/index.html. Be sure to select the right manuscript type when submitting. Articles must be original and should be approximately 5,000 words long, preferably not exceeding 10 references. Visit www.computer.org/pervasive for editorial guidelines.

Editorial: Unless otherwise stated, bylined articles as well as products and services reflect the author's or firm's opinion; inclusion does not necessarily constitute endorsement by the IEEE Computer Society or the IEEE.

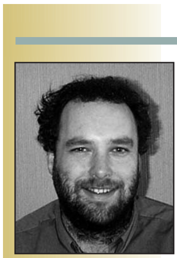
IEEE pervasive COMPUTING
MOBILE AND UBIQUITOUS SYSTEMS

tings have been changing with regard to infrastructure and context of use. When creating complex interactive systems and services, it's hard to predict how users will react. In traditional systems, researchers can base predictions (for example, how long it will take to fill in a form on a screen) on well-established data. In pervasive computing systems, such data often isn't available. John and Salvucci show how we can begin to address some of these questions for expert performance; however, many questions about novice user performance and overall user acceptance can't be easily answered in advance. In such cases, building prototype systems is an essential means to finding answers.

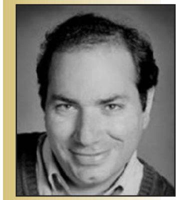
Research shows that prototyping and deploying systems for study is important to understanding how systems fit into the user's world and how they can be used effectively. Designing, building, and deploying systems help both researchers and developers better understand a particular application domain's key issues. This issue provides a rich body of experience in issues associated with prototype deployment.

This issue's articles represent some of the best recent advances in applying rapid prototyping to ubiquitous-systems development. The articles have well-grounded approaches and include practical advice about the pitfalls and difficulties (as well as successes) that can accompany prototyping in this area. They review a wide range of methods suitable at different design and development stages and, as a whole, offer rich experience that should be helpful for anyone seeking to use rapid prototyping approaches in ubicomp work. P

the AUTHORS



Nigel Davies is a professor of computer science at Lancaster University and an adjunct associate professor of computer science at the University of Arizona. His research interests include systems support for mobile and pervasive computing. He focuses in particular on the challenges of creating deployable mobile and ubiquitous computing systems that can be used and evaluated "in the wild." He's an associate editor of *IEEE Pervasive Computing*. Contact him at the Computing Dept., InfoLab 21, South Dr., Lancaster Univ., Lancaster, LA1 4WA UK; nigel@comp.lancs.ac.uk.



James Landay is an associate professor in the Computer Science and Engineering Department at the University of Washington, specializing in human-computer interaction. He is also the laboratory director of Intel Research Seattle. His research interests include automated usability evaluation, demonstrational interfaces, ubiquitous computing, user interface design tools, and Web design. He received his PhD in computer science from Carnegie Mellon University. He's an associate editor of *IEEE Pervasive Computing*. Contact him at the Univ. of Washington, Dept. of Computer Science & Eng., 642 Paul G. Allen Center, Box 352350, Seattle, WA 98195-2350; landay@cs.washington.edu; www.cs.washington.edu/homes/landay.

Scott Hudson is a professor of human-computer interaction in the HCI Institute at Carnegie Mellon University, where he serves as the director of the HCII PhD program. His research interests include topics in technological HCI, including situationally appropriate interaction, user interface tools, and aspects of aesthetics, design, and desirability in the interface. He has served as the general and program chair for the ACM User Interface Software and Technology conference and as an associate editor for the *ACM Transactions on Computer Human Interaction*. Contact him at the Human-Computer Interaction Inst., School of Computer Science, Carnegie Mellon Univ., 5000 Forbes Ave., Pittsburgh, PA 15213-3891; scott.hudson@cs.cmu.edu; www.cs.cmu.edu/~hudson.

Albrecht Schmidt is the head of the Embedded Interaction research group at the University of Munich, Germany. His research interests include ubiquitous computing and context awareness, specifically novel user interfaces and new interaction methods. He received his PhD in computer science from Lancaster University. Contact him at the Univ. of Munich, Amalienstr. 17, 80333 Munich, Germany; albrecht@hcilab.org; www.hcilab.org/albrecht.