Can Early-Stage Tools and Techniques for Iterative Design Help Researchers Understand a Problem Space?

Scott Davidoff [1], Scott Carter [2], Jennifer Mankoff [1]

[1] Human-Computer Interaction Institute
Carnegie Mellon University
{scott.davidoff,jmankoff}@cs.cmu.edu

[2] EECS Department
UC Berkeley
sacarter@cs.berkeley.edu

Abstract. Researchers developing Ubicomp applications often must make ill-informed but irrevocable decisions early in the design process. While desktop computing researchers have multiple methods at their disposal to manage the risk involved in these decisions, the complexity of Ubicomp research affords few alternatives. We suggest that Ubicomp research faces a poverty of effective design process. We explore alternatives that might supplement existing design processes so that designers can make decisions from positions of information. This suggests an opportunity to develop both tools and techniques that support early-stage evaluation.

1 Introduction

Fourteen years after Weiser's vision [18], Ubicomp research has made modest progress towards achieving that end. Now is an appropriate time to reflect on why.

In this paper, we suggest one reason: when developing applications, *researchers cannot evaluate the problem space*. Ubicomp applications are bound by an interesting set of circumstances. Because applications must address multiple, unpredictable contexts of use, researchers need to evaluate them in the field. But developing applications that are field-deployable often involves sufficient cost that researchers are forced to arbitrarily limit the design space early in the design process. Researchers cannot evaluate a problem without first building a technology. But deployment requires knowledge that researchers do not have.

Assuming no "silver bullets" [1] arrive on the scene, what can Ubicomp researchers do? We suggest that new tools and techniques that support early-stage evaluation might help researchers make better decisions, early enough to have impact.

In this paper, we describe knowledge gaps in the design process of one Ubicomp research project we worked on. We then survey evaluation techniques inherited from the desktop world, and describe their shortcomings for Ubicomp. Lastly, we describe the research opportunities afforded by these shortcomings.

2 Case Study

Since considerable barriers of access, skill and intimidation keep elders offline [10], we hypothesized that a device that hides email services within familiar objects might help elders get the benefits of email, without asking them to absorb high learning costs. We quickly identified two candidate technologies to deliver on this promise: an augmented telephone, and a paper-to-email bridge. But how would we know which system would produce superior results?

Traditional exploratory methods failed to meet our needs. Laboratory evaluation – even "future scenario" [2] games of imagination – could never simulate the highly contextual factors central to our investigation. To really know which system would provide superior results would mean developing and evaluating two functioning systems. But to do so would require significant expense.

Unable to commit such substantial resources, we opted to interview elders. After examining elders' communication process and technological comfort zone, we choose letter-writing. We ultimately developed ElderMail [9], a tangible email system that uses a book as a user interface, or BUI. But we cannot say in earnest that our decision produced optimal results. Lacking a low-cost way to evaluate vastly different competing technical alternatives, we had no way to accurately predict the relative efficacy of one solution over another.

3 Early-Stage Evaluation, Evaluated

Evaluation is traditionally characterized as *formative* – the up-front exploration of a problem space – or *summative* – the retrospective measurement of system impact. But since Boehm's spiral model [3] refocused software development, researchers often perform what we describe as *iterative evaluation* – a more rapid and repetitive design-build-test cycle. And though we have inherited multiple iterative evaluation techniques from the desktop world, few meet the diverse needs of early-stage Ubicomp research.

Software toolkits and interface builders evolved to substantially meet the needs of desktop research. And we can already find toolkits designed for Ubicomp, e.g. [12]. But while these toolkits lower the cost of creating field-deployable technologies, they still do not support the kind of low-cost, high-level decision-making early-stage research requires.

Another strategy suggests replacing the user with a computer models that simulate human input, e.g. [6]. But Ubicomp systems exist in a variety of environments that are currently too complex and dynamic to accurately model with the predictive power required to enable design decisions.

Other researchers, e.g. [1], turn to social science theory for guidance. But social science theory is currently too limited to provide accurate predictions for complex behaviors in situations involving multiple, and still largely-unknown variables and effects.

Paper prototyping [16] has evolved as an extremely low-cost system proxy. Because paper is such a naturally flexible medium [17], it shows distinct promise for prototyping smaller Ubicomp systems. Some researchers are exploring how paper can be applied to more complex Ubicomp scenarios, e.g. [7]. But paper currently falls short when it meets the dynamic needs of many Ubicomp systems. It cannot scale well to distributed, multi-user applications, or situations that require vast or dynamic input from multiple channels.

Some researchers have explored simulation as a means to explore a complex problem space. We more commonly find these so-called Wizard of Oz (WOz) studies in speech and multimodal interface, e.g. [15], or intelligent user interfaces e.g. [8]. Despite their simulated components, WOz studies often involve substantial programming investments. Researchers still need to create largely working systems, and replace particular components with methods for experimenters to simulate machine input or output.

Some researchers are addressing this particular issue by providing tools to help researchers build WOz simulations for particular domains, such as location-aware applications [13], or speech [11]. Such tools, though helpful, are limited to a single domain. This forces researchers evaluating applications in multiple domains to re-implement simulations to suit the semantics of each prototyping toolkit.

4 Conclusion

To understand how burgeoning Ubicomp applications function in context, researchers will have to find lower-cost ways to evaluate emerging design alternatives. We see an opportunity to develop both tools and techniques to better support this process.

Paper has proven it can capably simulate the user interface. But what techniques are available when researchers want to simulate an environment? What techniques can help researchers design representative tasks, and select appropriate performance metrics for poorly-understood domains? And what techniques are available to compare, analyze, and visualize the complex behavioral variables produced through such evaluations?

When tools are appropriate, what tools might provide the structure to design, evaluate, and analyze multiple design iterations. We suggest that a WOz toolkit might fill this role. A WOz toolkit would have to generalize WOz patterns across multiple application domains. It would also have to provide low-cost ways to integrate multiple input streams, both real and simulated. and support the wizard's complex real-time performance needs during testing. It might also provide special tools to visualize, analyze and explore data from complex and multiple semantics.

Ultimately, any tools and techniques developed would seek to lower the development costs so that researchers can make fewer assumptions about the new and unpredictable contexts of use for which we are only just now beginning to explore, and develop applications that can meet the real and demonstrated needs of users, as observed in their environment.

References

1. Barley, S.R., Dutton, W.H., Kiesler, S., Resnick, P., Kraut, R.E., Yates, J. (2004) Does CSCW Need Organization Theory?, in *Proceedings of CSCW'04*, 122-124.
2. Beyer, H., & Holtzblatt, K. (1998) *Contextual Design: Defining Customer-Centered Systems*, Morgan Kaufman Publishers.
3. Boehm, B. W. (1988) A Spiral Model of Software Development and Enhancement, in *IEEE Computer*, 21(5): 61 - 72.
4. Brooks, F. P. (1995) The Mythical Man-Month: After 20 Years, in *IEEE Software*, 12(5): 57-60.
5. Buchanan, R. (1995) Wicked Problems in Design Thinking. In V. Margolin and R. Buchanan (Eds.) *The Idea of Design*, MIT Press.
6. Card, S. K., Moran, T.P. & Newell, A. (1983) *The Psychology of Human-Computer Interaction*, Erlbaum.
7. Chandler, C. D., Lo, G., & Sinha, A. K. (2002) Multimodal Theater: Extending Low Fidelity Prototyping to Multimodal Applications, in *Proceedings of CHI'02*, 874-875
8. Dahlbäck, N., Jönsson, A., Ahrenberg, L. (1993) Wizard of Oz Studies: Why and How, in *Proceedings of IUI'03*, 193-200.
9. Davidoff, S., Bloomberg, C., Li, I. A. R., Mankoff, J., & Fussell, S. R., (2005) The Book as User Interface: Lowering the Entry Cost to Email for Elders, in *Proceedings of CHI'05*, to appear.
10. Fox, S. (2004) Older Americans and the Internet, *Pew Internet & American Life Project*, http://www.pewinternet.org/pdfs/PIP_Seniors_Online_2004.pdf
11. Klemmer, S. R., Sinha, A. K., Chen, J., Landay, J. A., Aboobaker, N., Wang, A. (2000) Suede: a Wizard of Oz Prototyping Tool for Speech User Interfaces, in *Proceedings of UIST'00*, 1-10.
12. Klemmer, S., Li, J., Lin, J., & Landay, J. (2004) Papier-Mache: Toolkit Support for Tangible Input, in *Proceedings of CHI'04*, 399-406.
13. Li, Y., Hong, J. I., & Landay, J. A. (2004) Topiary: A Tool for Prototyping Location-Enhanced Applications, in Proceedings of UIST'04, 217 – 226.
14. Luff, P., & Heath, C. (1998) Mobility in Collaboration, in *Proceedings of CSCW'04*, 305-314.
15. Oviatt, S., Cohen, P., Fong, M., & Frank, M. (1992) A Rapid Semi-Automatic Simulation Technique for Investigating Interactive Speech and Handwriting, in *Proceedings of Spoken Language Processing'92*, 1351-1354.
16. Rettig. M. (1994) Practical Programmer: Prototyping for Tiny Fingers, *Communications of the ACM*, 37(4): 21-27.
17. Sellen, A. J., Harper, R. H. R. (2003) *The Myth of the Paperless Office*, MIT Press.
18. Weiser, M. (1991) The Computer for the Twenty-First Century, In Scientific American, 3(3): 94-104.