Chapter 1

Introduction

"You don't have to please everyone–you have to please the user."

—Brenda Laurel

Mark Weiser envisioned ubiquitous computing as a world where computation and communication "*blend into the fabric of our everyday lives*" (Weiser, 1991). To realize Weiser's vision, we must find interfaces that are useful, intuitive, efficient, and enjoyable for users in the ubiquitous computing domain. An iterative human-centered design process (Nielsen, 1993) is required to find these interfaces. Currently, only experts can design, prototype, and deploy ubiquitous computing applications; others are lacking the tools and conceptual frameworks to fully support an iterative human-centered design process for ubiquitous computing. This work starts to fill the gap by providing contributions that support each phase of the iterative human-centered design process that addresses the complexity of ubiquitous computing application scenarios.

Ubicomp needs iterative human-centered design

1.1 Iterative Human-Centered Design

The field of Human–Computer Interaction (HCI) has long recognized that user interfaces should be designed iteratively (Nielsen, 1993; Buxton and Sniderman, 1980; Gould and Lewis, 1985), because the requirements for an interactive system cannot be completely specified at the beginning of the lifecycle (Dix *et al.*, 2004). Instead, the road to success in interaction design is to *fail early and often*. The design needs to be prototyped and tested with real users to reveal any false assumptions or unforeseen design problems. These problems can then be corrected in the next iteration of the prototype, which should then again be tested to ensure the problems are resolved. Each prototype is more detailed and functional than the last, thus converging towards the final system (see figure 1.1). The main phases of iterative design are:

the road to success in interaction design is to fail early and often

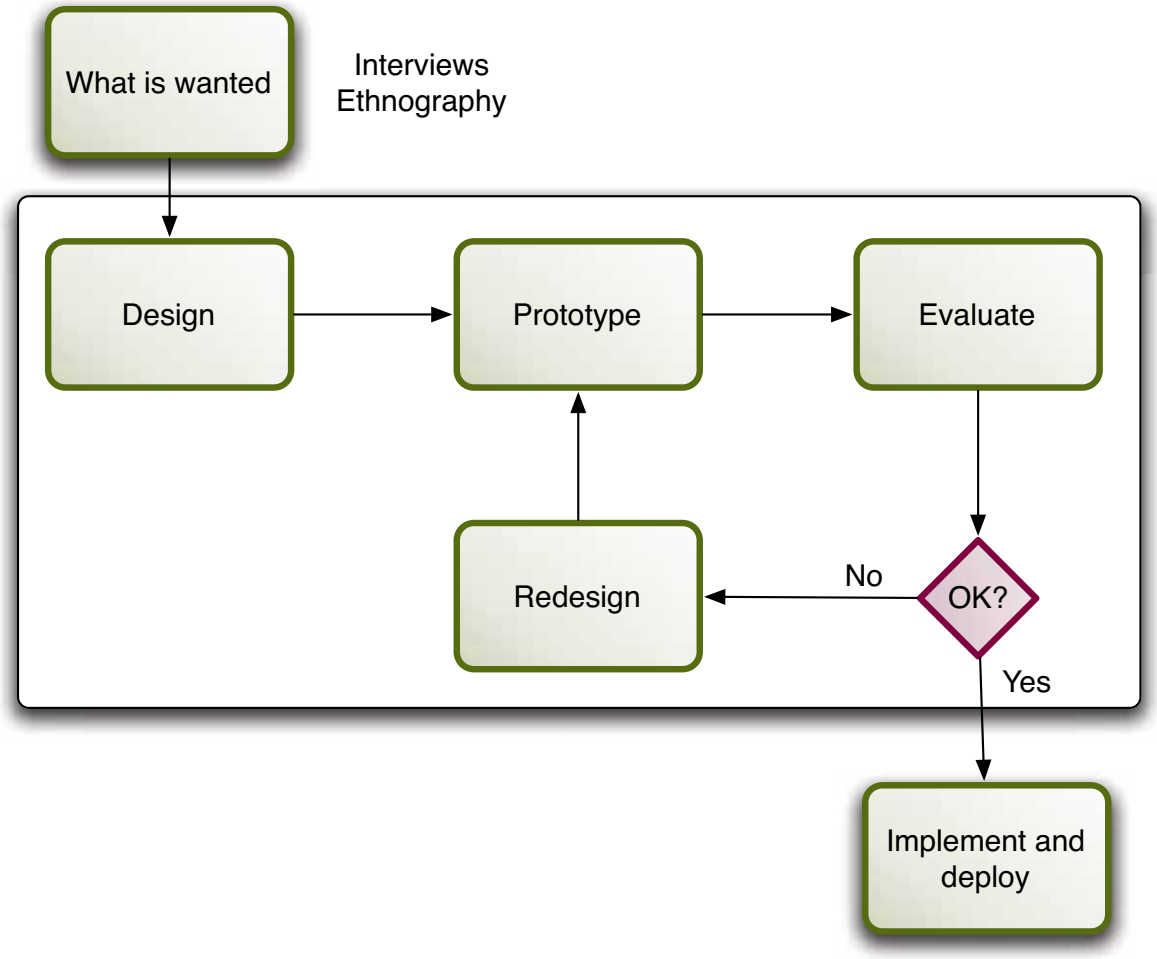

Figure 1.1: A high-level diagram illustrating the iterative design process (adapted from Dix *et al.* (2004))

know the user

- **Requirements (what is wanted)**– In human-centered design, the primary task is to 'know the user' (Hansen, 1971). When beginning to develop any interactive system, it is important to clearly identify who the system is intended to support, and for which tasks. After the user group is identified, exploratory techniques such as contextual inquiry or ethnography can be used to derive user needs and system requirements.

existing design knowledge can help conceptualize an interface

- **Design** – This is the stage of the design process where the system requirements are translated into a design solution. This stage can be informed through design knowledge captured by abstract design guidelines (Mayhew, 1991), platform specific design guidelines (Apple Computer Inc., 1992), heuristics (such as Shneiderman's golden rules (Shneiderman, 1992)), and HCI design patterns (Borchers, 2001). Other tools that can assist in making informed design decisions are design spaces, such as Card *et al.*'s design space of input devices (Card *et al.*, 1991), which helps designers reason about design alternatives and identify the most appropriate design for the given

task.

- **Prototyping** – Early in the development of a product, prototypes are typically conceptual in the form of scenarios, sketches, and storyboards that illustrate the basic usage in context. Later after evaluation, more detailed prototypes flush out concrete design ideas. With each cycle in the iterative design process, the ideas are further refined with a combination of functional (works like) and form (looks like) prototyping strategies.

Prototype refinement advances with each iteration

- **Evaluation** – The essence of iterative design is to evaluate the prototypes early and often to identify problems and design flaws early in the design process. Delaying meaningful testing increases the cost of correcting fundamental design problems. Analysis can either be done without users, such as employing an expert to perform a heuristic analysis, or they can be tested by observing real users interacting with the product either in controlled experiments or in real context of use. All of these forms of evaluation should be used together throughout the design process to identify potential difficulties users may have with a product. After the problems are identified, they can be translated into design changes for the next iteration of the prototype (Dix *et al.*, 2004).

Evaluate early and often with real users

- **Implementation and Deployment** – Once the design is of acceptable quality, the creation of production quality code, the manufacturing of robust and integrated hardware, and the creation of documentation and manuals can begin. Commonly, the output of the iterative design process is a full design specification and reference prototype, not the final implementation. Evaluations should continue throughout the implementation stage to ensure that the implementation meets the quality required by the design. If the quality cannot be met, further design iterations may be necessary.

Design before implementation

Prototyping structures innovation, collaboration, and creativity in the most successful design studios (Kelley and Littman, 2001). Designers use prototypes as physical representations of ideas, effectively externalizing cognition and facilitating a "conversation with materials" to uncover surprising problems or generate suggestions for new designs (Schön and Bennett, 1996). Prototypes also serve as artifacts that represent tacit knowledge of developers as a communication tool to clients or other members of a design team (Schrage, 1999). Most importantly, prototypes provide an artifact to test with real users as a part of a human-centered iterative design process.

Prototypes externalize cognition

Research has shown that, generally speaking, the more iterations in the design process, the better the user interface (Nielsen, 1993). Figure 1.2 illustrates how usability improves with each iteration in the design process. At the surface, it appears that each iteration is a time consuming and expensive process, but studies have shown that iterative design has economic value (Karat, 1990). Additionally, the cost of performing design iterations can be dramatically decreased with rapid prototyping strategies, but the

More iterations lead to better designs

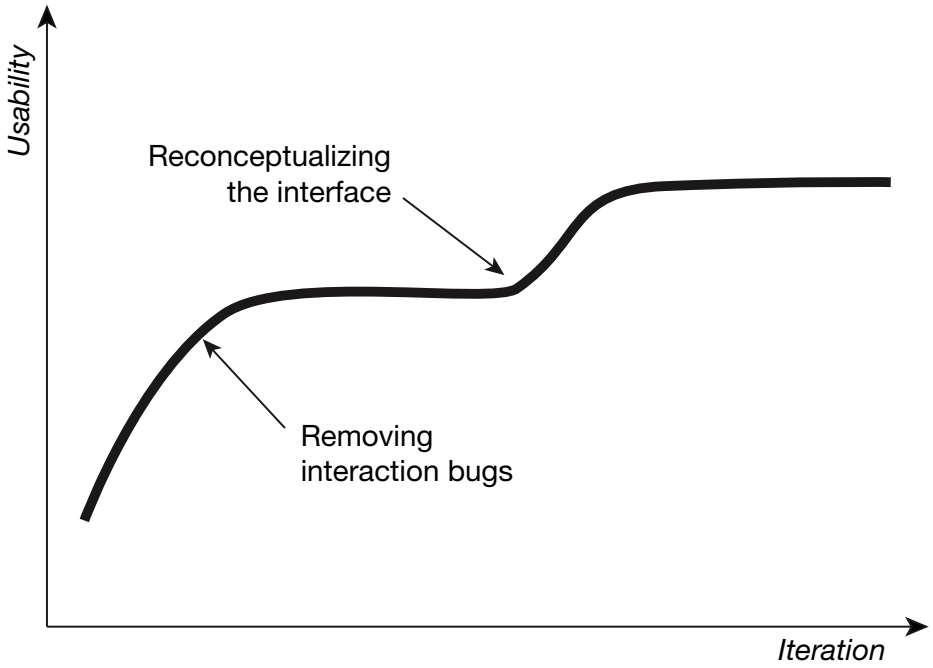

Figure 1.2: An illustration of interface quality as a function of the number of design iterations. Each additional iteration increases the usability of the design until a potential "usability plateau" is reached. (Nielsen, 1993)

nature of the prototype influences the nature of the problems that can be identified.

Prototype both form and function

Prototypes can generally be characterized as one of two variants: functional (*works like*) prototypes try to match the interactive experience as closely as possible (Buchenau and Suri, 2000), and form (*looks like*) prototypes are passive and try to match the appearance and affordances of the final design. These two characterizations can be seen as two extremes of a prototype continuum where, in most practical situations, prototypes have aspects of both. For example, humans can simulate the interactive functionality of form prototypes by updating the state of the form prototype manually. This practice, known as Wizard of Oz prototyping (Dahlbäck *et al.*, 1993), was originally developed in the context of natural language interfaces where a hidden human stenographer typed in spoken text to simulate high-performance natural language processing (Kelley, 1984). This technique allows the interactions to be tested before significant effort is placed in the implementation.

Beware of false starts

One of the pitfalls of iterative human-centered design is that if you pick a poor starting point, you may reach a peak in the usability of a particular design without reaching the desired usability goals. In this case, it may be necessary to throw the design away and start over. False starts are relatively painless early in the design process if low-fidelity prototyping techniques are used, but can be extremely expensive if determined late in the design process. In order to minimize the risk of false starts, a parallel design

strategy (Nielsen and Faber, 1996) can be used, where multiple designs can be explored independently early in the design process. As the designs mature, the best design becomes clear, or the strengths of the top designs can be merged to a unified design. Parallel design is more practical early in the design process when rapid prototyping techniques are used.

1.2 Applying Iterative Design to Ubicomp

Today, strategies for applying an iterative design process to desktop graphical user interfaces are generally well-defined. Figure 1.3 exemplifies how these strategies might be applied over the course of a design process for a desktop application. However, attempting to apply an identical design process to ubiquitous computing is problematic, often because ubiquitous computing application scenarios require a functional prototypes to convey the intended experience. Currently, functional prototypes for ubiquitous computing are costly, time-consuming, and require technical expertise to construct. For example, Heiner *et al.* (1999) report spending about one person-year developing a ubiquitous peripheral display. If meaningful testing is delayed until too late in the design process, monetary constraints and resource commitments prohibit fundamental design changes (Ulrich and Eppinger, 1995).

Ubicomp requires functional prototypes

1.2.1 Fieldwork

Fieldwork that has examined current design practices indicates that there are many issues obstructing iterative design in ubiquitous computing applications.

Hartmann *et al.* (2006) conducted fieldwork interviewing product designers. They found that most product designers have had exposure to programming, but few were proficient. Although access to programmers and engineers was available, there were not enough to complete large prototyping projects. This resulted in a perception that interdisciplinary teams slow the interaction design process and increase costs. Thus, prototypes that combined form and function were not built until late in the design process. These prototypes were typically expensive, one-off presentation tools instead of artifacts for human-centered reflective practice.

Functional prototypes are often delayed

Klemmer (2004) conducted structured interviews with tangible user interface developers. For these developers, dealing with physical input was the primary challenge requiring a high level of technical expertise and extensive development effort. One developer commented "the sensing hardware is not perfect, so sometimes we had to change interactions a bit to make them work in the face of tracking errors." Developers reported that often extensive system redesigns were required to perform straightforward changes to input technologies (e.g., exchanging a camera and barcode reader). Addi-

Effort to build functional prototypes is too high

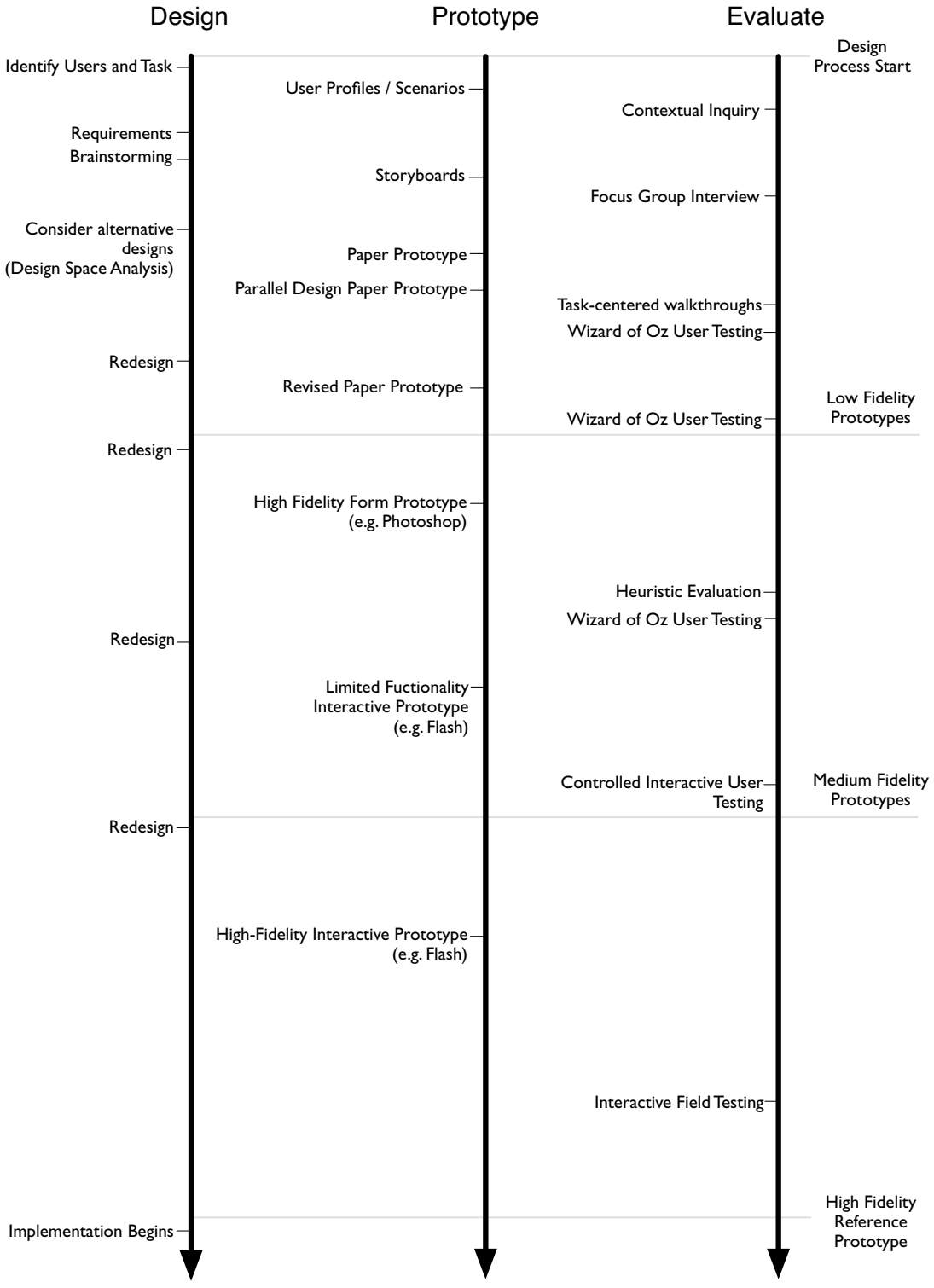

Figure 1.3: An abstract timeline illustrating a sample desktop iterative design process. Low-fidelity prototypes can be prototyped and evaluated much quicker than high fidelity prototypes. Identifying design flaws earlier in the iterative design process saves time and money.

tionally, each development team was creating their own software architectures based on basic event-based software design patterns from the ground up because no tool existed that could save developers time and effort.

Carter *et al.* (2007b) conducted fieldwork examining current design practices of applications for mobile devices (personal digital assistants or mobile phones). The biggest challenge in this domain was developing prototypes robust enough for use "in the wild". Similarly, Kjeldskov and Graham (2003) reviewed many mobile HCI projects and concluded that many mobile developers rarely used lightweight prototypes because they strongly believed it was important to test their tools in a realistic and ecologically valid setting. Developers found that lightweight prototypes were insufficient to perform these types of evaluations.

Functional prototypes are needed for ecologically valid evaluations

Matthews (2005) conducted fieldwork of peripheral display developers. One developer interviewed commented "I would say the hardest part about implementing these displays is the mechanics of doing it...". Participants were interested in building and deploying functional prototypes as rapidly as possible because the "real value in many of these systems is only apparent longitudinally." Developers interviewed also expressed a need for tools that support building applications that combined distributed input and output over multiple modalities (physical, graphical, or audio).

Functional prototypes are needed for longitudinal studies

1.2.2 Lightweight Prototypes

The most prevalent low-fidelity prototyping technique for graphical user interfaces is paper prototyping (Snyder, 2003) (see Figure 1.4). Paper prototypes are valuable because of the speed and low cost with which they can be constructed, evaluated, and thrown away or modified. Paper prototypes can be tested using a Wizard of Oz technique (Dahlbäck *et al.*, 1993), where the designer plays the role of the computer to update the paper "display" to respond to user input. Paper prototypes by themselves are low-fidelity form (*looks-like*) prototypes; when combined with Wizard of Oz, they are low-fidelity prototypes of the form and function of the proposed design. The unfinished nature and rough form of these early prototypes can be particularly valuable; end-users often see them as unfinished and provide richer design suggestions (Landay, 1996). A more polished prototype, on the other hand, implies effort and may discourage comments from testers that imply drastic design changes.

Unpolished prototypes can be valuable

This form of low-fidelity prototyping is well suited for the desktop paradigm as the constrained 2D nature of paper is a good match to the experience of using the standard 2D display of desktop environments. However, paper prototyping does not translate well to the ubiquitous computing domain, because it falls short of capturing the ubiquitous computing user experience convincingly (Liu and Khooshabeh, 2003). For example, Rudström *et al.* (2003) used paper prototypes to evaluate a mobile social application, but reported that users had difficulty reflecting upon how their use

Paper prototypes don't capture ubiquitous computing user experiences