Chapter 8.7
Intelligent User Interfaces for Ubiquitous Computing

Rainer Malaka
Bremen University, Bremen

ABSTRACT

Designing user interfaces for ubiquitous computing applications is a challenging task. In this chapter we discuss how to build intelligent interfaces. The foundations are usability criteria that are valid for all computer products. There are a number of established methods for the design process that can help to meet these goals. In particular participatory and iterative so-called human centered approaches are important for interfaces in ubiquitous computing. The question on how to make interfaces more intelligent is not trivial and there are multiple approaches to enhance either the intelligence of the system or that of the user. Novel interface approaches follow the idea of embodied interaction and put particular emphasis on the situated use of a system and the mental models humans develop in their real-world environment.

User interfaces for computational devices can be challenging for both their users and their designers. Even such *simple* things as VCRs or TV sets feature interfaces that many people find too difficult to understand. Reviews and tests of consumer electronic devices very often rank bad usability even higher than technical aspects and the originally intended main function of the devices or features. Moreover, for most modern appliances there is not much technical difference in their core functions. For instance TV sets differ less in quality of display and sound and more in the way the user interacts with the device. This already shows why user interface design is crucial for any successful product. However, we want to extend the question of user interface design in two directions: the user interface should become more intelligent and adaptive and we want more suitable interfaces for ubiquitous computing scenarios.

The first aspect seems to be clear at first sight: intelligent user interfaces are just what we want and nobody will neglect the need for smart, clever, and intelligent technology. But it becomes more difficult if we strip away the buzzwords and dig a bit deeper into the question of what an intelligent user interface actually should do and how

it would differ from an ordinary interface. Would the standard interface then be a stupid one?

The second aspect introduces a new level of complexity: an interface is by definition a clear boundary between two entities. A user interface resides between human and machine; other interfaces mediate, for instance, between networks and computers. In ubiquitous computing we have the problem that there might not be a clear boundary any more. Computers are no longer visible and, in the end, they can disappear from the user's conscious perception. We will, therefore, face the challenge of building an interface for something that is rather shapeless.

In the following, we will go into more detail through these questions and will introduce some general approaches for designing user interfaces. We will see that we can learn from good interface design for other—classical—devices, and that we can apply many of those user interface design principles for ubiquitous computing as well. A central aspect will be the design process that helps to find the right sequence of steps in building a good user interface. After discussing these general aspects of user interface design, we will focus on the specific needs for ubiquitous computing scenarios and finally on how to build intelligent user interfaces—or to be less euphemistic: to avoid stupid interfaces.

BUILDING GOOD USER INTERFACES

The design of a good user interface is an art, which has been ignored for a long time in the information and communication technology (ICT) business. Many software developers just implemented whatever they found useful for themselves and assumed it would also be beneficial for the respective users. However, most users are not software developers and their way of interacting with technology is very different. Sometimes, the result is technology that is highly functional and useful for a small group of people, namely the developers of the system, and highly inefficient, frustrating or even unusable for most other people. Some of the highlights of this dilemma can be found in the communication with the user when something goes wrong: An error message notifying the user: "an error occurred, code 127" might be of some use for the developer and help in his efforts in debugging the system, but a user will hardly be able to understand what went wrong.

Today usability plays a much bigger role and many systems (including computer systems) are now designed with more care for easy and safe usage. On the one hand this is due to legal constraints demanding accessibility, but also due to the fact that many systems do not differ so much in their technical details and vendors have to diversify their products solely in terms of their "look and feel." We now have a wealth of methods, tools, and guidelines, which all help to develop a good user interface (Dix et al., 1998; Mayhew, 1999). However, there is not one single recipe whose application guarantees 100% success. The essence of usability engineering is to work iteratively in order to achieve the goal of better usability. Let us briefly go through these steps and summarize some of the most important issues of usability engineering. For more detailed information, a number of textbooks and research articles can be consulted (Dix et al., 1998; Nielsen, 1993; Shneiderman, 1997).

The first question of usability engineering is the question of what goals we actually want to achieve. The typical list of usability goals contains at least the following five (ISO 9241, 2006):

- **Safety and Security:** Good design should not harm users or other people affected by the use of a product. It should also help to avoid errors made by humans in using the system.
- **Effectiveness:** A good user interface supports a user in solving a task effectively,