Chapter 1

What is interaction design?

1.1   Introduction

How many interactive products are there in everyday use? Think for a minute about what you use in a typical day: cell phone, computer, personal organizer, remote control, soft drink machine, coffee machine, ATM, ticket machine, library information system, the web, photocopier, watch, printer, stereo, calculator, video game . . . the list is endless. Now think for a minute about how usable they are. How many are actually easy, effortless, and enjoyable to use? All of them, several, or just one or two? This list is probably considerably shorter. Why is this so?

Think about when some device caused you considerable grief—how much time did you waste trying to get it to work? Two well-known interactive devices that cause numerous people immense grief are the photocopier that doesn't copy the way they want and the VCR that records a different program from the one they thought they had set or none at all. Why do you think these things happen time and time again? Moreover, can anything be done about it?

Many products that require users to interact with them to carry out their tasks (e.g., buying a ticket online from the web, photocopying an article, pre-recording a TV program) have not necessarily been designed with the users in mind. Typically, they have been engineered as systems to perform set functions. While they may work effectively from an engineering perspective, it is often at the expense of how the system will be used by real people. The aim of interaction design is to redress this concern by

bringing usability into the design process. In essence, it is about developing interactive products[1] that are easy, effective, and enjoyable to use—from the users' perspective.

In this chapter we begin by examining what interaction design is. We look at the difference between good and poor design, highlighting how products can differ radically in their usability. We then describe what and who is involved in interaction design. In the last part of the chapter we outline core aspects of usability and how these are used to assess interactive products. An assignment is presented at the end of the chapter in which you have the opportunity to put into practice what you have read, by evaluating an interactive product using various usability criteria.

The main aims of the chapter are to:

- Explain the difference between good and poor interaction design.
- Describe what interaction design is and how it relates to human-computer interaction and other fields.
- Explain what usability is.
- Describe what is involved in the process of interaction design.
- Outline the different forms of guidance used in interaction design.
- Enable you to evaluate an interactive product and explain what is good and bad about it in terms of the goals and principles of interaction design.

1.2 Good and poor design

A central concern of interaction design is to develop interactive products that are usable. By this is generally meant easy to learn, effective to use, and provide an enjoyable user experience. A good place to start thinking about how to design usable interactive products is to compare examples of well and poorly designed ones. Through identifying the specific weaknesses and strengths of different interactive systems, we can begin to understand what it means for something to be usable or not. Here, we begin with an example of a poorly designed system—voice mail— that is used in many organizations (businesses, hotels, and universities). We then compare this with an answering machine that exemplifies good design.

Imagine the following scenario. You're staying at a hotel for a week while on a business trip. You discover you have left your cell (mobile) phone at home so you have to rely on the hotel's facilities. The hotel has a voice-mail system for each room. To find out if you have a message, you pick up the handset and listen to the tone. If it goes "beep beep beep" there is a message. To find out how to access the message you have to read a set of instructions next to the phone.

You read and follow the first step:

"1. Touch 491".
The system responds, "You have reached the Sunny Hotel voice message center. Please enter the room number for which you would like to leave a message."

---

[1]We use the term *interactive products* generically to refer to all classes of interactive systems, technologies, environments, tools, applications, and devices.

You wait to hear how to listen to a recorded message. But there are no further instructions from the phone. You look down at the instruction sheet again and read:

"2. Touch*, your room number, and #". You do so and the system replies,

"You have reached the mailbox for room 106. To leave a message type in your password."

You type in the room number again and the system replies, "Please enter room number again and then your password."

You don't know what your password is. You thought it was the same as your room number. But clearly not. At this point you give up and call reception for help. The person at the desk explains the correct procedure for recording and listening to messages. This involves typing in, at the appropriate times, the room number and the extension number of the phone (the latter is your password, which is different from the room number). Moreover, it takes six steps to access a message and five steps to leave a message. You go out and buy a new cell phone.

What is problematic with this voice-mail system?

- It is infuriating.
- It is confusing.
- It is inefficient, requiring you to carry out a number of steps for basic tasks.
- It is difficult to use.
- It has no means of letting you know at a glance whether any messages have been left or how many there are. You have to pick up the handset to find out and then go through a series of steps to listen to them.
- It is not obvious what to do: the instructions are provided partially by the system and partially by a card beside the phone.

Now consider the following phone answering machine. Figure 1.1 shows two small sketches of an answering machine phone. Incoming messages are represented using physical marbles. The number of marbles that have moved into the pinball-like chute indicates the number of messages. Dropping one of these marbles into a slot in the machine causes the recorded message to play. Dropping the same marble into another slot on the phone dials the caller who left the message.
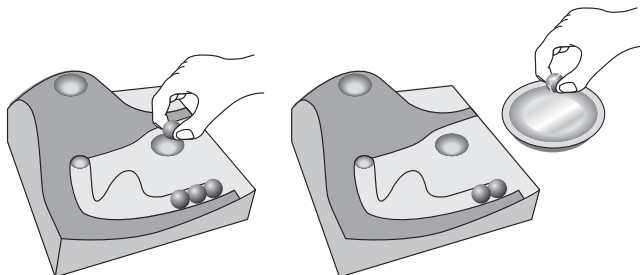

Figure 1.1 Two small sketches showing answering phone.

How does the "marble" answering machine differ from the voice-mail system?

• It uses familiar physical objects that indicate visually at a glance how many messages have been left.

• It is aesthetically pleasing and enjoyable to use.

• It only requires one-step actions to perform core tasks.

• It is a simple but elegant design.

• It offers less functionality and allows anyone to listen to any of the messages.

The marble answering machine was designed by Durrell Bishop while a student at the Royal College of Art in London (described by Crampton-Smith, 1995). One of his goals was to design a messaging system that represented its basic functionality in terms of the behavior of everyday objects. To do this, he capitalized on people's everyday knowledge of how the physical world works. In particular, he made use of the ubiquitous everyday action of picking up a physical object and putting it down in another place. This is an example of an interactive product designed with the users in mind. The focus is on providing them with an enjoyable experience but one that also makes efficient the activity of receiving messages. However, it is important to note that although the marble answering machine is a very elegant and usable design, it would not be practical in a hotel setting. One of the main reasons is that it is not robust enough to be used in public places, for example, the marbles could easily get lost or taken as souvenirs. Also, the need to identify the user before allowing the messages to be played is essential in a hotel setting. When considering the usability of a design, therefore, it is important to take into account *where* it is going to be used and *who* is going to use it. The marble answering machine would be more suited in a home setting—provided there were no children who might be tempted to play with the marbles!

1.2.1   What to design

Designing usable interactive products thus requires considering who is going to be using them and where they are going to be used. Another key concern is understanding the kind of *activities* people are doing when *interacting* with the products. The appropriateness of different kinds of interfaces and arrangements of input and output devices depends on what kinds of activities need to be supported. For example, if the activity to be supported is to let people communicate with each other at a distance, then a system that allows easy input of messages (spoken or written) that can be readily accessed by the intended recipient is most appropriate. In addition, an interface that allows the users to interact with the messages (e.g., edit, annotate, store) would be very useful.

The range of activities that can be supported is diverse. Just think for a minute what you can currently do using computer-based systems: send messages, gather information, write essays, control power plants, program, draw, plan, calculate, play games—to name but a few. Now think about the number of interfaces and interactive devices that are available. They, too, are equally diverse:

multimedia applications virtual-reality environments, speech-based systems, personal digital assistants and large displays—to name but a few. There are also many ways of designing the way users can interact with a system (e.g., via the use of menus, commands, forms, icons, etc.). Furthermore, more and more novel forms of interaction are appearing that comprise physical devices with embedded computational power, such as electronic ink, interactive toys, smart fridges, and networked clothing (See Figure 1.2 on Color Plate 1). What this all amounts to is a multitude of choices and decisions that confront designers when developing interactive products.

A key question for interaction design is: how do you optimize the users' interactions with a system, environment or product, so that they match the users' activities that are being supported and extended? One could use intuition and hope for the best. Alternatively, one can be more principled in deciding which choices to make by basing them on an understanding of the users. This involves:

- taking into account what people are good and bad at
- considering what might help people with the way they currently do things
- thinking through what might provide quality user experiences
- listening to what people want and getting them involved in the design
- using "tried and tested" user-based techniques during the design process

The aim of this book is to cover these aspects with the goal of teaching you how to carry out interaction design. In particular, it focuses on how to identify users' needs, and from this understanding, move to designing usable, useful, and enjoyable systems.

ACTIVITY 1.1   How does making a phone call differ when using:

- a public phone box
- a cell phone?

How have these devices been designed to take into account (a) the kind of users, (b) type of activity being supported, and (c) context of use?

Comment   (a) Public phones are designed to be used by the general public. Many have Braille embossed on the keys and speaker volume control to enable people who are blind and hard of hearing to use them.

Cell phones are intended for all user groups, although they can be difficult to use for people who are blind or have limited manual dexterity.

(b) Most phone boxes are designed with a simple mode of interaction: insert card or money and key in the phone number. If engaged or unable to connect the money or card is returned when the receiver is replaced. There is also the option of allowing the caller to make a follow-on call by pressing a button rather than collecting the money and reinserting it again. This function enables the making of multiple calls to be more efficient.

Cell phones have a more complex mode of interaction. More functionality is provided, requiring the user to spend time learning how to use them. For example, users can save phone numbers in an address book and then assign these to "hotkeys," allowing them to be called simply through pressing one or two keys.

(c) Phone boxes are intended to be used in public places, say on the street or in a bus station, and so have been designed to give the user a degree of privacy and noise protection through the use of hoods and booths.

Cell phones have have been designed to be used any place and any time. However, little consideration has been given to how such flexibility affects others who may be in the same public place (e.g., sitting on trains and buses).

1.3   What is interaction design?

By interaction design, we mean

designing interactive products to support people in their everyday and working lives.

In particular, it is about creating user experiences that enhance and extend the way people work, communicate and interact. Winograd (1997) describes it as "the design of spaces for human communication and interaction." In this sense, it is about finding ways of supporting people. This contrasts with software engineering, which focuses primarily on the production of software solutions for given applications. A simple analogy to another profession, concerned with creating buildings, may clarify this distinction. In his account of interaction design, Terry Winograd asks how architects and civil engineers differ when faced with the problem of building a house. Architects are concerned with the people and their interactions with each other and within the house being built. For example, is there the right mix of family and private spaces? Are the spaces for cooking and eating in close proximity? Will people live in the space being designed in the way it was intended to be used? In contrast, engineers are interested in issues to do with realizing the project. These include practical concerns like cost, durability, structural aspects, environmental aspects, fire regulations, and construction methods. Just as there is a difference between designing and building a house, so too, is there a distinction between interaction design and software engineering. In a nutshell, interaction design is related to software engineering in the same way as architecture is related to civil engineering.

1.3.1   The makeup of interaction design

It has always been acknowledged that for interaction design to succeed many disciplines need to be involved. The importance of understanding how users act and react to events and how they communicate and interact together has led people from a variety of disciplines, such as psychologists and sociologists, to become involved. Likewise, the growing importance of understanding how to design different kinds of interactive media in effective and aesthetically pleasing ways has led to a

diversity of other practitioners becoming involved, including graphic designers, artists, animators, photographers, film experts, and product designers. Below we outline a brief history of interaction design.

In the early days, engineers designed hardware systems for engineers to use. The computer interface was relatively straightforward, comprising various switch panels and dials that controlled a set of internal registers. With the advent of monitors (then referred to as visual display units or VDUs) and personal workstations in the late '70s and early '80s, interface design came into being (Grudin, 1990). The new concept of the user interface presented many challenges:

> *Terror. You have to confront the documentation. You have to learn a new language. Did you ever use the word 'interface' before you started using the computer?*
>
> —Advertising executive Arthur Einstein (1990)

One of the biggest challenges at that time was to develop computers that could be accessible and usable by other people, besides engineers, to support tasks involving human cognition (e.g., doing sums, writing documents, managing accounts, drawing plans). To make this possible, computer scientists and psychologists became involved in designing user interfaces. Computer scientists and software engineers developed high-level programming languages (e.g., BASIC, Prolog), system architectures, software design methods, and command-based languages to help in such tasks, while psychologists provided information about human capabilities (e.g., memory, decision making).

The scope afforded by the interactive computing technology of that time (i.e., the combined use of visual displays and interactive keyboards) brought about many new challenges. Research into and development of graphical user interfaces (GUI for short, pronounced "goo-ee") for office-based systems took off in a big way. There was much research into the design of widgets (e.g., menus, windows, palettes, icons) in terms of how best to structure and present them in a GUI.

In the mid '80s, the next wave of computing technologies—including speech recognition, multimedia, information visualization, and virtual reality—presented even more opportunities for designing applications to support even more people. Education and training were two areas that received much attention. Interactive learning environments, educational software, and training simulators were some of the main outcomes. To build these new kinds of interactive systems, however, required a different kind of expertise from that of psychologists and computer programmers. Educational technologists, developmental psychologists, and training experts joined in the enterprise.

As further waves of technological development surfaced in the '90s—networking, mobile computing, and infrared sensing—the creation of a diversity of applications for *all* people became a real possibility. All aspects of a person's life—at home, on the move, at school, at leisure as well as at work, alone, with family or friends—began to be seen as areas that could be enhanced and extended by designing and integrating various arrangements of computer technologies. New ways of learning, communicating, working, discovering, and living were envisioned.

BOX 1.1 The Relationship between Interaction Design, Human-Computer Interaction, and Other Approaches

We view interaction design as fundamental to all disciplines, fields, and approaches that are concerned with researching and designing computer-based systems for people (see Figure 1.3). The best-known interdisciplinary field is human-computer interaction (HCI), which is "concerned with the design, evaluation, and implementation of interactive computing systems for human use and with the study of major phenomena surrounding them" (ACM SIGCHI, 1992, p. 6). Until the early '90's, the focus of HCI was primarily designing interfaces for single users. In response to a growing concern for the need also to support multiple individuals working together using computer systems, the interdisciplinary field of computer-supported cooperative work (CSCW) emerged (Greif, 1988). Information systems is another area concerned with the application of computing technology in domains like business, health, and education. Other fields related to interaction design include human factors, cognitive ergonomics, and cognitive engineering. All are concerned with designing systems to match users' goals, however, each has a different focus and methodology.
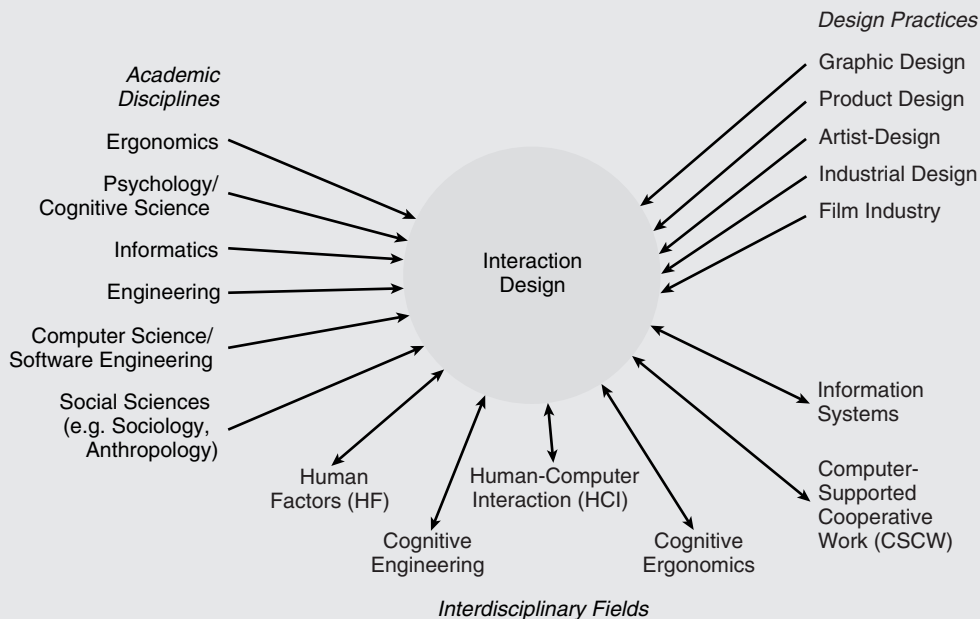

Figure 1.3 Relationship among contributing academic disciplines, design practices, and interdisciplinary fields concerned with interaction design.

In the mid '90s, many companies realized it was necessary again to extend their existing multidisciplinary design teams to include professionals trained in media and design, including graphical design, industrial design, film, and narrative. Sociologists, anthropologists, and dramaturgists were also brought on board, all having quite a different take on human interaction from psychologists. This wider set of

people were thought to have the right mix of skills and understanding of the different application areas necessary to design the new generation of interactive systems. For example, designing a reminder application for the family requires understanding how families interact; creating an interactive story kit for children requires understanding how children write and understand narrative, and developing an interactive guide for art-gallery visitors requires appreciating what people do and how they move through public spaces.

Now in the '00s, the possibilities afforded by emerging hardware capabilities—e.g., radio-frequency tags, large interactive screens, and information appliances—has led to a further realization that engineers, who know about hardware, software, and electronics are needed to configure, assemble, and program the consumer electronics and other devices to be able to communicate with each other (often referred to as middleware).

1.3.2   Working together as a multidisciplinary team

Bringing together so many people with different backgrounds and training has meant many more ideas being generated, new methods being developed, and more creative and original designs being produced. However, the down side is the costs involved. The more people there are with different backgrounds in a design team, the more difficult it can be to communicate and progress forward the designs being generated. Why? People with different backgrounds have different perspectives and ways of seeing and talking about the world (see Figure 1.4). What one person values as important others may not even see (Kim, 1990). Similarly, a computer scientist's understanding of the term representation is often very different from a graphic designer's or a psychologist's.
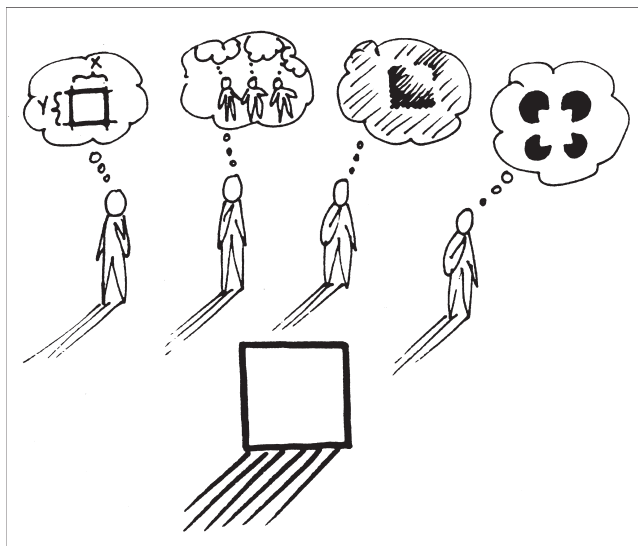

Figure 1.4 Four different team members looking at the same square, but each seeing it quite differently.

What this means in practice is that confusion, misunderstanding, and communication breakdowns can often surface in a team. The various team members may have different ways of talking about design and may use the same terms to mean quite different things. Other problems can arise when a group of people is "thrown" together who have not worked as a team. For example, the Philips Vision of the Future Project found that its multidisciplinary teams—who were responsible for developing ideas and products for the future—experienced a number of difficulties, namely, that project team members did not always have a clear idea of who needed what information, when, and in what form (Lambourne et al., 1997).

ACTIVITY 1.2 In practice, the makeup of a given design team depends on the kind of interactive product being built. Who do you think would need to be involved in developing:

(a) a public kiosk providing information about the exhibits available in a science museum?

(b) an interactive educational website to accompany a TV series?

Comment Each team will need a number of different people with different skill sets. For example, the first interactive product would need:

(a) graphic and interaction designers, museum curators, educational advisors, software engineers, software designers, usability engineers, ergonomists

The second project would need:

(b) TV producers, graphic and interaction designers, teachers, video experts, software engineers, software designers, usability engineers

In addition, as both systems are being developed for use by the general public, representative users, such as school children and parents, should be involved.

In practice, design teams often end up being quite large, especially if they are working on a big project to meet a fixed deadline. For example, it is common to find teams of fifteen people or more working on a website project for an extensive period of time, like six months. This means that a number of people from each area of expertise are likely to be working as part of the project team.

1.3.3 Interaction design in business

Interaction design is now big business. In particular, website consultants, start-up companies, and mobile computing industries have all realized its pivotal role in successful interactive products. To get noticed in the highly competitive field of web products requires standing out. Being able to say that your product is easy and effective to use is seen as central to this. Marketing departments are realizing how branding, the number of hits, customer return rate, and customer satisfaction are greatly affected by the usability of a website. Furthermore, the presence or absence of good interaction design can make or break a company.

One infamous dot.com fashion clothes company that failed to appreciate the importance of good interaction design paid heavily for its oversight, becoming bankrupt within a few months of going public.[2] Their approach had been to go for an "all singing and all dancing," glossy 3D graphical interface. One of the problems with this was that it required several minutes to download. Furthermore, it often took more than 20 minutes to place an order by going through a painfully long and slow process of filling out an online form—only to discover that the order was not successful. Customers simply got frustrated with the site and never returned.

In response to the growing demand for interaction design, an increasing number of consultancies are establishing themselves as interaction design experts. One such company is Swim, set up by Gitta Salomon to assist clients with the design of interactive products (see the interview with her at the end of this chapter). She points out how often companies realize the importance of interaction design but don't know how to do it themselves. So they get in touch with companies, like Swim, with their partially developed products and ask them for help. This can come in the form of an expert "crit" in which a detailed review of the usability and design of the product is given (for more on expert evaluation, see Chapter 13). More extensively, it can involve helping clients create their products.

Another established design company that practices interaction design is IDEO, which now has many branches worldwide. Drawing on over 20 years of experience in the area, they design products, services, and environments for other companies, pioneering new user experiences (Spreenberg et al., 1995). They have developed

---

BOX 1.2 What's In a Name? From Interface Designers to Information Architects

Ten years ago, when a company wanted to develop an interface for an interactive product it advertised for interface designers. Such professionals were primarily involved in the design and evaluation of widgets for desktop applications. Now that the potential range of interactive products has greatly diversified, coupled with the growing realization of the importance of getting the interface right, a number of other job descriptions have begun to emerge. These include:

- interactive/interaction designers (people involved in the design of all the interactive aspects of a product, not just the graphic design of an interface)

- usability engineers (people who focus on evaluating products, using usability methods and principles)

- web designers (people who develop and create the visual design of websites, such as layouts)

- information architects (people who come up with ideas of how to plan and structure interactive products, especially websites)

- user-experience designers (people who do all the above but who may also carry out field studies to inform the design of products)

---

[2]This happened before the dot.com crash in 2001.

Figure 1.5 An innovative product developed by IDEO: Scout Modo, a wireless handheld device delivering up-to-date information about what's going on in a city.

thousands of products for numerous clients, each time following their particular brand of user-centered design (see Figure 1.5).

1.4 What is involved in the process of interaction design?

Essentially, the process of interaction design involves four basic activities:

1. identifying needs and establishing requirements
2. developing alternative designs that meet those requirements
3. building interactive versions of the designs so that they can be communicated and assessed
4. evaluating what is being built throughout the process

These activities are intended to inform one another and to be repeated. For example, measuring the usability of what has been built in terms of whether it is easy to use provides feedback that certain changes must be made or that certain requirements have not yet been met.

Evaluating what has been built is very much at the heart of interaction design. Its focus is on ensuring that the product is usable. It is usually addressed through a user-centered approach to design, which, as the name suggests, seeks to involve users throughout the design process. There are many different ways of achieving this: for example, through observing users, talking to them, interviewing them, testing them using performance tasks, modeling their performance, asking them to fill

in questionnaires, and even asking them to become co-designers. The findings from the different ways of engaging and eliciting knowledge from users are then interpreted with respect to ongoing design activities (we give more detail about all these aspects of evaluation in Chapters 10–14).

Equally important as involving users in evaluating an interactive product is understanding what people currently do. This form of research should take place before building any interactive product. Chapters 3, 4, and 5 cover a lot of this ground by explaining in detail how people act and interact with one another, with information, and with various technologies, together with describing their strengths and weaknesses. Such knowledge can greatly help designers determine which solutions to choose from the many design alternatives available and how to develop and test these further. Chapter 7 describes how an understanding of users' needs can be translated to requirements, while Chapter 9 explains how to involve users effectively in the design process.

A main reason for having a better understanding of users is that different users have different needs and interactive products need to be designed accordingly. For example, children have different expectations about how they want to learn or play from adults. They may find having interactive quizzes and cartoon characters helping them along to be highly motivating, whereas most adults find them annoying. Conversely, adults often like talking-heads discussions about topics, but children find them boring. Just as everyday objects like clothes, food, and games are designed differently for children, teenagers, and adults, so, too, must interactive products be designed to match the needs of different kinds of users.

In addition to the four basic activities of design, there are three key characteristics of the interaction design process:

1. Users should be involved through the development of the project.
2. Specific usability and user experience goals should be identified, clearly documented, and agreed upon at the beginning of the project.
3. Iteration through the four activities is inevitable.

We have already mentioned the importance of involving users and will return to this topic throughout the book. Iterative design will also be addressed later when we talk about the various design and evaluation methods by which this can be achieved. In the next section we describe usability and user experience goals.

1.5   The goals of interaction design

Part of the process of understanding users' needs, with respect to designing an interactive system to support them, is to be clear about your primary objective. Is it to design a very efficient system that will allow users to be highly productive in their work, or is it to design a system that will be challenging and motivating so that it supports effective learning, or is it something else? We call these top-level concerns usability goals and user experience goals. The two differ in terms of how they are operationalized, i.e., how they can be met and through what means. Usability

goals are concerned with meeting specific usability criteria (e.g., efficiency) and user experience goals are concerned with explicating the quality of the user experience (e.g., to be aesthetically pleasing).

1.5.1   Usability goals

To recap, usability is generally regarded as ensuring that interactive products are easy to learn, effective to use, and enjoyable from the user's perspective. It involves optimizing the interactions people have with interactive products to enable them to carry out their activities at work, school, and in their everyday life. More specifically, usability is broken down into the following goals:

- effective to use (effectiveness)
- efficient to use (efficiency)
- safe to use (safety)
- have good utility (utility)
- easy to learn (learnability)
- easy to remember how to use (memorability)

For each goal, we describe it in more detail and provide a key question.

Effectiveness is a very general goal and refers to how good a system is at doing what it is supposed to do.

Question: Is the system capable of allowing people to learn well, carry out their work efficiently, access the information they need, buy the goods they want, and so on?

Efficiency refers to the way a system supports users in carrying out their tasks. The answering machine described at the beginning of the chapter was considered efficient in that it let the user carry out common tasks (e.g., listening to messages) through a minimal number of steps. In contrast, the voice-mail system was considered inefficient because it required the user to carry out many steps and learn an arbitrary set of sequences for the same common task. This implies that an efficient way of supporting common tasks is to let the user use single button or key presses. An example of where this kind of efficiency mechanism has been effectively employed is in e-tailing. Once users have entered all the necessary personal details on an e-commerce site to make a purchase, they can let the site save all their personal details. Then, if they want to make another purchase at that site, they don't have to re-enter all their personal details again. A clever mechanism patented by Amazon.com is the one-click option, which requires users only to click a single button when they want to make another purchase.

Question: Once users have learned how to use a system to carry out their tasks, can they sustain a high level of productivity?

Safety involves protecting the user from dangerous conditions and undesirable situations. In relation to the first ergonomic aspect, it refers to the external conditions where people work. For example, where there are hazardous conditions—like X-ray machines or chemical plants—operators should be able to interact with and control computer-based systems remotely. The second aspect refers to helping any

kind of user in any kind of situation avoid the dangers of carrying out unwanted actions accidentally. It also refers to the perceived fears users might have of the consequences of making errors and how this affects their behavior. To make computer-based systems safer in this sense involves (i) preventing the user from making serious errors by reducing the risk of wrong keys/buttons being mistakenly activated (an example is *not* placing the quit or delete-file command right next to the save command on a menu) and (ii) providing users with various means of recovery should they make errors. Safe interactive systems should engender confidence and allow the user the opportunity to explore the interface to try out new operations (see Figure 1.6a). Other safety mechanisms include undo facilities and
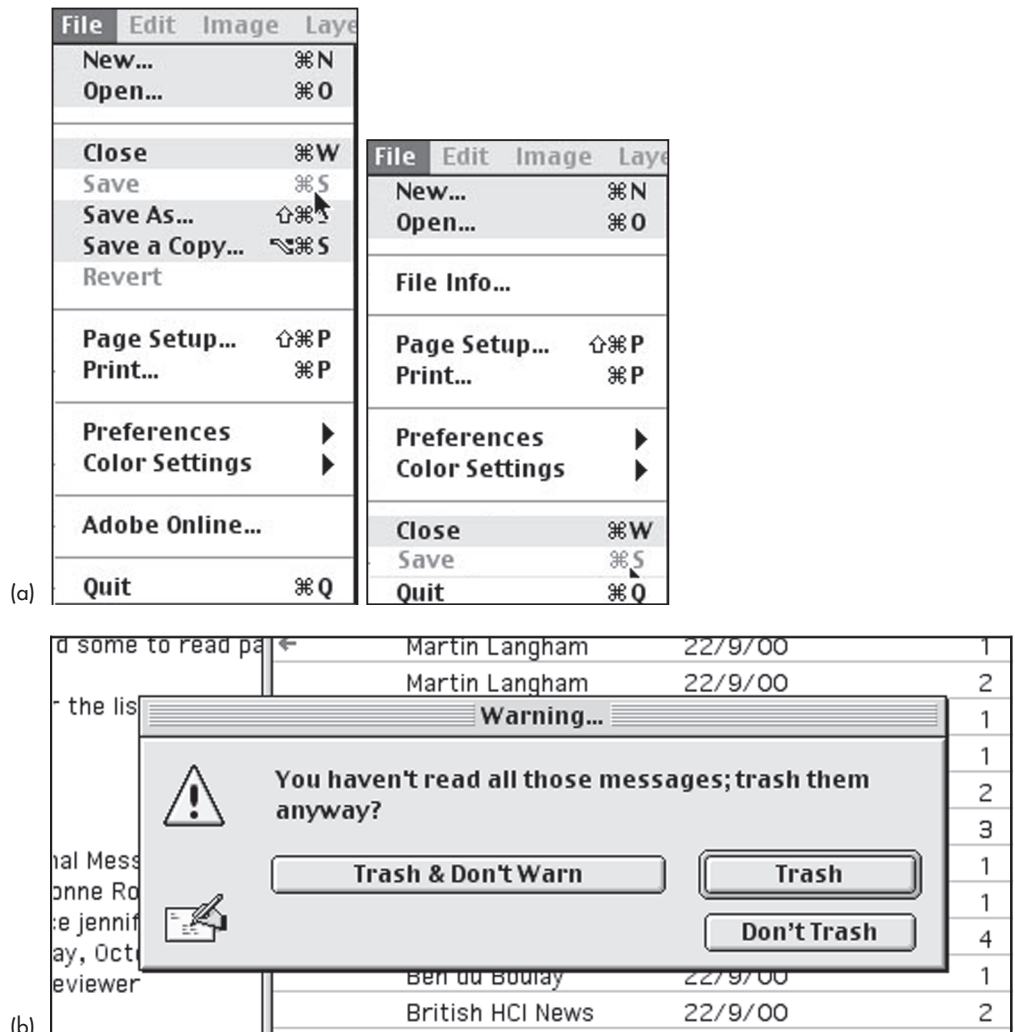

Figure 1.6 (a) A safe and an unsafe menu. Which is which and why? (b) Warning dialog message from Eudora.

confirmatory dialog boxes that give users another chance to consider their intentions (a well-known example used in e-mail applications is the appearance of a dialog box, after the user has highlighted messages to be deleted, saying: "Are you sure you want to delete all these messages?" See Figure 1.6(b)).

Question: Does the system prevent users from making serious errors and, if they do make an error, does it permit them to recover easily?

Utility refers to the extent to which the system provides the right kind of functionality so that users can do what they need or want to do. An example of a system with high utility is an accounting software package providing a powerful computational tool that accountants can use to work out tax returns. A example of a system with low utility is a software drawing tool that does not allow users to draw freehand but forces them to use a mouse to create their drawings, using only polygon shapes.

Question: Does the system provide an appropriate set of functions that enable users to carry out all their tasks in the way they want to do them?

Learnability refers to how easy a system is to learn to use. It is well known that people don't like spending a long time learning how to use a system. They want to get started straight away and become competent at carrying out tasks without too much effort. This is especially so for interactive products intended for everyday use (e.g., interactive TV, email) and those used only infrequently (e.g., videoconferencing). To a certain extent, people are prepared to spend longer learning more complex systems that provide a wider range of functionality (e.g., web authoring tools, word processors). In these situations, CD-ROM and online tutorials can help by providing interactive step-by-step material with hands-on exercises. However, many people find these tedious and often difficult to relate to the tasks they want to

---

BOX 1.3 | The Ten-Minute Rule

A criterion for assessing whether a system is easy to learn is to apply the "ten-minute rule" (Nelson, 1980). It proposes that novice users should be able to learn how to use a system in under 10 minutes. If not the system fails. As pointed out by Rubinstein and Hersh (1984), many computer systems do not meet this criterion. To make systems easy to learn, they suggest that designers capitalize on people's existing knowledge: "A computer system for architects is not expected to teach architecture. Quite the reverse: the ten-minute rule requires that what an architect already knows be helpful in learning to use the architecture system," (Rubinstein and Hersh, 1984 p. 9).

When is the ten-minute rule not appropriate?

The ten-minute rule is a useful rule of thumb for evaluating many kinds of systems. However, it is inappropriate for using with complex systems, where it would be difficult and reckless to think that a user could learn how to use them in under ten minutes. For example, would you feel safe knowing that the pilots flying your plane had only spent ten minutes learning how to use the various devices in the cockpit? You would expect them to have spent considerable time (in addition to the years of training to become a pilot) thoroughly learning how to use the array of controls and displays for that particular kind of plane and what to do if any of them malfunction. Likewise, it is unrealistic to assume that ten minutes is enough to learn a system that provides diverse functionality (e.g., a word processor) or that needs high levels of skill to use (e.g., a video game).

accomplish. A key concern is determining how much time users are prepared to spend learning a system. There seems little point in developing a range of functionality if the majority of users are unable or not prepared to spend time learning how to use it.

Question: How easy is it and how long does it take (i) to get started using a system to perform core tasks and (ii) to learn the range of operations to perform a wider set of tasks?

Memorability refers to how easy a system is to remember how to use, once learned. This is especially important for interactive systems that are used infrequently. If users haven't used a system or an operation for a few months or longer, they should be able to remember or at least rapidly be reminded how to use it. Users shouldn't have to keep relearning how to carry out tasks. Unfortunately, this tends to happen when the operations required to be learned are obscure, illogical, or poorly sequenced. Users need to be helped to remember how to do tasks. There are many ways of designing the interaction to support this. For example, users can be helped to remember the sequence of operations at different stages of a task through meaningful icons, command names, and menu options. Also, structuring options and icons so they are placed in relevant categories of options (e.g., placing all the drawing tools in the same place on the screen) can help the user remember where to look to find a particular tool at a given stage of a task.

Question: What kinds of interface support have been provided to help users remember how to carry out tasks, especially for systems and operations that are used infrequently?

ACTIVITY 1.3   How long do you think it *should* take to learn how to use the following interactive products and how long does it *actually* take most people to learn them? How memorable are they?

    (a) using a VCR to play a video

    (b) using a VCR to pre-record two programs

    (c) using an authoring tool to create a website

Comment   (a) To play a video should be as simple as turning the radio on, should take less than 30 seconds to work out, and then should be straightforward to do subsequently. Most people are able to fathom how to play a video. However, some systems require the user to switch to the "video" channel using one or two remote control devices, selecting from a choice of 50 or more channels. Other settings may also need to be configured before the video will play. Most people are able to remember how to play a video once they have used a particular VCR.

    (b) This is a more complex operation and should take a couple of minutes to learn how to do and to check that the programming is correct. In reality, many VCRs are so poorly designed that 80% of the population is unable to accomplish this task, despite several attempts. Very few people remember how to pre-record a program, largely because the interaction required to do this is poorly designed, with poor or no feedback, and is often illogical from the user's perspective. Of those, only a few will bother to go through the manual again.

(c) A well-designed authoring tool should let the user create a basic page in about 20 minutes. Learning the full range of operations and possibilities is likely to take much longer, possibly a few days. In reality, there are some good authoring tools that allow the user to get started straight away, providing templates that they can adapt. Most users will extend their repertoire, taking another hour or so to learn more functions. However, very few people actually learn to use the full range of functions provided by the authoring tool. Users will tend to remember frequently used operations (e.g., cut and paste, inserting images), especially if they are consistent with the way they are carried out in other software applications. However, less frequently used operations may need to be relearned (e.g., formatting tables).

The usability goals discussed so far are well suited to the design of business systems intended to support working practices. In particular, they are highly relevant for companies and organizations who are introducing or updating applications running on desktop and networked systems—that are intended to increase productivity by improving and enhancing how work gets done. As well as couching them in terms of specific questions, usability goals are turned into *usability criteria*. These are specific objectives that enable the usability of a product to be assessed in terms of how it can improve (or not) a user's performance. Examples of commonly used usability criteria are time to complete a task (efficiency), time to learn a task (learnability), and the number of errors made when carrying out a given task over time (memorability).

1.5.2   User experience goals

The realization that new technologies are offering increasing opportunities for supporting people in their everyday lives has led researchers and practitioners to consider further goals. The emergence of technologies (e.g., virtual reality, the web, mobile computing) in a diversity of application areas (e.g., entertainment, education, home, public areas) has brought about a much wider set of concerns. Instead of focusing primarily on improving efficiency and productivity at work, interaction design is increasingly concerning itself with creating systems that are:

- satisfying
- enjoyable
- fun
- entertaining
- helpful
- motivating
- aesthetically pleasing
- supportive of creativity
- rewarding
- emotionally fulfilling

The goals of designing interactive products to be fun, enjoyable, pleasurable, aesthetically pleasing and so on are concerned primarily with the user experience. By this we mean what the interaction with the system *feels* like to the users. This involves explicating the nature of the user experience in subjective terms. For example, a new software package for children to create their own music may be designed with the primary objectives of being fun and entertaining. Hence, user experience goals differ from the more objective usability goals in that they are concerned with how users experience an interactive product from their perspective, rather than assessing how useful or productive a system is from its own perspective. The relationship between the two is shown in Figure 1.7.

Much of the work on enjoyment, fun, etc., has been carried out in the entertainment and computer games industry, which has a vested interest in understanding the role of pleasure in considerable detail. Aspects that have been described as contributing to pleasure include: attention, pace, play, interactivity, conscious and unconscious control, engagement, and style of narrative. It has even been suggested that in these pleasure contexts, it might be interesting to build systems that are *non-easy* to use, providing opportunities for quite different user experiences from those designed based on usability goals (Frohlich and Murphy, 1999). Interacting with a virtual representation using a physical device (e.g., banging a plastic
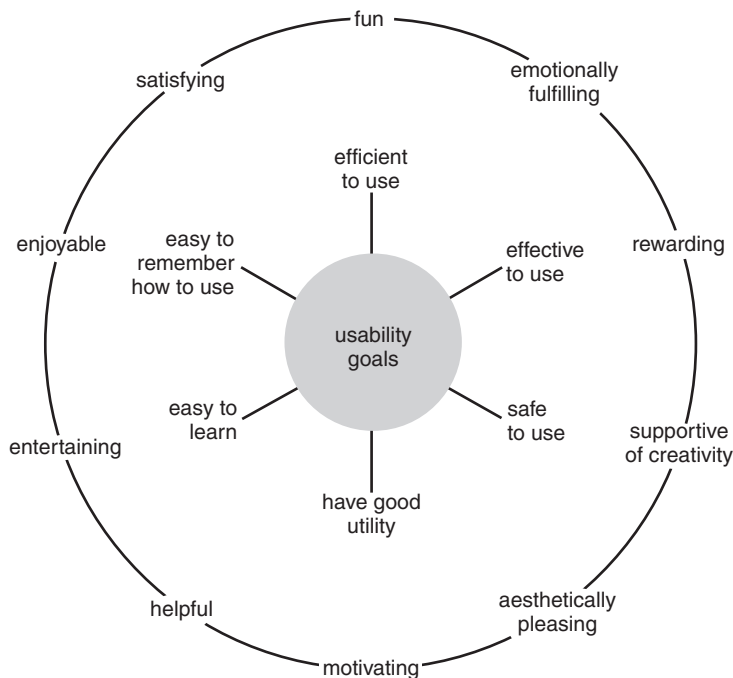

Figure 1.7 Usability and user experience goals. Usability goals are central to interaction design and are operationalized through specific criteria. User experience goals are shown in the outer circle and are less clearly defined.

hammer to hit a virtual nail represented on the computer screen) compared with using a more efficient way to do the same thing (e.g., selecting an option using command keys) may require *more effort* but could, conversely, result in a *more enjoyable* and *fun* experience.

Recognizing and understanding the trade-offs between usability and user experience goals is important. In particular, this enables designers to become aware of the consequences of pursuing different combinations of them in relation to fulfilling different users' needs. Obviously, not all of the usability goals and user experience goals apply to every interactive product being developed. Some combinations will also be incompatible. For example, it may not be possible or desirable to design a process control system that is both safe and fun. As stressed throughout this chapter, what is important depends on the use context, the task at hand, and who the intended users are.

ACTIVITY1.4   Below are a number of proposed interactive products. What do you think are the key usability goals and user experience goals for each of them?

(a) a mobile device that allows young children to communicate with each other and play collaborative games

(b) a video and computer conferencing system that allows students to learn at home

(c) an Internet application that allows the general public to access their medical records via interactive TV

(d) a CAD system for architects and engineers

(e) an online community that provides support for people who have recently been bereaved

Comment   (a) Such a collaborative device should be easy to use, effective, efficient, easy to learn and use, fun and entertaining.

(b) Such a learning device should be easy to learn, easy to use, effective, motivating and rewarding.

(c) Such a personal system needs to be safe, easy to use and remember how to use, efficient and effective.

(d) Such a tool needs to be easy to learn, easy to remember, have good utility, be safe, efficient, effective, support creativity and be aesthetically pleasing.

(e) Such a system needs to be easy to learn, easy to use, motivating, emotionally satisfying and rewarding.

1.6 More on usability: design and usability principles

Another way of conceptualizing usability is in terms of design principles. These are generalizable abstractions intended to orient designers towards thinking about different aspects of their designs. A well-known example is feedback: systems should be designed to provide adequate feedback to the users to ensure they know what to

do next in their tasks. Design principles are derived from a mix of theory-based knowledge, experience, and common sense. They tend to be written in a prescriptive manner, suggesting to designers what to provide and what to avoid at the interface—if you like, the do's and don'ts of interaction design. More specifically, they are intended to help designers explain and improve the design (Thimbleby, 1990). However, they are not intended to specify how to design an actual interface (e.g., telling the designer how to design a particular icon or how to structure a web portal) but act more like a set of reminders to designers, ensuring that they have provided certain things at the interface.

A number of design principles have been promoted. The best known are concerned with how to determine what users should see and do when carrying out their tasks using an interactive product. Here we briefly describe the most common ones: visibility, feedback, constraints, mapping, consistency, and affordances. Each of these has been written about extensively by Don Norman (1988) in his bestseller *The Design of Everyday Things.*

Visibility   The importance of visibility is exemplified by our two contrasting examples at the beginning of the chapter. The voice-mail system made the presence and number of waiting messages invisible, while the answer machine made both aspects highly visible. The more visible functions are, the more likely users will be able to know what to do next. In contrast, when functions are "out of sight," it makes them more difficult to find and know how to use. Norman (1988) describes the controls of a car to emphasize this point. The controls for different operations are clearly visible (e.g., indicators, headlights, horn, hazard warning lights), indicating what can be done. The relationship between the way the controls have been positioned in the car and what they do makes it easy for the driver to find the appropriate control for the task at hand.

Feedback   Related to the concept of visibility is feedback. This is best illustrated by an analogy to what everyday life would be like without it. Imagine trying to play a guitar, slice bread using a knife, or write using a pen if none of the actions produced any effect for several seconds. There would be an unbearable delay before the music was produced, the bread was cut, or the words appeared on the paper, making it almost impossible for the person to continue with the next strum, saw, or stroke.

Feedback is about sending back information about what action has been done and what has been accomplished, allowing the person to continue with the activity. Various kinds of feedback are available for interaction design—audio, tactile, verbal, visual, and combinations of these. Deciding which combinations are appropriate for different kinds of activities and interactivities is central. Using feedback in the right way can also provide the necessary visibility for user interaction.

Constraints   The design concept of constraining refers to determining ways of restricting the kind of user interaction that can take place at a given moment. There are various ways this can be achieved. A common design practice in graphical user interfaces is to deactivate certain menu options by shading them, thereby restrict-
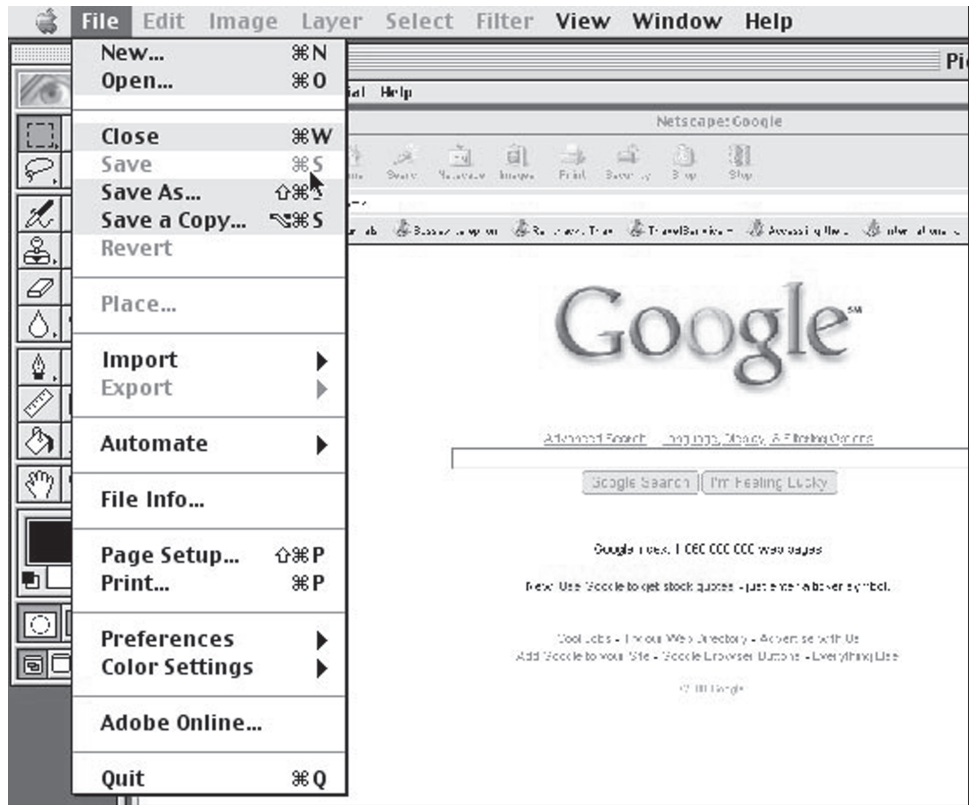
Figure 1.8 A menu illustrating restricted availability of options as an example of logical constraining. Shaded areas indicate deactivated options.

ing the user to only actions permissible at that stage of the activity (see Figure 1.8). One of the advantages of this form of constraining is it prevents the user from selecting incorrect options and thereby reduces the chance of making a mistake. The use of different kinds of graphical representations can also constrain a person's interpretation of a problem or information space. For example, flow chart diagrams show which objects are related to which, thereby constraining the way the information can be perceived.

Norman (1999) classifies constraints into three categories: physical, logical, and cultural. Physical constraints refer to the way physical objects restrict the movement of things. For example, the way an external disk can be placed into a disk drive is physically constrained by its shape and size, so that it can be inserted in only one way. Likewise, keys on a pad can usually be pressed in only one way.

Logical constraints rely on people's understanding of the way the world works (cf. the marbles answering machine design). They rely on people's common-sense reasoning about actions and their consequences. Picking up a physical marble and placing it in another location on the phone would be expected by most people to

Figure 1.9 (a) Natural mapping between rewind, play, and fast forward on a tape recorder device. (b) An alternative arbitrary mapping.

trigger something else to happen. Making actions and their effects obvious enables people to logically deduce what further actions are required. Disabling menu options when not appropriate for the task in hand provides logical constraining. It allows users to reason why (or why not) they have been designed this way and what options are available.

Cultural constraints rely on learned conventions, like the use of red for warning, the use of certain kinds of audio signals for danger, and the use of the smiley face to represent happy emotions. Most cultural constraints are arbitrary in the sense that their relationship with what is being represented is abstract, and could have equally evolved to be represented in another form (e.g., the use of yellow instead of red for warning). Accordingly, they have to be learned. Once learned and accepted by a cultural group, they become universally accepted conventions. Two universally accepted interface conventions are the use of windowing for displaying information and the use of icons on the desktop to represent operations and documents.

Mapping   This refers to the relationship between controls and their effects in the world. Nearly all artifacts need some kind of mapping between controls and effects, whether it is a flashlight, car, power plant, or cockpit. An example of a good mapping between control and effect is the up and down arrows used to represent the up and down movement of the cursor, respectively, on a computer keyboard. The mapping of the relative position of controls and their effects is also important. Consider the various musical playing devices (e.g., MP3, CD player, tape recorder). How are the controls of playing, rewinding, and fast forward mapped onto the desired effects? They usually follow a common convention of providing a sequence of buttons, with the play button in the middle, the rewind button on the left and the fast-forward on the right. This configuration maps directly onto the directionality of the actions (see Figure 1.9a). Imagine how difficult it would be if the mappings in Figure 1.9b were used. Look at Figure 1.10 and determine from the various mappings which is good and which would cause problems to the person using it.

Figure 1.10 Four possible combinations of arrow-key mappings. Which is the most natural mapping?

Consistency   This refers to designing interfaces to have similar operations and use similar elements for achieving similar tasks. In particular, a consistent interface is one that follows rules, such as using the same operation to select all objects. For example, a consistent operation is using the same input action to highlight any graphical object at the interface, such as always clicking the left mouse button. Inconsistent interfaces, on the other hand, allow exceptions to a rule. An example of this is where certain graphical objects (e.g., email messages presented in a table) can be highlighted only by using the right mouse button, while all other operations are highlighted using the left button. A problem with this kind of inconsistency is that it is quite arbitrary, making it difficult for users to remember and making the users more prone to mistakes.

One of the benefits of consistent interfaces, therefore, is that they are easier to learn and use. Users have to learn only a single mode of operation that is applicable to all objects. This principle works well for simple interfaces with limited operations, like a mini CD player with a small number of operations mapped onto separate buttons. Here, all the user has to do is learn what each button represents and select accordingly. However, it can be more problematic to apply the concept of consistency to more complex interfaces, especially when many different operations need to be designed for. For example, consider how to design an interface for an application that offers hundreds of operations (e.g. a word-processing application). There is simply not enough space for a thousand buttons, each of which maps onto an individual operation. Even if there were, it would be extremely difficult and time-consuming for the user to search through them all to find the desired operation.

A much more effective design solution is to create categories of commands that can be mapped into subsets of operations. For the word-processing application, the hundreds of operations available are categorized into subsets of different menus. All commands that are concerned with file operations (e.g., save, open, close) are placed together in the same file menu. Likewise, all commands concerned with formatting text are placed in a format menu. Selecting an operation then becomes a matter of homing in on the right category (menu) of options and scanning it for the desired one, rather than scrolling through one long list. However, the consistency rule of having a visible one-to-one mapping between command and operation is broken. Operations are not immediately visible at the interface, but are instead hidden under different categories of menus. Furthermore, some menu items are immediately visible, when a top-level menu is first pulled down, while others remain hidden until the visible items are scrolled over. Thus, users need to learn what items are visible in each menu category and which are hidden in submenus.

The way the items are divided between the categories of menu items can also appear inconsistent to users. Various operations appear in menus where they do not belong. For example, the sorting operation (very useful for listing references or names in alphabetical order) in Microsoft Word 2001 is in the Table menu (the Mac Version). In the previous Word 98 version, it was in both the Tools and Table menus. I always thought of it as a Tool operation (like Word Count), and became very frustrated to discover that as a default for Word 2001 it is only in the Table menu. This makes it inconsistent for me in two ways: (i) with the previous version and (ii) in the category it has been placed. Of course, I can customize the new ver-

sion so that the menus are structured in the way I think they should be, but this all takes considerable time (especially when I use different machines at work, home, and when travelling).

Another problem with consistency is determining what aspect of an interface to make consistent with what else. There are often many choices, some of which can be inconsistent with other aspects of the interface or ways of carrying out actions. Consider the design problem of developing a mechanism to let users lock their files on a shared server. Should the designer try to design it to be consistent with the way people lock things in the outside world (called external consistency) or with the way they lock objects in the existing system (called internal consistency)? However, there are many different ways of locking objects in the physical world (e.g., placing in a safe, using a padlock, using a key, using a child safety lock), just as there are different ways of locking electronically (e.g., using PIN numbers, passwords, permissions, moving the physical switches on floppy disks). The problem facing designers is knowing which one to be consistent with.

Affordance   is a term used to refer to an attribute of an object that allows people to know how to use it. For example, a mouse button invites pushing (in so doing activating clicking) by the way it is physically constrained in its plastic shell. At a very simple level, to afford means "to give a clue" (Norman, 1988). When the affordances of a physical object are perceptually obvious it is easy to know how to interact with it. For example, a door handle affords pulling, a cup handle affords grasping, and a mouse button affords pushing. Norman introduced this concept in the late '80s in his discussion of the design of everyday objects. Since then, it has been much popularized, being used to describe how interface objects should be designed so that they make obvious what can be done to them. For example, graphical elements like buttons, icons, links, and scroll bars are talked about with respect to how to make it appear obvious how they should be used: icons should be designed to afford clicking, scroll bars to afford moving up and down, buttons to afford pushing.

Unfortunately, the term affordance has become rather a catch-all phrase, losing much of its potency as a design principle. Norman (1999), who was largely responsible for originally promoting the concept in his book *The Design of Everyday Things* (1988), now despairs at the way it has come to be used in common parlance:

> *"I put an affordance there," a participant would say, "I wonder if the object affords clicking . . . " affordances this, affordances that. And no data, just opinion. Yikes! What had I unleashed upon the world?* Norman's (1999) reaction to a recent CHI-Web discussion.

He has since tried to clarify his argument about the utility of the concept by saying there are two kinds of affordance: perceived and real. Physical objects are said to have real affordances, like grasping, that are perceptually obvious and do not have to be learned. In contrast, user interfaces that are screen-based are virtual and do not have these kinds of real affordances. Using this distinction, he argues that it does not make sense to try to design for real affordances at the interface—except when designing physical devices, like control consoles, where affordances like pulling and pressing are helpful in guiding the user to know what to do. Alternatively, screen-based

---

BOX 1.4 Can You Afford a Screen?

A problem in applying the concept of affordance to interface objects is that virtual objects have quite different properties from physical objects. A physical door handle affords pulling because its physical properties constrain what can be done with it in relation to the person and environment. It results in it opening (if it is closed) and it closing (if it is open). It is obvious to the person how to interact with it. However, a virtual object like an icon button invites clicking on only because a user has learned initially that the graphical element on the screen is a representation that, when clicked on, makes something else happen (such as moving to another page). It could equally trigger other system responses, like a window closing down. Hence, the mapping between a virtual representation and its behavior is arbitrary, relying on the user learning the accepted conventions.

A problem in using the concept of affordance in this context is that it can be misleading. Designers can be misled into thinking that virtual objects should be designed to look and behave like physical objects because people know intuitively how to interact with these. This may lead them into thinking that interfaces that exhibit this kind of realism will be easier to learn and use. These assumptions, however, are incorrect for the reason stated above. To illustrate this point further, consider the design of screen buttons. A number of them have been designed to have a 3D look, giving the appearance of protruding. An assumption is that this kind of illusion will give the buttons the *affordance* of pushing, inviting the user to click on them, as they would do with actual physical buttons. While users may readily learn this association, it is equally the case that they will be able to learn how to interact with a simple, 2D representation of a button on the screen. The effort to learn the association is similar. However, the effort to design 3D buttons is likely to be greater than simple 2D buttons.

A danger with trying to design graphical interfaces to afford in the way physical objects do is that it can inadvertently lead to poor design. The use of shadowing and other perceptual illusions to give the effect of 3D can have the undesirable effect of cluttering up an interface, often making it more difficult to find particular objects. Simple, plain 2D abstract shapes (e.g., a square or a circle) used to represent objects like buttons, on the other hand, can be easier to perceive and recognize at the interface (See Figure 1.11 on Color Plate 1).

---

interfaces are better conceptualized as *perceived* affordances, which are essentially learned conventions. In conclusion, Norman argues that other design concepts—conventions, feedback and cultural and logical constraints—are far more useful for helping designers develop graphical user interfaces.

1.6.1   Heuristics and usability principles

When design principles are used in practice they are commonly referred to as heuristics. This term emphasizes that something has to be done with them when they are applied to a given problem. In particular, they need to be interpreted in the design context, drawing on past experience of, for example, how to design feedback and what it means for something to be consistent.

Another form of guidance is usability principles. An example is "speak the user's language." These are quite similar to design principles, except that they tend to be more prescriptive. In addition, whereas design principles tend to be used mainly for informing a design, usability principles are used mostly as the basis for evaluating prototypes and existing systems. In particular, they provide the framework for heuristic evaluation (see Chapter 13). They, too, are called heuristics when used as part of

an evaluation. Below are the ten main usability principles, developed by Nielsen (2001) and his colleagues. Note how some of them overlap with the design principles.

1. **Visibility of system status**—always keep users informed about what is going on, through providing appropriate feedback within reasonable time

2. **Match between system and the real world**—speak the users' language, using words, phrases and concepts familiar to the user, rather than system-oriented terms

3. **User control and freedom**—provide ways of allowing users to easily escape from places they unexpectedly find themselves, by using clearly marked 'emergency exits'

4. **Consistency and standards**—avoid making users wonder whether different words, situations, or actions mean the same thing

5. **Help users recognize, diagnose, and recover from errors**—use plain language to describe the nature of the problem and suggest a way of solving it

6. **Error prevention**—where possible prevent errors occurring in the first place

7. **Recognition rather than recall**—make objects, actions, and options visible

8. **Flexibility and efficiency of use**—provide accelerators that are invisible to novice users, but allow more experienced users to carry out tasks more quickly

9. **Aesthetic and minimalist design**—avoid using information that is irrelevant or rarely needed

10. **Help and documentation**—provide information that can be easily searched and provides help in a set of concrete steps that can easily be followed

ACTIVITY 1.5 One of the main design principles which Nielsen has proselytized, especially for website design, is simplicity. He proposes that designers go through all of their design elements and remove them one by one. If a design works just as well without an element, then remove it. Do you think this is a good design principle? If you have your own website, try doing this and seeing what happens. At what point does the interaction break down?

Comment Simplicity is certainly an important design principle. Many designers try to cram too much into a screenful of space, making it unwieldy for people to find what they are interested in. Removing design elements to see what can be discarded without affecting the overall function of the website can be a salutary lesson. Unnecessary icons, buttons, boxes, lines, graphics, shading, and text can be stripped, leaving a cleaner, crisper, and easier-to-navigate website. However, a certain amount of graphics, shading, coloring, and formatting can make a site aesthetically pleasing and enjoyable to use. Plain vanilla sites with just lists of text and a few hyperlinks may not be as appealing and may put certain visitors off returning. The key is getting the right balance between aesthetic appeal and the right amount and kind of information per page.

Design and usability principles have also been operationalized into even more specific prescriptions called rules. These are guidelines that should be followed. An example is "always place the quit or exit button at the bottom of the first menu list in an application."

BOX 1.5　Usable Usability: Which Terms Do I Use?

The various terms proposed for describing the different aspects of usability can be confusing. They are often used interchangeably and in different combinations. Some people talk about usability design principles, others usability heuristics, and others design concepts. The key is understanding how to use the different levels of guidance. Guidelines is the most general term used to refer to *all forms* of guidance. Goals refer to the high-level usability aims of the system (e.g., it should be efficient to use). Principles refer to general guidance intended to inform the design and evaluation of a system. Rules are low-level guidance that refer to a particular prescription that must be followed. Heuristics is a general term used to refer to design and usability principles when applied to a particular design problem.

Concept	Level of guidance	Also sometimes called	How To Use
Usability goals	General		Setting up usability criteria for assessing the acceptability of a system (e.g., "How long does it take to perform a task?").
User experience goals	General	Pleasure factors	Identifying the important aspects of the user experience (e.g., "How can you make the interactive product fun and enjoyable?").
Design principles	General	Heuristics when used in practice. Design concepts	As reminders of what to provide and what to avoid when designing an interface (e.g., "What kind of feedback are you going to provide at the interface?").
Usability principles	Specific	Heuristics when used in practice	Assessing the acceptability of interfaces, used during heuristic evaluation (e.g., "Does the system provide clearly marked exits?").
Rules	Specific		To determine if an interface adheres to a specific rule when being designed and evaluated (e.g., "Always provide a backwards and forwards navigation button on a web browser").

Assignment

This assignment is intended for you to put into practice what you have read about in this chapter. Specifically, the objective is to enable you to define usability and user experience goals and to use design and usability principles for evaluating the usability of an interactive product.

Find a handheld device (e.g. remote control, handheld computer, or cell phone) and examine how it has been designed, paying particular attention to how the user is meant to interact with it.

(a) From your first impressions, write down what first comes to mind as to what is good and bad about the way the device works. Then list (i) its functionality and (ii) the range of tasks a typical user would want to do using it. Is the functionality greater, equal, or less than what the user wants to do?

(b) Based on your reading of this chapter and any other material you have come across, compile your own set of usability and user experience goals that you think will be

DILEMMA   Usability Trade-Offs

One of the problems of applying more than one of the design principles in interaction design is that trade-offs can arise between them. For example, the more you try to constrain an interface, the less visible information becomes. The same can also happen when trying to apply a single design principle. For example, we saw how the more an interface is designed to "afford" through trying to resemble the way physical objects look, the more cluttered and difficult to use it can become. Consistency is another design principle that can be problematic to apply. As we saw earlier, trying to design an interface to be consistent with something can make it inconsistent with something else. Furthermore, sometimes inconsistent interfaces are actually easier to use than consistent interfaces. A trade-off, however, is that it can take longer to learn such an interface.

Grudin (1989) illustrates the consistency dilemma with an analogy to where knives are stored in a house. Knifes have a variety of forms—butter knives, steak knives, table knifes, fish knives. An easy place to put them all and subsequently locate them is in the top drawer by the sink. This makes it easy for everyone to find them and follows a simple consistent rule. But what about the knives that don't fit or are too sharp to put in the drawer, like carving knives and bread knives? They are placed in a wooden block. And what about the best knives kept only for special occasions? Another exception, as they are placed in the cabinet in the other room for safekeeping. And what about other knives like putty knives and paint-scraping knives used in home projects (kept in the garage) and jack knives (kept in one's pockets or backpack)? Very quickly the consistency rule begins to break down.

Grudin notes how in extending the number of places where knives are kept inconsistency is introduced, which in turn increases the time needed *to learn* where they are all stored. However, the placement of the knives in different places often makes it *easier* to find them because they are at hand for the context in which they are used and also next to the other objects used for a specific task (e.g. all the home project tools are stored together in a box in the garage). The same is true when designing interfaces: introducing inconsistency can make it more difficult to learn an interface but in the long run can make it easier to use.

        most useful in evaluating the device. Decide which are the most important ones and explain why.

(c) Translate the core usability and user experience goals you have selected into two or three questions. Then use them to assess how well your device fares (e.g., *Usability goals.* What specific mechanisms have been used to ensure safety? How easy is it to learn? *User experience goals:* Is it fun to use? Does the user get frustrated easily? If so, why?).

(d) Repeat (b) and (c) for design concepts and usability principles (again choose a relevant set).

(e) Finally, discuss possible improvements to the interface based on your usability evaluation.

Summary

In this chapter we have looked at what interaction design is and how it has evolved. We examined briefly its makeup and the various processes involved. We pointed out how the notion of usability is fundamental to interaction design. This was explained in some detail, describing what it is and how it is operationalized to assess the appropriateness, effectiveness, and quality of interactive products. A number of high-level design principles were also introduced that provide different forms of guidance for interaction design.

Key points

- Interaction design is concerned with designing interactive products to support people in their everyday and working lives.
- Interaction design is multidisciplinary, involving many inputs from wide-reaching disciplines and fields.
- Interaction design is now big business: many companies want it but don't know how to do it.
- Optimizing the interaction between users and interactive products requires taking into account a number of interdependent factors, including context of use, type of task, and kind of user.
- Interactive products need to be designed to match usability goals like ease of use and learning.
- User experience goals are concerned with creating systems that enhance the user experience in terms of making it enjoyable, fun, helpful, motivating, and pleasurable.
- Design and usability principles, like feedback and simplicity, are useful heuristics for analyzing and evaluating aspects of an interactive product.

Further reading

Here we recommend a few seminal readings. A more comprehensive list of useful books, articles, websites, videos, and other material can be found at our website.

WINOGRAD, T. (1997) From computing machinery to interaction design. In P. Denning and R. Metcalfe (eds.) *Beyond Calculation: the Next Fifty Years of Computing*. New York: Springer-Verlag, 149–162. Terry Winograd provides an overview of how interaction design has emerged as a new area, explaining how it does not fit into any existing design or computing fields. He describes the new demands and challenges facing the profession.

NORMAN, D. *The Design of Everyday Things*. New York: Doubleday, 1988 (especially Chapter 1). Norman's writing is highly accessible and enjoyable to read. He writes extensively about the design and usability of everyday objects like doors, faucets, and fridges. These examples provide much food for thought in relation to designing interfaces. The Voyager CD-ROM (sadly, now no longer published) of his collected works provides additional videos and animations that illustrate in an entertaining way many of the problems, design ideas and issues raised in the text.

NORMAN, D. (1999) *ACM Interactions Magazine*, May/June, 38–42. Affordances, conventions and design. This is a short and thought-provoking critique of design principles.

GRUDIN, J. (1990) The computer reaches out: the historical continuity of interface design. In *CHI'90 Proc.* 261–268. GRUDIN, J. (1989) The case against user interface consistency. *Communications of the ACM*, 32(10), 1164–1173 Jonathan Grudin is a prolific writer and many of his earlier works provide thought-provoking and well documented accounts of topical issues in HCI. The first paper talks about how interface design has expanded to cover many more aspects in its relatively short history. The second paper, considered a classic of its time, discusses why the concept of consistency—which had been universally accepted as good interface design up until then—was in fact highly problematic.

Interactions, January/February 2000, ACM. This special issue provides a collection of visions, critiques, and sound bites on the achievements and future of HCI from a number of researchers, designers, and practitioners.

IDEO provides a well illustrated online archive of a range of interactive products it has designed. (see *www.ideo.com*)

INTERVIEW   with Gitta Salomon

Gitta Salomon is a consultant interaction designer. She founded Swim Interaction Design Studio (swimstudio.com) in 1996 as a consultancy company to assist clients with the design of interactive products. Recently, many of her clients have included start-up companies, developing web-based and other products, who realize the importance of interaction design in ensuring their products are successful but don't know how to do this. Often they get in touch with Swim with partially developed products and ask for help with their interaction design. Swim has consulted for a range of clients, including Apple Computer, Nike, IBM, DoubleClick, Webex, and RioPort.

YR: What is your approach to interaction design?

GS: I've devised my own definition: interaction design is the design of products that reveal themselves over time. Users don't necessarily see all the functionality in interactive products when they first look at them. For example, the first screen you see on a cell phone doesn't show you everything you can do with it. As you use it, additional functionality is revealed to you. Same thing with a web-based application or a Window's application—as you use them you find yourself in different states and suddenly you can do different things. This idea of revealing over time is possible because there is a microprocessor behind the product and usually there is also a dynamic display. I believe this definition characterizes the kind of products we work on—which is a very wide range, not just web products.

YR: How would you say interaction design has changed in the years since you started Swim?

GS: I don't think what we do has changed fundamentally, but the time frame for product development is much shorter. And seemingly more people think they want interaction design assistance. That has definitely changed. There are more people who don't necessarily know what interaction design is, but they are calling us and saying "we need it." All of a sudden there is a great deal of focus and money on all of these products that are virtual and computationally based, which require a different type of design thinking.

YR: So what were the kinds of projects you were working on when you first started Swim?

GS: They were less web-centric. There was more software application design and a few hardware/software type things. For the last year and a half the focus shifted to almost exclusively web-based applications. However, these are quite similar to software applications—they just have different implementation constraints. Right at the moment, the hardware/software products are starting to pick up again—it does seem that information appliances are going to take off. The nature of the problems we solve hasn't changed much; it's the platform and associated constraints that change.

YR: What would you say are the biggest challenges facing yourself and other consultants doing interaction design these days?

GS: One of the biggest challenges is remembering that half of what we do is the design work and the other half is the communication of that design work. The clients almost never bridge the gap for us: we need to bridge it. We always have to figure out how to deliver the work so it is going to have impact. We are the ones who need to ensure that the client is going to understand it and know what to do with it. That part of the work is oftentimes the most difficult. It means we've got to figure out what is going on internally with the client and decide how what we deliver will be effective. In some cases you just start seeing there is no place to engage with the client. And I think that is a very difficult problem. Most people right now don't have a product development process. They are just going for it. And we have to figure out how to fit into what is best described as a moving train.

YR: And what do you use when you try to communicate with them? Is it a combination of talking, meetings, and reports?

GS: We do a number of different things. Usually we will give them a written document, like a report or a critique of their product. Sometimes we will give them interactive prototypes in Director or HTML, things that simulate what the product experience would feel like. In the written materials, I

Figure 1 Steelcase Worklife New York retail showroom. One of the projects Gitta Salomon was involved in was to develop an interactive sales showroom for the company called Steelcase, based in New York. The sales environment was developed to provide various sales tools, including an interactive device allowing salespeople to access case-study videos that can be projected onto the large screens in the background.

often name the things that we all need to be talking about. Then at least we all have a common terminology to discuss things. It is a measure of our success if they start using the words that we gave them, because we truly have influenced their thinking. A lot of times we'll give them a diagram of what their system is like, because nobody has ever visualized it. We serve as the visualizers, taking a random assortment of vaguely defined concepts and giving some shape to them. We'll make an artifact, which allows them to say "Yes, it is like that" or "No, it's not like that, it's like this. . . ." Without something to point to they couldn't even say to each other "No, that is not what I mean" because they didn't know if they were talking about the same thing. Many times we'll use schematic diagrams to represent system behavior. Once they have these diagrams then they can say "Oh no, we need all this other stuff in there, we forgot to tell you." It seems that nobody is writing complete lists of functionality, requirements specifications, or complete documentation anymore. This means the product ideas stay in somebody's head until we make them tangible through visualization.

YR: So this communication process is just as important as the ideas?

GS: I think it is, a lot of times.

YR: So, how do you start with a client?

GS: For clients who already have something built, I find that usually the best way for us to get started, is to begin with the client doing a comprehensive demo of their product for us. We will usually spend a whole day collecting information. Besides the demo, they tell us about their target market, competitors, and a whole range of things. It then takes a longer period of time for us to use the product and observe other people using it to get a much broader picture. Because the client's own vision of their product is so narrow, we really have to step back from what they initially tell us.

YR: So do you write notes, and then try and put it together afterwards, or—what?

GS: We use all kinds of things. We use notes and video, and we sit around with tracing paper and marker pens. When reviewing the materials, I often

try and bring them together in some sort of thematic way. It's often mind-boggling to bring a software product that's been thrown together into any kind of coherent framework. It's easy to write a shopping list of observations, but we want to assemble a larger structure and framework and that takes several weeks to construct. We need time to reflect and stew on what was done and what maybe should have been done. We need to highlight the issues and put them into some kind of larger order. If you always operate at a low level of detail, like worrying and critiquing the size of a button, you end up solving only local issues. You never really get to the big interaction design problems of the product, the ones that should be solved first.

YR: If you're given a prototype or product to evaluate and you discover that it is really bad, what do you do?

GS: Well, I never have the guts to go in and say something is fundamentally flawed. And that's maybe not the best strategy anyway, because it's your word against theirs. Instead, I think it is always about making the case for why something is wrong or flawed. Sometimes I think we are like lawyers. We have to assemble the case for what's wrong with the product. We have to make a convincing argument. A lot of times I think the kind of argumentation we do is very much like what lawyers do.

YR: Finally, how do you see interaction design moving in the next five years? More of the same kind of problems with new emerging technologies? Or do you think there are going to be more challenges, especially with the hardware/software integration?

GS: I think there will be different constraints as new technologies arise. No matter what we are designing, we have to understand the constraints of the implementation. And yes, different things will happen when we get more into designing hardware/software products. There are different kinds of cost constraints and different kinds of interactions you can do when there is special purpose hardware involved. Whereas designing the interaction for applications requires visual design expertise, designing information appliances or other hardware products requires experience with product design. Definitely, there will be some new challenges.

Hopefully, in the next few years, people will stop looking for interaction design rules. There's been a bit of a push towards making interaction design a science lately. Maybe this has happened because so many people are trying to do it and they don't know where to start because they don't have much experience. I'm hoping people will start understanding that interaction design is a design discipline—that there are some guidelines and ways to do good practice—and creativity combined with analytical thinking are necessary to arrive at good products. And then, even more so than now, it is going to get interesting and be a really exciting time.