1: Why and When Do We Need HCD? [Ritter]

The people who use technology must be considered part of the systems that they use. People, in addition to their diversity, have many aspects in common. These characteristics are not all visible or available to system developers without additional study. Understanding the characteristics of users allows designers to create better, safer, more efficient systems. If you understand this approach and the characteristics of users, you understand this book.

1.1 Introduction

Optimizing how people use systems requires an understanding not only of the technologies used but also an understanding of how human beings respond to given situations, how they achieve their goals using devices, how they come to choose the particular actions they choose, how they learn to operate particular artifacts, in sum, how they use their capabilities to interact with technology. More generally, good design requires knowing why, and how users do what they do when they do it. The point of this book is to introduce what we know about why human beings do what they do when they do it as users of technology.

A large number of disciplines study users, with numerous different viewpoints. The data on how people use technology is vast: more than enough to fill a course of any length. Therefore, we will give a conceptual introduction to users, their tasks, and characteristics, hopefully to help you begin thinking about the philosophies, theories, data, and practice for yourselves. This will also serve as an advanced organizer, a structure, for your later learning.

Because the book is largely conceptual, there may seem to be few learnable facts. This may frustrate you, but it will require you to think about things more deeply. There are not always right or wrong answers to many of the questions we raise, but too often there seems to be no more than opinions. The purpose of this book is to help you make your opinions principled, to understand the assumptions that lead to these theories appearing as just opinions, and maybe to also recognize other's unprincipled opinions. Opinions that are grounded in established theories, facts, and assumptions about the user's tasks may not seem as valuable as definitive answers, but they are worth much more than just opinions. (And as these fields mature, more will become known and practiced to provide more science about how people differ, and how tasks and situations differ, which are the factors that make it difficult to predict in this area).

1.2 Dummies for Idiots

This book assumes no previous knowledge; it is designed to be accessible to those without previous study in psychology or computer science (although you should have used a computer, you don’t have to know how one works). If you have already had a traditional human-computer interaction course, this material may be a quite easy read.
and help you organize your thoughts. If you have had several psychology courses, you are likely to recognize much but perhaps not all of the material here.

This book is divided into four parts. These are (a) Introducing the study of users (Chapters 1, 2, and 3), (b) Design relevant user characteristics, with special emphasis on basic psychology for understanding users (Chapters 4 through 11), and (c) Evaluation methods to confirm the design assumptions (Chapters task analysis and usability testing). The book concludes with (d) Some summary examples of applying this knowledge to specific problems (the last two chapters).

This chapter introduces the **study of people using artifacts**, commonly called "users", and will give you some definitions and useful terms, and an overview to organize your further reading. We will also examine your preconceptions about what makes good design and note why studying the user is important. As a way to help you organize your later reading in the years to come, we examine in the second chapter several of the disciplines that study users.

The second part of the book examines specific characteristics of the user. We introduce the idea of "Design relevant user characteristics", which provide four ways of looking at the human user. Of course, this division is not hard and fast, but we offer it as heuristic (i.e., a rule-of-thumb) for dividing up things we need to consider about users. These are (a) **anthropometrics** (the shape of the body and how it influences design), (b) **behavioral issues** (perception and motor output and how they influence design), (c) **cognitive issues** (learning, attention, and other aspects of cognition and how they influence design), and (d) **social issues** (how groups of users behave, and how to support them through design). We will refer to these areas, anthropometric, behavioral, cognitive, and social issues, as the ABCS.

This book will pay more attention to the cognitive aspects of the user, partly because we think it is the most important part, and partly because that is our expertise. After an introduction, additional information on anthropometric research is readily available in a number of books and you can look things up pretty easily. With psychology, this is not as easy, as generalizable design relevant principles are not so readily apparent. As there is so much to know within psychology, the chapters will only introduce topics that are most relevant to designers. For example, in one chapter we will touch on some interesting issues in perception (in vision and hearing). There are entire courses that focus on these and their subcomponents, so we can only offer an introduction here.

The important thing is to acknowledge that there are some **baseline processing capabilities that are characteristics of the human operator**. Beyond that, there are useful regularities in behavior and thought patterns that may be exploited in design. Having raised the issues, we will try to illustrate how they are applicable to the design of interfaces and systems. Although we will give a few facts and some principles, some of these remain contestable. We do not believe that this is problematic in itself, for it illustrates the need for principled critiques and analyses, and often shows the current limits of our knowledge in this area.

The penultimate part of the book will introduce the design process and some evaluation methods that are used in improving interfaces. Again, as this is an active area of research there are now so many approaches it is not possible to cover them all. However, it is possible to introduce the categories of evaluation as well as covering a few in enough detail that you can do start to apply them. We conclude the book with a
summary of how to organize this information about the user and some possible
directions that are currently being explored to apply it.

Hopefully, after reading the book you will have acquired a few key concepts about users
and their behavior. You will have begun thinking about design and about designing with
the human operator in mind. This being the case, you will be able to pursue answers to
specific questions armed with a conceptual outlook and an idea of some of the tools at
your disposal. Reading this book alone will not make you a usability expert, but it should
give you some view as to what constitutes usability and human factors work, and make
you much more sympathetic to supporting the user with technology. You should also be
prepared to take further courses in this area if you think it is for you.

The remainder of this chapter provides motivation for studying the user. There are
specific payoffs we can point to that arise from supporting the user, including decreased
costs, market acceptance, and not killing people.

How to obtain these advantages are not obvious to most programmers. It was not, at
least, to one of us when he was a programmer. Thus the subtitle of "Dummies for Idiots"
(or if you prefer, "Idiots for Dummies", or "Dummies for Dummies", or "Dummies for
programmers", or most gently, "Users for programmers", or what programmers and
interface designers need to know about users).

We provide some example exercises that illustrate how there is real knowledge that has
to be studied. While you are most likely already a computer user and already understand
computers to some extent, it is not true that you fully understand their behavior
because you use a computer. It is similarly true that you do not understand your own
behavior just because you are a human. The largest reason you don't fully understand
your own behavior is because you are too busy behaving to remember what you did, and
you lack the ability to consistently make accurate time distinctions while doing a task.
That is not to say that you know nothing.

It is unarguable that, given the progress of psychological science, that you as a
designer are misinformed if you rely on your native or naive intuitions about how
people behave, particularly yourself, and particularly in the details.

As we shall see, the assumption that programmers understand users because they are
also users has cost lives and millions of dollars. This problem will continue until
programmers and system designers make it their business to study users and users’
cognitive architecture in the same way that they study computer system architectures.

We understand that this view of users is a basic position held by many programmers. It
was, in fact, held for many years by Ritter. So we will include many examples where a
programmer's expectations are likely to be incorrect or incomplete.
1.3 Why study of the user is necessary

OK, at this point, you might be agreeing that taking users into account may be important, but why study them? Aren't you one already? Don't you understand yourself and know what you like?

It turns out that designers, probably even you, have limited understanding of users and the ways they use technology. Some aspects of human behavior are clear and well understood by people without further training, for example, that tough choices seem to take more time to make, and that people generally like things that are bright and shiny.

Studies in psychology and human-computer interaction have found repeatedly that there are large areas of human behavior where the application of unprincipled intuitions to interface design can be quite dangerously incorrect. Table 1-1 lists a few of the many common misconceptions about users' behavior.

Table 1-1. Common misconceptions about user behavior.

- Myth: It is only beginners that make errors with systems.  
  Fact: Experts do too. One aspect of being an expert is being able to notice errors and being more able to know how to recover from them. (errors chapter)

- Myth: Smart people can learn to cope with just about anything.  
  Fact: Smart people can often learn to cope with a wide range of situations and tasks. Will your users be that smart and be able to cope well enough to like, use, and recommend your interface over their previous tool or that of your competitors? Will they have enough time to learn? (chapter on learning and problem solving) There are some problems that no users can cope with (like being two places at the same time or reacting faster than 100 ms).

- Myth: You can’t function without sleep.  
  Fact: A modest lack of sleep (i.e., sleeping one night with 4-5 hours) does not destroy all abilities. Reaction time pretty much remains the same, but higher level judgment and monitoring (vigilance) tasks are impaired to a certain extent (Coleman, 1986). Missing 2 nights of sleep or several nights at 4 hours/night will, however, seriously degrade (20-50%) performance. (Behavior moderators chapter)

- Myth: Asking more people for help will increase your chances of getting it.  
  Fact: Large groups are not necessarily more helpful than a single individual if asked individually (the diffusion of social responsibility, social chapter).

- Myth: People can report accurately how they think.  
  Fact: Asking people to introspect on how they think has proved to be surprisingly unreliable, but asking them to simply talk aloud while performing a task provides an accurate, but incomplete account of the information they are using, from which an observer can build a pretty good account of their thinking (reviewed in Ericsson & Simon, 1993 and discussed in decision making section).
• Myth: The outside world causes you stress.
Fact: The best theories of stress emphasize how stress arises as a response from the individual to external events, not something intrinsic to the event. For example, some people are stressed by computers, others pleased—the computer doesn't change, the person's response does. (behavior moderators and individual differences chapter)

• Myth: It is good to automate as much as possible of a task.
Fact: Automating more of task can make the task more difficult, for example, if it encourages the user either directly or indirectly to pay less attention to the task until the user has to intervene. (multi-tasking)

• Myth: You know what you don't know.
Fact: Students using an early computer vocabulary tutor learned 30% faster with automatic word learning choices made for them. Without assistance they were more likely to study the words they knew and were comfortable with (Atkinson, 1972). (Decision making)

• Myth: If the keystrokes or commands were to change in an editor, it would become essentially unusable.
Fact: Users in such a situation really don't like it (which is important), but their performance doesn't decrease by more than 30% for a short period of a day or two, and then was equivalent to the non-change users (Singley & Anderson, 1989). (Learning and transfer)

• Myth: The higher the fidelity of the training system to a real system, the better the learning.
Fact: While pilots often seem to like simulators based on surface fidelity, that is, shiny knobs and dials, the important aspect for learning is how well the underlying cognitive aspects of the task tutor match the task being trained for (cited in E. Salas, Wilson, Burke, & Bowers, 2002). (learning)

To provide you with concrete examples of this problem of common misunderstandings about users, consider Figures 1-1, 1-2, and 1-3. Figures 1-1 and 1-2 are taken from a study by Payne (1995) on how well naive subjects could judge the quality of interface designs, in particular with respect to the mapping between the controls and the artifact (stimulus-response compatibility). Better designs provide simpler, clearer mappings. These mappings can be between aspects of the tool, between parts of the tool and the world, or between objects in the tool and those in the world. While choosing the best design could be easy to assess because human performance on these tasks are pretty clear about which designs are better, and the best design is nearly 100% better than the worst in these figure, most of the subjects who were bright students at university, were incorrect (60 out of 70 could pick out best one, but only 4 got the complete order correct). Before going on, you may wish to rank them yourself. You can explore this further through an exercise at the end of this chapter.

In Figure 1-1, what is the mapping of lights to switches that give the fastest response time? Can you give a prediction of how long they will take on average? In Figure 1-2, which is the best stove burner to control knob set? If you think you know the best mapping, can you provide a quantitative measure of how much better? If the Layout one has 100 errors for a given amount of use, how many errors will the other two have?
Similarly, as a very simple example, most people would think that they could recognize a penny, but more than half of Nickerson and Adam’s (1979) American subjects could not pick out the penny from the set in Figure 1-3. (An analogous task is to write down the top level menu items in your word processor or another application). You can look up the answers in Exercise 4 at the end of this chapter. As for the penny, you might also find one in your pocket if you live in North America. Outside North America, you might generate similar stimuli with your own coins. The point is, most people do not memorize the features in detail, they memorize just enough to recognize and differentiate the coin from typical distracters, like other legal coins.

Although coinage systems may appear a long way removed from the design of user interfaces, they provide good examples of how and why we can benefit from considering the users’ perspective in design. France and the USA have both tried to introduce new coins (for example, the Susan B Anthony dollar) with little success, partly due to the lack of usability of the new coin. In Britain, by contrast, every major coin has either been introduced or changed over the past 20 years and although there was some resistance, the project has been a resounding success. One reason for the success was a substantial body of research on how people perceived the value of coins (e.g., Bruce, Gilmore, Mason, & Mayhew, 1983) as well as on how the different proposed coins might be made least confusing to the elderly or short-sighted. In the course of the research it was recognized that many people need to identify coins from touch alone (for example, in their pocket) and that designing for the blind user actually meant designing for everyone. The cost of this research was a very small component of the costs of introducing a new coinage system (for example, all of the new vending machines to be developed), but it helped ensure the success of the whole enterprise.

Solving these kinds of problems with brain power alone is hard for one primary reason—we only pay attention to and remember those details of our performance that require our consideration. Hence, we all know well enough what a penny looks like—relative to the other coins we might encounter—but not in any more detail than is strictly necessary. With the set of alternatives provided by Nickerson and Adams, the choice has to be based on recalling specific features of a penny, which most people have never encoded and never needed. Similar effects are seen in computer interfaces where users don’t recall where commands are located on menus.

In these examples we see one of the first universals of human behavior—people remember those details that they pay attention to and only in sufficient detail for the tasks they are performing. This is universal, but it does not enable us to fully predict what details someone will remember, because there are differences in how much attention people have to spare, what tasks they are performing, and thus what details they will remember. The first two problems in Figures 1-1 and 1-2 are difficult because the differences in performance of the tasks are not particularly available to consciousness, and most people’s representation of how they think they perform these tasks in this area do not reflect how people perform the task. The penny question represents the difference between recognition and recall memory. Usually recognizing a penny requires just recognizing it from dissimilar coins. With the set of alternatives provided by Nickerson and Adams, the choice has to be based on recalling the features of a penny, which most people don’t bother remembering (why would they?).

Another classic example is remembering your cell phone number. It takes a long time to learn it because you, yourself, never need to use it (unless you misplace your phone, then calling it is a good strategy for finding it!). But if someone asks you for it, you
have to either recall it, or go through the menus on your phone to find it, eventually recognizing the steps that take you to the number.

(a)

(b)

(c)

(d)

Figure 1-1. (a) Order the quality of these switch to light mappings. (b) Note how long, on average, it will take to push a button on each panel. (Redrawn from Payne, 1995.)

![Figure 1-1](image)

Layout 1

<table>
<thead>
<tr>
<th>Number of Errors</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
</tr>
</tbody>
</table>

Layout 2

<table>
<thead>
<tr>
<th>Number of Errors</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
</tr>
</tbody>
</table>

Layout 3

<table>
<thead>
<tr>
<th>Number of Errors</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
</tr>
</tbody>
</table>

Figure 1-2. Order the quality of these stove burner to knob pairings. If Layout 1 will give 100 errors, how many errors will the other pairings lead to? Redrawn from Chapanis and Lindenbaum (1959).
This disconnection between how users behave and how we think that we behave is common and there are plenty of reasons for it. In most cases, we are too busy doing a task to properly observe how we are doing it. When we can observe how we are doing it, it is rare that we can correctly and completely infer why we are doing the task—the observation of behavior is separate from the generation of it (see Ericsson & Simon, 1993 for a full presentation of this argument). When we recognize how we behave, we rarely make written notes and thus the memories are subject to the frailties of human memory. We don't have a very accurate measure of time and have trouble keeping track of success and failures. Finally, at least for this list, there are certain aspects of our own behavior that are very hard to observe such as basic perception and certain aspects that are hard to describe and reason about verbally, such as certain spatial reasoning tasks.

As a more technical, actual example of not understanding the user, one of us long ago created an interface for a tutoring system that was "obviously easy to use" because it included the "natural" key bindings to open and save application files that matched other applications on the computer it was developed on. A passing psychologist, who did not use that type of computer (a Symbolics), noted that the target audience of that tutor, student pilots, weren't used to using the Symbolics, were they? Luckily, or more likely because of good management, a manual was written and the tutor made more menu driven to assist the actual users, who were not programmers. This is a typical step on the process to becoming interested in users, of knowing your users, particularly when they are different from you.

Know your user, particularly when they are different from you.
1.4 Why study the user: The payoffs

We provide examples here of the types of payoffs that can arise when design of computing systems is done with the user in mind. There are other payoffs as well, which we will add as we think of them, but these are three primary categories. It is useful to know the payoffs as they are the inverse to the risks noted earlier.

1.4.1 Understanding the user can save lives

There are now numerous examples illustrating where ignoring the user can lead to loss of life. Much of the work that has made airplanes the safest transportation per passenger mile has gone into supporting pilots and air traffic controllers to avoid, and more importantly, catch and recover from errors. This has lead to a drastic decrease in accidents previously ascribed to 'pilot error'. As these accidents were typically caused by well trained and alert pilots, it is fairer to diagnose these errors as poor fits between the pilot's capabilities and the machine at particular times and for particular sets of tasks. Improving this fit thus improved airplane safety. The numerous books, journals, and conferences on aviation safety, including the Human Factor's Society's publications in this area, document the problems, approaches to improve safety, and the results.

This improved safety, from understanding the pilot and air traffic controller as user does not always save lives, however, as passengers are another kind of user and subject to their own limitations. Gigerenzer (2004) has argued that while planes are safer than cars, users fear airplanes more, leading them to use cars more than planes and increase the total accident rate.

The area of medicine provides numerous examples as well. For instance, interfaces that allow users (e.g., nurses), to type in the digits of a drug dose are inherently more dangerous than those that force users to dial them in using a wheel for each digit. When typing, a repeated digit can increase the dosage by a factor of ten. This mistake is just not possible with a dial-based interface.

Medical X-ray machines need to be well designed. They are powerful and often offer little margin for error. They are commonly used, which makes them important as well. In addition to their technical requirements, they can have usability problems because their effects are not directly visible (they are, after all, x-rays), and their use has a lot of context. In the case of radiation treatments for cancer, multiple professionals are involved in their use, from the oncologists and radiologists who specify the treatment, the technicians who administer it, and the physicists who maintain it.

There are now multiple examples of where interface design for treatment X-ray machines and other medical devices have ignored the user's capabilities, tasks, context, or some combination of these, and this has lead to loss of life. Although there are other cases of people being killed by poor X-ray machine interfaces, perhaps the most famous case is the Therac 25 (Leveson & Turner, 1993). Between 1985 and 1987 there were six known accidents involving massive overdoses with the Therac. To be fair, such accidents as will be explained later rarely arise from a single cause, and the user interface was but one of the contributors. In addition to problems with the technology and safety interlocks, the system was poorly prepared to deal with typing mistakes by the
From The ABCS of HCI.
If you are using this after 30 June 2011, please contact Ritter for a later version.

Technician, and in many installations provided poor feedback to the radiation technician that could have lead to catching the mistakes sooner.

The ACM Committee on Computers and Public Policy's Risks Digest at catless.ncl.ac.uk/Risks/ provides numerous examples of where poor usability has led to problems. Larger scale examples are available in multiple places, including Norman's books (e.g., D. A. Norman, 1988). Booher and Minninger's (2003) review lists several large (>$10 million) systems that were cancelled or delayed because of poor usability.

1.4.2 Understanding the user can lead to better products

Understanding the user, in addition to saving money and saving lives, can lead to better products through making systems that are more usable, more learnable, and more efficient. There are many examples of cases where being usable made a system into a better, more successful product.

For example, the use of email has surely become more widespread because of the decreased cost of hardware as well as the network effect that as more people take it up, there are more people you can email, making it more attractive. But added to these effects are increased usability of email readers that have progressed from literally requiring "a degree in computer science" to support the installation and use, to a 20 to 30 page manual to now, in some cases, just a web address.

The proliferation of web pages follows this same process. Changes in the costs of hardware and bandwidth have supported this process, but our friends and families and many small businesses would not be generating as many web pages if they had to produce them using raw HTML files. The rise of special purpose editors for HTML has to be acknowledged as one of the driving forces in the increase of the Web. Holmes (2005) also argues this point, and encourages the use of capitalization to reference the Web as the html portion of the Internet. Similarly, America OnLine, despite its limitations, opened up many applications of the Internet to its users because it makes it easier for less experienced people to use these applications. Its success is currently playing out on the stock market, where its founders have been well rewarded (at least for a while) for improving usability.

Technology and usability sometimes go hand-in-hand. The decreasing weight of cell phones has allowed them to be carried. Usability sometimes is its own product. VCRs (although they are disappearing themselves) provide a functionality that’s important to many people. Their usability has been so poor that simply new ways to use them have become products. New coding systems, where the user just types in a code to record a program, and special controllers to turn the VCR on at set times, have generated a product that does nothing new, but just makes existing capabilities more available.

Sometimes the usability of a tool does not increase directly through improving an interface but by providing a completely new interface. Part of the success of the web arises out of the creation of search engines. These systems (all available at www.name.com or through other search engines), like the general search engines AltaVista, Google, Ask, Yahoo, and the more specific search engines like citeseer and DBLP (dblp.uni-trier.de), increase the usability of the web by helping users find information based on the topic they are looking for rather then by the location of the
information. This increase of usability is not driven by the visual interface, but at a deeper level of supporting the user’s tasks.

A theory that we will explain later is Norman’s theory of the gulfs of execution and evaluation. His description points to problems users have, first with understanding what they see (Gulf of Evaluation) and then knowing or finding out what they have to do to execute their task with an interface (Gulf of Execution). The Internet examples above are examples where the Gulf of Execution has been made smaller, and thus made more tractable for a wider range of users.

### 1.4.3 Understanding the user can save money

Designing to support users can save companies a lot of money. We can note some examples here where the savings and losses can be publicly measured in the millions of dollars.

One of the best known amongst user interface professionals comes from Nynex, the New York telephone company in the early 1990’s. The toll and assistance operators (TAO) are the people who help you when you dial "0". They help customers with collect calls, billing, and other more complex calls. In the early 1990's Nynex was considering upgrading their TAO workstation. They had a room with about 100 of these operators located on the East Coast and it was believed that new graphical user workstations could improve productivity. The cost of upgrading all the workstations was going to be about $500,000 (in 1990’s dollars; it would be more today).

To be on the safe side, they had a team of applied psychologists look at just how much faster the new workstations would be. Their analysis, which used a method based on GOMS (described in more detail in Chapter 12) suggested that the new workstations would not be faster, but would, in fact, be 4% slower to operate. This may seem like a small difference, but a 4% reduction in productivity was going to cost Nynex $2.4 million a year—in addition to the cost of the workstations. Of course, Nynex did not find it easy to accept this analysis, and so they ran a user study to discover how much faster the new workstations would really be. After allowing time for the operators to learn the workstations, the operators' performance plateaued about where it was predicted--4% slower. NYNEX now claims that this study saved them $2.4 million per year. The user study paid for itself in the first week. (See Gray, John, & Atwood, 1992; Gray, John, & Atwood, 1993 for more details. Papers like these are often available online from their author's home pages on the web.)

In this particular instance the slowdown in operator performance was not caused by the fact that the new workstations simply required the user to take more steps to achieve the same goal, but because the user could not do two things at once in the new workstation, which they could with the old. This was an effect that couldn’t be compensated for by the improved processor speed of the new workstations because the limitation was not in the computer but in the user and their interaction with the workstation.

The NYNEX example reveals benefits of considering the user even when everything is going right. In many instances the main advantage of user studies is to help us avoid making mistakes. Users often don’t type what they want to type, and sometimes push buttons that they don’t intend to push. Strangely enough this problem can be more
prevalent amongst highly-skilled expert users, than amongst beginners. Errors that occur when someone knows the right thing to do, but accidentally does something different are commonly referred to as 'slips', to distinguish them from mistakes, where the action is taken on the basis of an incorrect plan.

These slips can also occur on well practiced interfaces that don’t attempt to catch these slips. And these slips can be very expensive. A Pennsylvania local paper (Centre Daily Times, 15 Feb 2002, p. C38) reported that a financial services firm lost up to $100 million because they executed a sell order of 610,000 shares at 16 yen instead of the correct order of 16 shares at 610,000 yen (approximately 100 yen/US$). Yikes. And this is not that an uncommon story.

These effects occur in areas besides finance. Booher and Minninger (2003) review the application of user models to evaluate Army systems. They report many instances where millions of dollars over the life of the device were saved by better interface design, through reduced training, for example, and numerous examples where systems had to be scrapped or modified at great expense because they were not usable.

Use of machines that have different modes often help mislead users to make errors. Photocopier machines that can also FAX have a common problem. These machines’ base mode is to copy, but users may not see and may not understand this. Users can start to fax a page somewhere. After the area code gets put in by the user (say, 415), the copier can and does starts to copy, 415 times! More explicit displays and more intelligent systems might attempt to catch this type of error. Copies are cheap, but this type of problem with airline tickets, machine tools, or photographic film would get expensive quickly.

In addition to avoiding expensive errors, understanding the user and supporting them can also lead to success through useful interfaces. America OnLine and the initial Netscape browser are examples of products that were successful because they made an existing service more accessible. Recent work in E-business suggest that ease of learning can also lead to a type of brand recognition and later to loyalty (E. J. Johnson, Bellman, & Lohse, 2002).

So, where interfaces manipulate expensive resources or where the user’s time costs money (which is nearly every commercial system), keeping the user in mind can often lead to saving money—sometimes quite a lot of money.

1.4.4 But understanding the user does not guarantee success

After having made the point that improving the usability of a system can save money, save lives, and lead to product success, it is worth acknowledging that usability is neither a necessary nor sufficient condition for success nor is it protection against loss of money or life. Systems with poor usability can still be successful for a variety of reasons. For example, they may offer a functionality that is unique and useful. Early planes, computers, printing presses, and satellite phones were all difficult to use, but successful because of their unique functionality.

Products that are well designed with the user in mind may still not be successful. Most or all aspects must be right for a product or system to succeed. Making the usability right does not make the time to market right, it does not make the price appropriate,
and other critical aspects such as reliability or marketing may fail. The system also has
to be usable in a larger sense as well. The interface may be well designed on a local
level, but if it clashes too much with existing practice (even if the new system is
correct), and management does not support the transition, the new system can fall into
disuse. Glashko and Tabas (2009) argue that success is even more complicated and
arises from understanding the user, the business model, and the technology.

Recent work edited by Pew and Mavor (2007) help explain this. We will take up their
theory about a risk-driven spiral model later in the book. Briefly, for some systems
usability is not the biggest risk against success. Usability is an important aspect, just
not the only aspect or the biggest risk to success. All other reasons for success have
this same logic in that they are not necessary nor sufficient. The lack of usability is,
however, a sufficient reason for failure that is not always kept in mind as much as it
should be.

1.5 When do you need to study the user?

Historically, innovation is the starting point for any system. That is, the system gets
created based on a new idea. Typically this idea is a technical idea. Sometimes this can
be a somewhat small iterative idea, like a slightly improved spreadsheet, as well as big
new ideas such as combining a very small hard drive with a way to translate files on it to
sound.

Computers began as highly specialized devices, used by only a few people. In those
days, the users were largely the programmers who were highly technologically literate
and who had probably built the machines themselves. From this developed the batch
processor, punched holes in cards that embodied the program and which was fed into a
reader. This was highly error-prone activity, even for the designers of the cards, which
led to the development of the teletype so that one could see what was being entered.
As the market grew, so did the importance of usability issues.

When there is a single user for the system, that is, its creator, the need for interface
design is not very high. When the users helped create the system, they can use a much
more primitive interface. Many research systems fit this description. Even when there
are small groups of expert users, interfaces may not be important because there simply
are not enough users to make the improvements available from a better interface worth
more than the cost of creating it. Early aircraft are another example. No-one would
claim that the Wright brother's plane was usable, nor that pre-synchro-mesh (early
manual transmission) cars were easy to learn to drive. In both cases there was no other
affordable means of building these systems, and certain aspects of usability could not be
supported by the technology. Also in both cases, the systems were usable provided
they were used by experts.

There are thus three potential contributions from psychology for the design of an
artifact and how it is used:

(a) to select the people who can use the system: through psychometric testing,
job samples, qualifications and so on; (we will briefly discuss this in the section on
individual differences)
(b) to train the people to be right for the system; (this is often used, so we will discuss learning later)

(c) to redesign the system (which the bulk of the book is devoted to).

As technology improves and the number of users increases, particularly in the area covered by human-computer interaction (HCI), the answer of redesigning the system becomes more and more practical and important. The pace of technological development has meant that option (a) and option (b) have become increasingly insufficient over the last 20 years. In most cases everyone expects to be able to use a new system without much training. Furthermore, political changes have made it a necessity to make systems accessible to more people (e.g., the Americans with Disabilities Act), making option (a) less acceptable. Changes in employment practices where people switch jobs much more often has made training an expensive undertaking for an organization, with the effect that people demand technological systems that require less training than they would have done 20-30 years ago.

Thus, option (c) has become increasingly important, and this is where HCI and human factors as technical disciplines make the greatest contribution. It is this option that is addressed in this book. We offer here a few rules of thumb about when to particularly study the user and to take account of their characteristics in your design, where the payoffs from Section 1.3 are most likely to be worthwhile.

1.5.1 As early in the design process as possible

It is a well known and true statement in the area of design that changes to a design cost much more later in the process. The HCI literature is full of examples of attempts to improve an interface or system to support users after the system has been created and is least changeable. There are very few success stories that start with "After the system was built we considered the user and ....".

One of the take-away messages from this book should be that support for the users and their tasks should be incorporated from that start. For example, it is much easier to create a web site to provide the information that you know that users will need than to create a web site, check it for the knowledge that users will need, and then modify it.

You can also find this effect in your writing, an important aspect of interface design we will argue later. The preferred way to write, at least according to the work we respect and what we teach and were taught, is to decide what you are trying to say, outline how that will be delivered, and then write. This allows the design of the information to support the reader (the user), and for the design to support it early on when the design is more mutable (outlines are easier to manipulate than prose). Sitting down and writing is as bad as sitting down and programming, it is not the approach used by the most successful writers (Flower, 1981).

1.5.2 When there are lots of 'em or they are important

Another place to particularly take account of the user is when the payoff is large. This occurs when there are many users or the users are important.
Large commercial software, as well as open-source or shareware that expects to be taken up by large numbers of people, can offer a large payoff. If your software can be installed 50% faster by 10,000 people, it is likely to be a competitive advantage, and save your users time. This may not be your time, but if the users are your employees or students, the increased usability will more than pay for your investment in usability.

You will also want to support users in extreme environments (e.g., astronauts and pilots) and users with limited time (e.g., executives). In addition to these examples, there are other users with time limitations and important missions. University teachers, for example, appreciate ease of use of technology enhanced classrooms. So do their students, except they will only notice when the classrooms are hard to use, when light switches are not labeled or hard to find, or when it is hard to select the projector. This is a more mundane example of where time is limited, and improved technology may not be used if it is difficult to learn or cannot be used quickly.

### 1.5.3 When you are not like them

A time particularly to study users is when you do not know them very well because they are not like you. Too many designers assume that they know the users and their environments. This is usually not a problem when there are obvious differences, but there are many areas where there are real differences between designer's perceptions and the target users' resources, capabilities, and tasks. Sometimes this leads to success stories. For example, the spreadsheet designer did not realize that many people could use a spreadsheet for tabular data, because it was designed for accountants.

More often, lack of understanding of users leads to excluding groups of users. We have worked with web sites that now make sure that they have text only versions to not only support the visually impaired (through text readers and descriptions of pictures), but also two types of users that do not come to mind easily in a US college environment with ubiquitous broadband—those users separated from the site via dialup lines or by vast distances (Ritter, Freed, & Haskett, 2005). Both of these types of users must be kept in mind because they are not like the designers.

If you can define your users, this may help you know when not knowing them is a risk to system development. If the users are all office workers and you are an office worker, you can probably use people around you to plan and test it. If the system will be used out in the field (and you are not in the field), you should look either at existing systems (e.g., ATMs) like the system you are designing, or, better, consult some books or run a small study and meet the users and see their context and tasks.

### 1.5.4 When lives are at risk

Another time to study users is when their task is important. Human factors, and to a lesser extent HCI, have studied interfaces to support users that perform tasks that have large impacts. Power plant operators, robotic surgical tool operators, and air traffic controllers are typically not at risk themselves as they use interfaces. The people they serve, however, are.

This point can be generalized to any interface where the outcomes of its use are important. The interface may offer a way to save money, or it may be that errors lead to losses of time, money, or even lives.
1.5.5 Further descriptions are available

The rest of the book starts to introduce more about users, what they are like, what they like, and more about specific users and tasks. We also include in later sections easy to use approaches to include this knowledge in the design process.

This book also includes numerous pointers to further information. As an introductory text it cannot cover all that is known about users, but it's designed to provide pointers that can let you learn more when you want or need to.

We also need to add that users need to be included when misunderstanding them can lead to problems. A useful word to keep in mind is risk. If there is not a risk that arises in this area, then users do not have to be kept in mind. For example, the 10th redesign of an existing system used in house by the company that makes the system has less risk than developing a system for users in another country. Developing a new type of paper has less risk for usability than for developing a new type of lethal technology. So, equally, where there is no or little risk, less attention should be paid to the user. For a final example, when the main user is the developer, there is little risk for being usable. When the main test user is the developer and the developer is not like the final user, there is huge risk that the system will not support the final users.

Kegworth Air Crash [sidebar]

The Kegworth air disaster is a useful case study to keep in mind when reading this book. It illustrates many of the points made in this book, and can help organize your understanding as you read. Most air accidents are similar not in the details, but in the range of mistakes that contribute to the disaster, including the Gimli Glider incident and other incidents.

The Kegworth Air Disaster occurred on 8 January 1989 when a British Midlands plane crashed on the way from London to Belfast. The plane crashed, we argue, for several reasons, not just a single reason. If you start at the beginning and go to the end, you can see that it was not a single error or mistake, but a series of mistakes and errors at several levels.

The plane was a modification to an existing plane, but had a slightly new engine. This new engine, only a few percent more powerful, was not tested on a real plane at a high altitude, but was tested on the ground. Thus, the engine was at fault, the technology should have failed. The pilots were not trained on the new plane's controls, but had a short lecture course and were expected to learn while flying it. Thus, the employer did not train the pilots on the new plane. When the engine failed, the pilots indication of failure was vibration and smoke. They could not tell which engine was failing (but a small, single gauge that was not required to be working to fly). Thus, the system designers of that flight gauge and the training were at fault. They turned off one engine, and the juddering stopped. They thought it was because the bad engine was off, but it was, in fact, that the auto-throttle could handle the fuel flow. This is a fault of training and perhaps cockpit system design and training. The choice of the wrong engine is also just bad luck in this case. The pilots announced this to the crew and passengers, noting which engine they had shut down. While the passengers could see the engine on fire (!), they did not inform the pilots that they had shut down the wrong
engine. This is an example of social distance. The crew might not have looked out, but if they did, they too would have been separated by social distance and not wanting to correct the person in charge, or not sure. This is part of human nature, but better systems help teams avoid this limitation. Air traffic control offered two places to land. They chose to land on their flight path, but slightly further away. This was an unlucky choice based on not understanding the problem. While the pilots were trying to review the situation, air traffic control was talking with them a lot, and the pilots had problems resetting their navigation equipment to land at Nottingham. The disaster was perhaps savable but the system and the small navigation system contributed to lack of attention. Finally, when they were in sight of the airport, with the good engine turned off, the bad engine failed completely. As they landed, they landed onto and across a major road. This road had had noise abatement embankments (small hills) put up to shelter the surrounding land from motorway noise. This caused the plane to bounce, and probably compounded the crash. The formal enquiry ruled that it was pilot error, but as you consider this accident, you might think that it was a series of mistakes, errors, and bad luck from a wide range of people who were part of this system, broadly defined.

Thus, for larger system design, you can see that knowledge about multiple levels of users is necessary. This is a particularly broad example, but even small systems will see several aspects and several levels.

Further information is available on the web and further examples that are very similar in the range of errors that lead up to the final disaster is found in many other stories of aircraft and power plant disasters.

1.6 How much do you have to study the user?

Given that we have to study the user, the next questions are how much, how long, how thorough? On the one hand, we have argued that if you do nothing, you can fail. On the other hand, one can spend a lot of time studying the user, indeed, to the point of not getting an interface built, which is another risk.

Understanding when to stop and when to build is a reasonable question. It is clearly one of the future questions for HCI and system design in general. It can also be a very reasonable question. What is the payoff curve for working on studying the user? Can you, for example, ask for $1,000 worth of usability please? For example, Nielson (1993) has argued for studying several users, which is almost certainly better than no users or only one user.

In their edited book, Pew and Mavor (2007) put forward a risk-driven spiral design model of system design. This approach provides a subjective answer to how much to study the user. As the system design process progresses, the risks to success are evaluated. At times, progress with the technology is the largest risk to success. At other times and on other projects, not knowing the user and their tasks should be considered the largest risk. Thus, their answer is to study the user and their tasks until the risk of not knowing more is lower than the other risks. This provides an answer, albeit a slightly subjective one, to when to switch from studying and designing to implementing.
1.7 Conclusions

We introduced the theme of this book, that understanding the user supports the design of better systems and can provide useful payoffs and help avoid disasters. Unlike other system components, users will remain important, and are unlikely to be replaced by a more current technology. The aspects of the user include their physical shape, how they perceive and think, and how they interact with other people. These aspects are taken up in turn in later chapters. We have sketched some of the fields that study the user and some of the various approaches for summarizing and using this knowledge, and in the next chapter we review these resources and their history in more detail.

The book also notes additional resources, so you should also be able to help yourself learn more, either now as you are using it, or later when you are called upon to design a system that includes users, build systems that includes users, or fix systems that includes users.

While we now know a lot about users, there are still many open problems. As you read the rest of this book, you may wish to keep in mind the examples of some of the remaining problems in Table 1-2.

Table 1-2. Remaining hard problems in contemporary system design.

- The size of systems and diversity of users and tasks are increasing. How are we to find, represent, and use this information?

- The complexity of the systems are increasing: users don't always get adequate feedback of what is going on, and can't see the internal state of the system. Norman (1988) provides further examples. How are we to keep users informed?

- The nuances of social and organization factors and the communication of these nuances through computer supported communication are not fully understood and predictable. How can designers get and use this information? How can it be represented?

- There remain problems including what we already know about users in the design and construction of systems in a way that makes it easy for designers to do so. How can we improve usability for designers?

- There are attempts to create a unified theory of how users behave, but this theory is not yet fully created or available for automatic application. Our study of the user needs to go beyond recommendations about the design of technology—can we offer a conceptual basis for these recommendations?

- With further understanding come questions about lower and higher levels. Once we know how users work in small groups we can see that larger groups also have...
influences as do previous groups who used the system. How do we include this information?

1.8 Glossary of terms

We will provide you with a list of terms used in each chapter. If you don't recognize these terms, you should go back and reread the section. As a study aid, we have deliberately left room for notes here, and do not spell them out, except where the text is unclear or where we can add additional material.

anthropometrics, behavioral, cognitive, social levels, HCI

users, operators, human operators

designers

heuristic. A rule-of-thumb. Heuristics include putting commonly used items first on an interface, eating desert first, and, a particularly weak one, if you are lost, follow someone, they may be going where you are.

1.9 Other resources

Here, we note some further books and online resources. Our approach reflects our teaching style, to provide a relatively sparse text here, with pointers to further reading, where we expect that any given student is likely to read one of the references per chapter either as additional assigned reading or as an optional part of a course.

Christopher Wickens has worked in the area of human factors for quite some time. His textbooks (C. D. Wickens, Gordon, & Liu, 1998; C. D. Wickens & Hollands, 2000) provide more details than this book does. His books focus more on human factors and physical engineering of workplaces, but as we noted above, these two areas overlap with HCI and his books are very useful as further reading.

Don Norman started out as an experimental psychologist. He has since migrated towards cognitive science and human-computer interaction. His books, The design of everyday things (D. A. Norman, 1988) and related books, provide a useful set of examples of why design matters. At times the examples may seem small, but in nearly every case their impact is magnified by the number of people affected, the possible severity of the consequences, or their clarity. Numerous people have been convinced of the importance of human-computer interaction based on this book.
There are several excellent larger textbooks on HCI. Ones that we particularly like or have used include Preece’s (1994), Dix, Finlay, Abowd and Beale's books, both titled *Human-Computer Interaction*, and *Interaction Design: Beyond Human-Computer Interaction, 2nd Edition*, by Sharp, Rogers, and Preece. These books include more on technology and more on designing interface technology, and as they are not as focused on users they do not cover as many aspects of users, which is what this book is about. Clayton Lewis and John Rieman’s shareware book: *Task-Centered User Interface Design*, hcibib.org/tcuid/ covers similar material, but focuses more on design and task analyses, and does not include as much psychology. It does help, however, put this material into perspective for design, and could be a useful companion book to this book for a course.

There are numerous online resources available on HCI and human factors. If you want to know more, nearly every search engine can provide you with additional information. In particular, the Computer-Human Interaction Special Interest Group of the Association for Computing Machinery (ACM-SIGCHI) has a very useful web site at www.acm.org/sigchi/ . They produce the CHI conference every year, a magazine *interactions*, and a now online newsletter, *SigCHI Bulletin*, bulletin.sigchi.org/.

Students find Jacob Nielsen's site particularly assessable: www.useit.com/alertbox/ . It notes the practical applications and problems that can arise in web site usability.

The HCI Bibliography Project, hcibib.org, provides an online bibliography of papers in journals and conferences related to HCI. In some cases the papers are available as well.

*Software failure: Management failure* by Flowers (1997) provides several examples where ignoring multiple aspects of users, on an individual and group level, as well as social processes between users and among developers led to software-based systems going dangerously awry.

### 1.10 Exercises

**Exercise 1.1**

Consider a wireless personal digital assistant (PDA), either a specific one or a composite one and consider the human factors of using it. What are the common errors that people make? What difficulties might people encounter in using the PDA? What difficulties might people have in understanding it?

Write short notes (about one side of a page, total) on some of these problems, classifying them as anthropometric/ biomechanical, behavioral, cognitive, social, or technical.

**Exercise 1.2**

Go to the web or the library and bring back a horror story about a bad interface and its consequences. Prepare a short talk for class noting who the users were, what they were trying to do, and what went wrong.
Exercise 1.3

With respect to a web site for a university department (or your company's), what types of users will use that site? As a short, perhaps 20 minute exercise, create a list of all the types of users you think might use the site. When recently looking at our own department's site (Ritter, Freed, & Haskett, 2005) we came up with more than 15 types of users as defined by age, tasks, access to the Internet, physical location, and status.

Exercise 1.4

Payne's (1995) study suggests that designers without training on the user will often make poor design choices. In his study, using Figure 1-1 above, only 4/70 got the correct order of (a) (411 ms), (d) (469 ms), (b) and (c) (each 539 ms).

The lights in D have a simple rule that users can follow, hit the diagonal button for the light that is on (if you see what we mean). The lights in B and C have two rules (or at least a more complex rule): If the lights are in the middle, hit the button below. If the lights are on the end, hit the diagonal button. The extra conditions (or rules, depending on how you represent it), take extra time to learn and extra time to perform. They would also lead to more errors, but that is not covered in the text at this point.

For the examples in Figure 1-2, only 4 out of 53 subjects got the correct order of layout 3 (76 errors per 1200 trials), layout 2 (116 errors per 1200 trials), and then layout 1 (129 errors per 1200 trials), and only 15 out of 53 could correctly identify the best design.

Decide what makes a good design. You may wish to consider aesthetics and error rates. Ask some of your friends and or family to choose the best designs in Figure 1-1 and 1-2. How do they compare to your performance and to Payne's subjects? If you have access to interface designers, ask them. How does your own stove compare to these mappings? What do your results mean for interface design?

Exercise 1.5

What are some other payoffs? And, what are some remaining hard problems not noted in Table 1-2? As ways to brainstorm, consider the types of outputs the various fields would provide. As another hint, consider fields that study users or aspects of users, and consider what they might want from interfaces or from interactions with interfaces or systems.