

What Makes Something Usable?

What makes a product or service usable?

Usability is a quality that many products possess, but many, many more lack. There are historical, cultural, organizational, monetary, and other reasons for this, which are beyond the scope of this book. Fortunately, however, there are customary and reliable methods for assessing where design contributes to usability and where it does not, and for judging what changes to make to designs so a product can be usable enough to survive or even thrive in the marketplace.

It can *seem* hard to know what makes something usable because unless you have a breakthrough usability paradigm that actually drives sales (Apple's iPod comes to mind), usability is only an issue when it is lacking or absent. Imagine a customer trying to buy something from your company's e-commerce web site. The inner dialogue they may be having with the site might sound like this: *I can't find what I'm looking for. Okay, I have found what I'm looking for, but I can't tell how much it costs. Is it in stock? Can it be shipped to where I need it to go? Is shipping free if I spend this much?* Nearly everyone who has ever tried to purchase something on a web site has encountered issues like these.

It is easy to pick on web sites (after all there are so very many of them), but there are myriad other situations where people encounter products and services that are difficult to use every day. Do you know how to use all of the features on your alarm clock, phone, or DVR? When you contact a vendor, how easy is it to know what to choose in their voice-based menu of options?

What Do We Mean by “Usable”?

In large part, what makes something usable is the *absence of frustration* in using it. As we lay out the process and method for conducting usability testing in this book, we will rely on this definition of “usability;” when a product or service is truly usable, *the user can do what he or she wants to do the way he or she expects to be able to do it, without hindrance, hesitation, or questions.*

But before we get into defining and exploring usability *testing*, let’s talk a bit more about the concept of usability and its attributes. To be usable, a product or service should be useful, efficient, effective, satisfying, learnable, and accessible.

Usefulness concerns the degree to which a product enables a user to achieve his or her goals, and is an assessment of the user’s willingness to use the product at all. Without that motivation, other measures make no sense, because the product will just sit on the shelf. If a system is easy to use, easy to learn, and even satisfying to use, but does not achieve the specific goals of a specific user, it will not be used even if it is given away for free. Interestingly enough, usefulness is probably the element that is most often overlooked during experiments and studies in the lab.

In the early stages of product development, it is up to the marketing team to ascertain what product or system features are desirable and necessary before other elements of usability are even considered. Lacking that, the development team is hard-pressed to take the user’s point of view and will simply guess or, even worse, use themselves as the user model. This is very often where a system-oriented design takes hold.

Efficiency is the quickness with which the user’s goal can be accomplished accurately and completely and is usually a measure of time. For example, you might set a usability testing benchmark that says “95 percent of all users will be able to load the software within 10 minutes.”

Effectiveness refers to the extent to which the product behaves in the way that users expect it to and the ease with which users can use it to do what they intend. This is usually measured quantitatively with error rate. Your usability testing measure for effectiveness, like that for efficiency, should be tied to some percentage of total users. Extending the example from efficiency, the benchmark might be expressed as “95 percent of all users will be able to load the software correctly on the first attempt.”

Learnability is a part of effectiveness and has to do with the user’s ability to operate the system to some defined level of competence after some predetermined amount and period of training (which may be no time at all). It can also refer to the ability of infrequent users to relearn the system after periods of inactivity.

Satisfaction refers to the user’s perceptions, feelings, and opinions of the product, usually captured through both written and oral questioning. Users are more likely to perform well on a product that meets their needs and

provides satisfaction than one that does not. Typically, users are asked to rate and rank products that they try, and this can often reveal causes and reasons for problems that occur.

Usability goals and objectives are typically defined in measurable terms of one or more of these attributes. However, let us caution that making a product usable is never simply the ability to generate numbers about usage and satisfaction. While the numbers can tell us whether a product “works” or not, there is a distinctive qualitative element to how usable something is as well, which is hard to capture with numbers and is difficult to pin down. It has to do with how one interprets the data in order to know *how* to fix a problem because the behavioral data tells you *why* there is a problem. Any doctor can measure a patient’s vital signs, such as blood pressure and pulse rate. But interpreting those numbers and recommending the appropriate course of action for a specific patient is the true value of the physician. Judging the several possible alternative causes of a design problem, and knowing which are especially likely in a particular case, often means looking beyond individual data points in order to design effective treatment. There exist these little subtleties that evade the untrained eye.

Accessibility and usability are siblings. In the broadest sense, accessibility is about having access to the products needed to accomplish a goal. But in this book when we talk about accessibility, we are looking at what makes products usable by people who have disabilities. Making a product usable for people with disabilities — or who are in special contexts, or both — almost always benefits people who do not have disabilities. Considering accessibility for people with disabilities can clarify and simplify design for people who face temporary limitations (for example, injury) or situational ones (such as divided attention or bad environmental conditions, such as bright light or not enough light). There are many tools and sets of guidelines available to assist you in making accessible designs. (We include pointers to accessibility resources on the web site that accompanies this book (see www.wiley.com/go/usabilitytesting for more information.) You should acquaint yourself with accessibility best practices so that you can implement them in your organization’s user-centered design process along with usability testing and other methods.

Making things more usable and accessible is part of the larger discipline of user-centered design (UCD), which encompasses a number of methods and techniques that we will talk about later in this chapter. In turn, user-centered design rolls up into an even larger, more holistic concept called *experience design*. Customers may be able to complete the purchase process on your web site, but how does that mesh with what happens when the product is delivered, maintained, serviced, and possibly returned? What does your organization do to support the research and decision-making process leading up to the purchase? All of these figure in to experience design.

Which brings us back to usability.

True usability is invisible. If something is going well, you don't notice it. If the temperature in a room is comfortable, no one complains. But usability in products happens along a continuum. How usable is your product? Could it be more usable *even though* users can accomplish their goals? Is it worth improving?

Most usability professionals spend most of their time working on eliminating design problems, trying to minimize frustration for users. This is a laudable goal! But know that it is a difficult one to attain for every user of your product. And it affects only a small part of the user's experience of accomplishing a goal. And, though there are quantitative approaches to testing the usability of products, it is impossible to measure the usability of something. You can only measure how *unusable* it is: how many problems people have using something, what the problems are and why.

By incorporating evaluation methods such as usability testing throughout an iterative design process, it is possible to make products and services that are useful and usable, and possibly even delightful.

What Makes Something Less Usable?

Why are so many high-tech products so hard to use?

In this section, we explore this question, discuss why the situation exists, and examine the overall antidote to this problem. Many of the examples in this book involve not only consumer hardware, software, and web sites but also documentation such as user's guides and embedded assistance such as on-screen instructions and error messages. The methods in this book also work for appliances such as music players, cell phones, and game consoles. Even products, such as the control panel for an ultrasound machine or the user manual for a digital camera, fall within the scope of this book.

Five Reasons Why Products Are Hard to Use

For those of you who currently work in the product development arena, as engineers, user-interface designers, technical communicators, training specialists, or managers in these disciplines, it seems likely that several of the reasons for the development of hard-to-use products and systems will sound painfully familiar.

- Development focuses on the machine or system.
- Target audiences change and adapt.
- Designing usable products is difficult.
- Team specialists don't always work in integrated ways.
- Design and implementation don't always match.

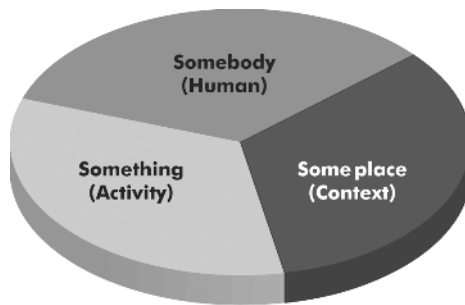


Figure 1-1 Bailey's Human Performance Model

Reason 1: Development Focuses on the Machine or System

During design and development of the product, the emphasis and focus may have been on the machine or system, not on the person who is the ultimate end user. The general model of human performance shown in Figure 1-1 helps to clarify this point.

There are three major components to consider in any type of human performance situation as shown in Bailey's Human performance model.

- The human
- The context
- The activity

Because the development of a system or product is an attempt to improve human performance in some area, designers should consider these three components during the design process. All three affect the final outcome of how well humans ultimately perform. Unfortunately, of these three components, designers, engineers, and programmers have traditionally placed the greatest emphasis on the *activity* component, and much less emphasis on the human and the context components. The relationship of the three components to each other has also been neglected. There are several explanations for this unbalanced approach:

- There has been an underlying assumption that because humans are so inherently flexible and adaptable, it is easier to let them adapt themselves to the machine, rather than vice versa.
- Developers traditionally have been more comfortable working with the seemingly "black and white," scientific, concrete issues associated with systems, than with the more gray, muddled, ambiguous issues associated with human beings.
- Developers have historically been hired and rewarded not for their interpersonal, "people" skills but for their ability to solve technical problems.

- The most important factor leading to the neglect of human needs has been that in the past, designers were developing products for end users who were much like themselves. There was simply no reason to study such a familiar colleague. That leads us to the next point.

Reason 2: Target Audiences Expand and Adapt

As technology has penetrated the mainstream consumer market, the target audience has expanded and continues to change dramatically. Development organizations have been slow to react to this evolution.

The original users of computer-based products were enthusiasts (also known as early adopters) possessing expert knowledge of computers and mechanical devices, a love of technology, the desire to tinker, and pride in their ability to troubleshoot and repair any problem. Developers of these products shared similar characteristics. In essence, users and developers of these systems were one and the same. Because of this similarity, the developers practiced “next-bench” design, a method of designing for the user who is literally sitting one bench away in the development lab. Not surprisingly, this approach met with relative success, and users rarely if ever complained about difficulties.

Why *would* they complain? Much of their joy in using the product was the amount of tinkering and fiddling required to make it work, and enthusiast users took immense pride in their abilities to make these complicated products function. Consequently, a “machine-oriented” or “system-oriented” approach met with little resistance and became the development norm.

Today, however, all that has changed dramatically. Users are apt to have little technical knowledge of computers and mechanical devices, little patience for tinkering with the product just purchased, and completely different expectations from those of the designer. More important, *today’s user is not even remotely comparable to the designer in skill set, aptitude, expectation, or almost any attribute that is relevant to the design process.* Where in the past, companies might have found Ph.D. chemists using their products, today they will find high-school graduates performing similar functions. Obviously, “next-bench” design simply falls apart as a workable design strategy when there is a great discrepancy between user and designer, and companies employing such a strategy, even inadvertently, will continue to produce hard-to-use products.

Designers aren’t hobbyist enthusiasts (necessarily) anymore; most are trained professionals educated in human computer interaction, industrial design, human factors engineering, or computer science, or a combination of these. Whereas before it was unusual for a nontechnical person to use electronic or computer-based equipment, today it is almost impossible for the average person *not to use* such a product in either the workplace or in private life. The overwhelming majority of products, whether in the workplace or the home, be they cell phones, DVRs, web sites, or sophisticated testing equipment, are

intended for this less technical user. Today's user wants a tool, not another hobby.

Reason 3: Designing Usable Products Is Difficult

The design of usable systems is a difficult, unpredictable endeavor, yet many organizations treat it as if it were just "common sense."

While much has been written about what makes something usable, the concept remains maddeningly elusive, especially for those without a background in either the behavioral or social sciences. Part art, part science, it seems that *everyone* has an opinion about usability, and how to achieve it — that is, until it is time to evaluate the usability of a product (which requires an operational definition and precise measurement).

This trivializing of usability creates a more dangerous situation than if product designers freely admitted that designing for usability was not their area of expertise and began to look for alternative ways of developing products. Or as Will Rogers so aptly stated "It's not the things that we don't know that gets us into trouble; it's the things we do know that ain't so." In many organizations usability engineering has been approached as if it were nothing more than "common sense."

When this book was first published in 1994, few systems designers and developers had knowledge of the basic principles of user-centered design. Today, most designers have some knowledge of — or at least exposure to — user-centered design practices, whether they are aware of them or not. However, there are still gaps between awareness and execution. Usability principles are still not obvious, and there is still a great need for education, assistance, and a systematic approach in applying so-called "common sense" to the design process.

Reason 4: Team Specialists Don't Always Work in Integrated Ways

Organizations employ very specialized teams and approaches to product and system development, yet fail to integrate them with each other.

To improve efficiency, many organizations have broken down the product development process into separate system components developed independently. For example, components of a software product include *the user interface*, *the help system*, and *the written materials*. Typically, these components are developed by separate individuals or teams. Now, there is nothing inherently wrong with specialization. The difficulty arises when there is little integration of these separate components and poor communication among the different development teams.

Often the product development proceeds in separate, compartmentalized sections. To an outsider looking on, the development would be seen as depicted in Figure 1-2.

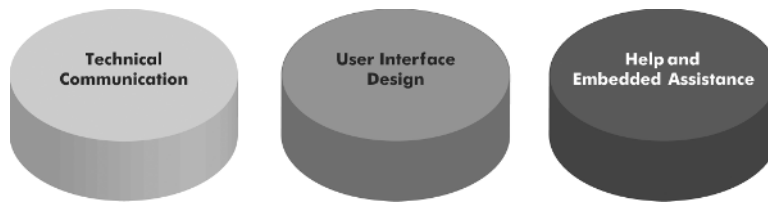


Figure 1-2 Nonintegrated approach to product development

Each development group functions independently, almost as a silo, and the final product often reflects this approach. The help center will not adequately support the user interface or it will be organized very differently from the interface. Or user documentation and help will be redundant with little cross-referencing. Or the documentation will not reflect the latest version of the user interface. You get the picture.

The problem occurs when the product is released. The end user, upon receiving this new product, views it and expects it to work as a single, integrated product, as shown in Figure 1-3. He or she makes no particular distinction among the three components, and each one is expected to *support and work seamlessly with the others*. When the product does not work in this way, it clashes with the user's expectations, and whatever advantages accrue through specialization are lost.

Even more interesting is how often organizations unknowingly exacerbate this lack of integration by *usability testing each of the components separately*. Documentation is tested separately from the interface, and the interface separately from the help. Ultimately, this approach is futile, because it matters little if each component is usable within itself. Only if the components work well together will the product be viewed as usable and meeting the user's needs.



Figure 1-3 Integrated approach to product development

Fortunately, there have been advances in application development methodologies in recent years that emphasize iterated design and interdisciplinary teams. Plus there are great examples of cutting-edge products and services built around usability advantages that are dominating their markets, such as Netflix, eBay, Yahoo!, and the iPod and iPhone, as well as Whirlpool's latest line of home appliances. Their integration of components is a key contributor to their success.

Reason 5: Design and Implementation Don't Always Match

The *design* of the user interface and the technical *implementation* of the user interface are different activities, requiring very different skills. Today, the emphasis and need are on design skills, while many engineers possess the mind-set and skill set for technical implementation.

Design, in this case, relates to how the product communicates, whereas implementation refers to how it works. Previously, this dichotomy between design and implementation was rarely even acknowledged. Engineers and designers were hired for their *technical expertise* (e.g., programming and machine-oriented analysis) rather than for their *design expertise* (e.g., communication and human-oriented analysis). This is understandable, because with early generation computer languages the great challenge lay in simply getting the product to work. If it communicated elegantly as well, so much the better, but that was not the prime directive.

With the advent of new-generation programming languages and tools to automatically develop program code, the challenge of technical implementation has diminished. The challenge of design, however, has increased dramatically due to the need to reach a broader, less sophisticated user population and the rising expectations for ease of use. To use a computer analogy, the focus has moved from the inside of the machine (how it works) to the outside where the end user resides (how it communicates).

This change in focus has altered the skills required of designers. This evolution toward design and away from implementation will continue. Someday, perhaps skills such as programming will be completely unnecessary when designing a user interface.


These five reasons merely brush the surface of how and why unusable products and systems continue to flourish. More important is the common theme among these problems and misperceptions; namely that too much emphasis has been placed on the product itself and too little on the desired effects the product needs to achieve. Especially in the heat of a development process that grows shorter and more frenetic all the time, it is not surprising that the user continues to receive too little attention and consideration.

It is easy for designers to lose touch with the fact that they are not designing products per se, but rather they are designing the *relationship* of product and

human. Furthermore, in designing this relationship, designers must allow the human to focus on the task at hand — help the human attain a goal — not on the means with which to do that task. They are also designing the relationship of the various product components to each other. This implies excellent communication among the different entities designing the total product and those involved in the larger experience of using the product in a life or work context. What has been done in the past simply will not work for today's user and today's technologies.

What is needed are methods and techniques to help designers change the way they view and design products — methods that work from the outside in, from the end user's needs and abilities to the eventual implementation of the product is *user-centered design* (UCD). Because it is only within the context of UCD that usability testing makes sense and thrives, let's explore this notion of user-centered design in more detail.

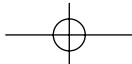
What Makes Products More Usable?



User-centered design (UCD) describes an approach that has been around for decades under different names, such as human factors engineering, ergonomics, and usability engineering. (The terms *human factors engineering* and *ergonomics* are almost interchangeable, the major difference between the two having more to do with geography than with real differences in approach and implementation. In the United States, human factors engineering is the more widely used term, and in other countries, most notably in Europe, ergonomics is more widely used.) UCD represents the techniques, processes, methods, and procedures for designing usable products and systems, but just as important, it is the philosophy that places the user at the center of the process.

Although the design team must think about the technology of the product first (can we build what we have in mind?), and then what the features will be (will it do what we want it to do?), they must also think about what the user's experience will be like when he or she uses the product. In user-centered design, development starts with the user as the focus, taking into account the abilities and limitations of the underlying technology and the features the company has in mind to offer.

As a design process, UCD seeks to support how target users actually work, rather than forcing users to change what they do to use something. The International Organization for Standardization (ISO) in standard 13407 says that UCD is "characterized by: the active involvement of users and a clear understanding of user and task requirements; an appropriate allocation of function between users and technology; the iteration of design solutions; multidisciplinary design."



Going beyond user-centered design of a product, we should be paying attention to the whole user experience in the entire cycle of user ownership of a product. Ideally, the entire process of interacting with potential customers, from the initial sales and marketing contact through the entire duration of ownership through the point at which another product is purchased or the current one upgraded, should also be included in a user-centered approach. In such a scenario, companies would extend their concern to include all prepurchase and postpurchase contacts and interactions. However, let's take one step at a time, and stick to the design process.

Numerous articles and books have been written on the subject of user-centered design (UCD) (for a list of our favorites, see the web site that accompanies this book, www.wiley.com/go/usabilitytesting). However, it is important for the reader to understand the basic principles of UCD in order to understand the context for performing usability testing. Usability testing is not UCD itself; it is merely one of several techniques for helping ensure a good, user-centered design.

We want to emphasize these basic principles of user-centered design:

- Early focus on users and their tasks
- Evaluation and measurement of product usage
- Iterated design

An Early Focus on Users and Tasks

More than just simply identifying and categorizing users, we recommend direct contact between users and the design team throughout the development lifecycle. Of course, your team needs training and coaching in how to manage these interactions. This is a responsibility that you can take on as you become more educated and practiced, yourself.

Though a goal should be to institutionalize customer contact, be wary of doing it merely to complete a check-off box on one's performance appraisal form. What is required is a systematic, structured approach to the collection of information from and about users. Designers require training from expert interviewers before conducting a data collection session. Otherwise, the results can be very misleading.

Evaluation and Measurement of Product Usage

Here, emphasis is placed on behavioral measurements of ease of learning and ease of use very early in the design process, through the development and testing of prototypes with actual users.

Iterative Design and Testing

Much has been made about the importance of design iteration. However, this is not just fine-tuning late in the development cycle. Rather, true iterative design allows for the complete overhaul and rethinking of a design, through early testing of conceptual models and design ideas. If designers are not prepared for such a major step, then the influence of iterative design becomes minimal and cosmetic. In essence, true iterative design allows one to “shape the product” through a process of design, test, redesign, and retest activities.

Attributes of Organizations That Practice UCD

User-centered design demands a rethinking of the way in which most companies do business, develop products, and think about their customers. While currently there exists no cookie-cutter formula for success, there are common attributes that companies practicing UCD share. For example:

- Phases that include user input
- Multidisciplinary teams
- Concerned, enlightened management
- A “learn as you go” perspective
- Defined usability goals and objectives

Phases That Include User Input

Unlike the typical phases we have all seen in traditional development methodologies, a user-centered approach is based on receiving user feedback or input during each phase, prior to moving to the next phase. This can involve a variety of techniques, usability testing being only one of these.

Today, most major companies that develop technology-based products or systems have product lifecycles that include some type of usability engineering/human factors process. In that process, questions arise. These questions and some suggested methods for answering them appear in Figure 1-4.

Within each phase, there will be a variety of usability engineering activities. Note that, although this particular lifecycle is written from the viewpoint of the human factors specialist’s activities, there are multiple places where collaboration is required among various team members. This leads to our next attribute of organizations practicing UCD.

A Multidisciplinary Team Approach

No longer can design be the province of one person or even of one specialty. While one designer may take ultimate responsibility for a product’s design, he

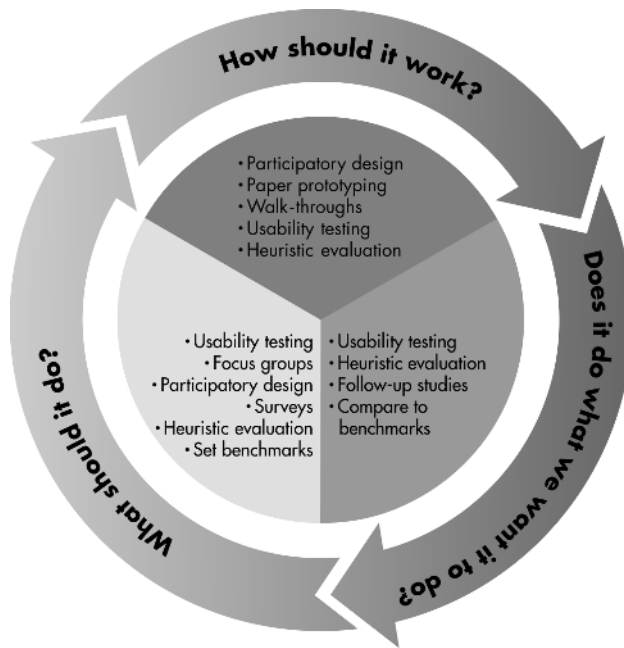


Figure 1-4 Questions and methods for answering them

or she is not all-knowing about how to proceed. There are simply too many factors to consider when designing very complex products for less technical end users. User-centered design requires a variety of skills, knowledge, and, most importantly, information about the intended user and usage. Today, teams composed of specialists from many fields, such as engineering, marketing, training, user-interface design, human factors, and multimedia, are becoming the norm. In turn, many of these specialists have training in complementary areas, so cross-discipline work is easier and more dynamic than ever before.

Concerned, Enlightened Management

Typically, the degree to which usability is a true corporate concern is the degree to which a company's management is committed to following its own lifecycle and giving its guidelines teeth by holding the design team accountable. Management understands that there are financial benefits to usability and market share to be won.

A "Learn as You Go" Perspective

UCD is an evolutionary process whereby the final product is shaped over time. It requires designers to take the attitude that the optimum design is acquired through a process of trial and error, discovery, and refinement. Assumptions

about how to proceed remain assumptions and are not cast in concrete until evaluated with the end user. The end user's performance and preferences are the final arbiters of design decisions.

Defined Usability Goals and Objectives

Designing a product to be useful must be a structured and systematic process, beginning with high-level goals and moving to specific objectives. You cannot achieve a goal — usability or otherwise — if it remains nebulous and ill-conceived. Even the term *usability* itself must be defined with your organization. An operational definition of what makes your product usable (tied to successful completion criteria, as we will talk about in Chapter 5) may include:

- Usefulness
- Efficiency
- Effectiveness
- Satisfaction
- Accessibility

Thus bringing us full circle to our original description of what makes a product usable. Now let's review some of the major techniques and methods a usability specialist uses to ensure a user-centered design.

What Are Techniques for Building in Usability?

UCD comprises a variety of techniques, methods, and practices, each applied at different points in the product development lifecycle. Reviewing the major methods will help to provide some context for usability testing, which itself is one of these techniques. Please note that the order in which the techniques are described is more or less the order in which they would be employed during a product's development lifecycle.

Ethnographic Research

Ethnographic research borrows techniques from anthropology. It involves observing users in the place where they would normally use the product (e.g., work, home, coffee bar, etc.) to gather data about who your target users are, what tasks and goals they have related to your planned product (or enhancements), and the context in which they work to accomplish their goals. From this qualitative research, you can develop user profiles, personas (archetype users), scenarios, and task descriptions on which you and the design team can base design decisions throughout the development lifecycle.

Participatory Design

Less a technique and more an embodiment of UCD, participatory design employs one or more representative users on the design team itself. Often used for the development of in-house systems, this approach thrusts the end user into the heart of the design process from the very commencement of the project by tapping the user's knowledge, skill set, and even emotional reactions to the design. The potential danger is that the representative users can become *too* close to the design team. They begin to react and think like the others, or by virtue of their desire to avoid admonishing their colleagues, withhold important concerns or criticism.

A variation on this technique is to arrange short, individual workshops where users, designers, and developers work together on an aspect of design. For example, users, designers, and engineers using workable models, work together to determine the best size and shape for the product.

Focus Group Research

Use focus group research at the very early stages of a project to evaluate preliminary concepts with representative users. It can be considered part of "proof of concept" review. In some cases it is used to identify and confirm the characteristics of the representative user altogether. All focus group research employs the simultaneous involvement of more than one participant, a key factor in differentiating this approach from many other techniques.

The concepts that participants evaluate in these group sessions can be presented in the most preliminary form, such as paper-and-pencil drawings, storyboards, and/or more elaborate screen-based prototypes or plastic models. The objective is to identify how acceptable the concepts are, in what ways they are unacceptable or unsatisfactory, and how they might be made more acceptable and useful. The beauty of the focus group is its ability to explore a few people's judgments and feelings in great depth, and in so doing learn something about how end users think and feel. In this way, focus groups are very different from — and no substitute for — usability tests. A focus group is good for general, qualitative information but not for learning about performance issues and real behaviors. Remember, people in focus groups are reporting what they feel like telling you, which is almost always different from what they actually do. Usability tests are best for observing behaviors and measuring performance issues, while perhaps gathering some qualitative information along the way.

Surveys

By administering surveys you can begin to understand the preferences of a broad base of users about an existing or potential product. While the survey

cannot match the focus group in its ability to plumb for in-depth responses and rationale, it can use larger samples to generalize to an entire population. For example, the Nielsen ratings, one of the most famous ongoing surveys, are used to make multimillion-dollar business decisions for a national population based on the preferences of about 1500 people. Surveys can be used at any time in the lifecycle but are most often used in the early stages to better understand the potential user. An important aspect of surveys is that their language must be crystal clear and understood in the same way by all readers, a task impossible to perform without multiple tested iterations and adequate preparation time. Again, asking people about what they do or have done is no substitute for observing them do it in a usability test.

Walk-Throughs

Once you have a good idea who your target users are and the task goals they have, walk-throughs are used to explore how a user might fare with a product by envisioning the user's route through an early concept or prototype of the product. Usually the designer responsible for the work guides his or her colleagues through actual user tasks (sometimes even playing the role for the user), while another team member records difficulties encountered or concerns of the team. In a structured walk-through, as first developed by IBM to perform code reviews, the participants assume specific roles (e.g., moderator, recorder) and follow explicit guidelines (e.g., no walk-through longer than two hours) to ensure the effectiveness of the effort. Rather than the designer taking on the role of the user, you may want to bring in a real user, perhaps someone from a favored client.

Open and Closed Card Sorting

Use card sorting to design in “findability” of content or functionality. This is a very inexpensive method for getting user input on content organization, vocabulary, and labeling in the user interface. You can either give participants cards showing content without titles or categories and have the users do the naming (an open card sort), or give participants preliminary or preexisting categories and ask participants to sort content or functions into those (a closed sort).

Paper Prototyping

In this technique users are shown an aspect of a product on paper and asked questions about it, or asked to respond in other ways. To learn whether the flow of screens or pages that you have planned supports users' expectations, you may mock up pages with paper and pencil on graph paper, or create line

drawings or wireframe drawings of screens, pages, or panels, with a version of the page for each state. For example, if the prototype is for a shopping cart for an e-commerce web site, you can show the cart with items, as items are being changed, and then with shipping and taxes added. (Or, you may simply decide to have the participant or the “computer” fill these items in as the session progresses.)

To learn whether the labels help users know what to expect next, and if the categories you have planned reflect how users think and talk about tasks, you can show the top-level navigation. As the participant indicates the top-level choice, you then show the next level of navigation for that choice. The process continues until the user has gone as deeply into the navigation as you have designed and prepared for the sessions.

Or, you may simply ask participants about the prototype you have created. The questions can range from particular attributes, such as organization and layout, to where one might find certain options or types of information.

The value of the paper prototype or paper-and-pencil evaluation is that critical information can be collected quickly and inexpensively. One can ascertain those functions and features that are intuitive and those that are not, *before one line of code has been written*. In addition, technical writers might use the technique to evaluate the intuitiveness of their table of contents before writing one word of text. The technique can be employed again and again with minimal drain on resources.

Expert or Heuristic Evaluations

Expert evaluations involve a review of a product or system, usually by a usability specialist or human factors specialist who has little or no involvement in the project. The specialist performs his or her review according to accepted usability principles (heuristics) from the body of research, human factors literature, and previous professional experience. The viewpoint is that of the specific target population that will use the product.

A “double” specialist, that is, someone who is an expert in usability principles or human factors as well as an expert in the domain area (such as healthcare, financial services, and so on, depending on the application), or in the particular technology employed by the product, can be more effective than one without such knowledge.

Usability Testing

Usability testing, the focus of this book, employs techniques to collect empirical data while observing representative end users using the product to perform realistic tasks. Testing is roughly divided into two main approaches. The first approach involves formal tests conducted as true experiments, in order to

confirm or refute specific hypotheses. The second approach, a less formal but still rigorous one (and the one we emphasize in this book), employs an iterative cycle of tests intended to expose usability deficiencies and gradually shape or mold the product in question.

Follow-Up Studies

A follow-up study occurs after formal release of the product. The idea is to collect data for the next release, using surveys, interviews, and observations. Structured follow-up studies are probably the truest and most accurate appraisals of usability, because the actual user, product, and environment are all in place and interacting with each other. That follow-up studies are so rare is unfortunate because designers would benefit immensely from learning what happened to the product that they spent two years of their lives perfecting. Sales figures, while helpful, add nothing to one's knowledge of the product's strengths and weaknesses.

This is not a definitive list of methods by any means, and it is meant merely to provide the reader with an appreciation for the wealth of techniques available and the complexity involved in implementing a UCD approach. It is a rare organization that performs all of these techniques, and just as few conduct them in their pure form. Typically, they are used in altered and combined form, as the specific needs and constraints of a project dictate. For more about these techniques, check out our list of resources on the web site that accompanies this book at www.wiley.com/go/usabilitytesting.

Now let's take a closer look at one of the most renowned techniques of all the ones discussed, and the focus of this book, usability testing, in Chapter 2.