Chapter 12

What Paper Is (and Isn't) Good For

Saying that paper prototyping is a good technique is like saying that hiking boots are a good type of footwear—you have to specify *for what purpose.* Hiking boots are great for a walk in the woods, but they'd suck for ballet. In this chapter I discuss the kinds of problems paper prototyping is likely and unlikely to find in an interface. I also describe some issues for which usability testing in general (whether paper or computer) is not the best means of discovery.

Dimensions of a Prototype

So exactly what is a prototype, anyway? I know that's a funny question to be posing this far through a book on paper prototyping, but understanding the nature of a prototype helps you decide what method is suitable for what you need to learn. In essence, a prototype is a representation of a design concept, but that definition is too high-level to provide much insight, so let's break it down.

Prototype Fidelity: A Misleading Concept

First, I want to describe a way of classifying prototypes that I have found to be more problematic than useful, although I still hear it used fairly often. Years ago I was taught to think in terms of "high-fidelity" (hi-fi) and "low-fidelity" (low-fi) prototypes. In his paper, Tullis (1990) defines fidelity* by how it appears to the

* Strangely enough, a few years after he started using the term *fidelity,* Tom Tullis went to work at Fidelity Investments. Coincidence? Clairvoyance? Or even . . . conspiracy?

user, which may or may not reflect the extent of the working technology supporting it. Hi-fi prototypes were coded to look and act a lot like the real thing, perhaps using some kind of prototyping software. They were often good enough to be mistaken for the actual interface. Low-fi prototypes, on the other hand, were obviously faked—no one who sees a Computer shuffling pieces of paper thinks they're looking at software.

Although fidelity is a useful concept, it's misleading to apply it to an entire prototype because fidelity takes what is actually several dimensions and rolls them up into one. This can lead to confusion—person A's definition of a low-fi prototype may be significantly different than person B's without either of them being aware of it. To avoid this confusion, I've stopped using the term *fidelity* when I'm referring to an entire prototype. I use it only when I'm referring to a specific aspect of the prototype; for example, a screen created using graphics software has a high-fidelity *look*. And whenever I hear "low-fidelity prototype," I'm careful to find out what the speaker or author means by that term.

Four Dimensions of a Prototype

So let's expand this one-dimensional view of prototyping. There are different ways to look at prototypes, but the one I've found most useful consists of four dimensions—breadth, depth, look, and interaction (I got this idea from the 1996 paper by Virzi, Sokolov, and Karis). As you'll see, some of these prototype dimensions are more important than others in terms of helping you answer the questions you have about *your* interface.

Breadth

*Breadth** refers to the percentage of the product's functionality that is represented in the prototype. Some prototypes contain all the functionality that will be present in the real design, whereas others have only a subset. In terms of usability testing, a prototype needs sufficient breadth to cover the tasks that have been created, but usually not much more.

Breadth is a relatively easy concept to understand, but it's not especially interesting to our discussion because any kind of prototype can be broad or narrow.

---

* If you read my descriptions of breadth and depth and wonder, "Isn't she talking about horizontal and vertical prototypes?" the answer is essentially yes. The terms *horizontal* and *vertical* are common in prototyping literature—I chose to use *breadth* and *depth* simply because it's more expedient to use nouns for the dimensions instead of adjectives.

Regardless of the method you're using, there is a fairly linear relationship between breadth and the effort required to create the prototype—if you want a broader prototype, you'll have to do more work. Although this factor is certainly important to an understanding of prototypes, for the most part I'm going to ignore breadth because it doesn't help us differentiate and choose among prototyping methods.

Depth

As Joel Spolsky said in Chapter 3, the user interface is only the tip of the software iceberg—beneath the surface are databases, networks, security, error handling, hardware, and more. The degree to which these things are in place is depth. Whereas breadth relates to the percentage of functionality that is represented, depth is the extent to which that functionality is fleshed out and, well, functional.

By *depth,* I mean not only the level of detail that's implemented but also its robustness. Say there are 10 options that users could choose from a drop-down list. One low-depth prototype might be hard-coded to ignore what users pick; it assumes that a particular choice has been made. Another low-depth prototype might allow users to pick any of the 10 options, but half of them cause the prototype to crash because they are incompatible with other choices users have made. From the perspective of users who choose one of those unsupported options, the net effect is the same: The interface doesn't do what they told it. On the other hand, a high-depth prototype would have logic and error handling to gracefully prevent, detect, or explain any combination of choices that didn't make sense. Thus, the high-depth prototype would proceed in accordance with what users chose.

Depth turns out to be a key factor in usability testing because it affects the amount of exploration the user can do. Experimentation is a part of learning, and depth (along with breadth) is the factor that makes it possible. Without sufficient depth, the user must be constrained to walk a narrow path and/or make numerous assumptions about whether the interface would behave as expected (not to mention the time spent waiting for a crashed prototype to be restarted). In any case, there's a good chance that important usability issues might be missed if the user isn't getting the full experience.

Look

The look factor is the easiest to determine by inspection and is probably what gave rise to the original concept of prototype fidelity. *Look* refers to whether the visual aspects of the prototype accurately represent the intended appearance, including fonts, colors, and graphics. Something hand-drawn would obviously get a low score

in the look department, and even printed-out color screen shots may earn only a medium score if they look substantially different from how those same screens will appear on a computer monitor—it's possible for a printout to look better than the same thing on a monitor, or vice versa. Also note that software prototypes don't automatically earn a high score for look—they may use different colors, graphics, spacing, and so on than the real thing. They just happen to have recognizable fonts and straight lines.

When it comes to look, don't automatically assume that a higher score is better. As discussed in Chapter 3, a rough-looking prototype can encourage users and other stakeholders to respond in a more creative manner. Especially in the early stages of a project, you might deliberately choose a prototyping method with a rough look to get this benefit.

Interaction

Interaction refers to the way that the prototype handles the inputs and outputs with the user—are the I/O methods simulated in a realistic manner? On a desktop computer, the input devices are typically the keyboard and mouse, and the monitor is the output. For a piece of medical equipment, the user might turn dials. For a handheld device, its weight and the spacing of the buttons might be important.

Time is an integral part of interaction, so consider things such as response time, cursor changes, flashing lights, and animation. There also may be senses other than sight involved—some interfaces use sound, and physical devices have tactile components.*

Note: Some people confuse interaction with depth—the former refers to the *methods* used to interact, whereas the latter is the *degree* to which the prototype responds as the real interface would.

Comparing Prototyping Methods

Now let's consider four different methods of prototyping and how well they do at representing the dimensions of look, similarity of interaction, and depth. Of course, these are not the only kinds of prototypes—I chose them because they illustrate interesting differences. For this example, I'll assume that the interface being prototyped is an e-commerce Web site, since that example is easy for people

* I hope to be safely retired from the usability business before I'm asked to test an interface incorporating smell and taste!

Table 12.1 Comparison of Prototyping Methods for an e-Commerce Web Site

Kind of Prototype	*Look*	*Interaction*	*Depth*
Working version (e.g., Dreamweaver)	Medium-high	High	Low-high
Slide show (e.g., Microsoft PowerPoint)	Medium-high	Medium	Low-medium
Paper prototype	Low-medium	Low	Medium-high
DENIM	Low	Medium	Low-medium

to relate to. Table 12.1 shows a summary of how the four methods compare; you may want to refer back to it as you read the following sections.

Working Version

Any less-than-complete version of the site can be considered a working prototype if it supports the tasks needed for usability testing. In the case of a Web site, the interface might be created in a WYSIWYG editor such as Dreamweaver or FrontPage.

The *look* depends on the extent to which graphic design was included—the prototype might be text-only or it might be designed down to the last pixel, but it's usually at least medium because it has straight lines and legible text. The *interaction* of a working prototype is nearly the same as a functioning site (although the response time might be different compared with the real Web server), so it gets a high score. The *depth* can range from low to high because it depends on the amount of time spent coding the site's behavior, such as getting product information from databases or processing an order form. Some so-called working prototypes are just a linked set of screens with nothing beneath them (low depth), whereas others have all of the underlying functionality in place to the point where they're indistinguishable from the real site.

Slide Show

Some prototypes are created as fairly static "slide slows," for example, by pasting screen shots into Microsoft PowerPoint. In the simplest variation, clicking the mouse anywhere advances to the next page, although it is possible to provide more complex branching. (For interfaces other than Web sites, something similar

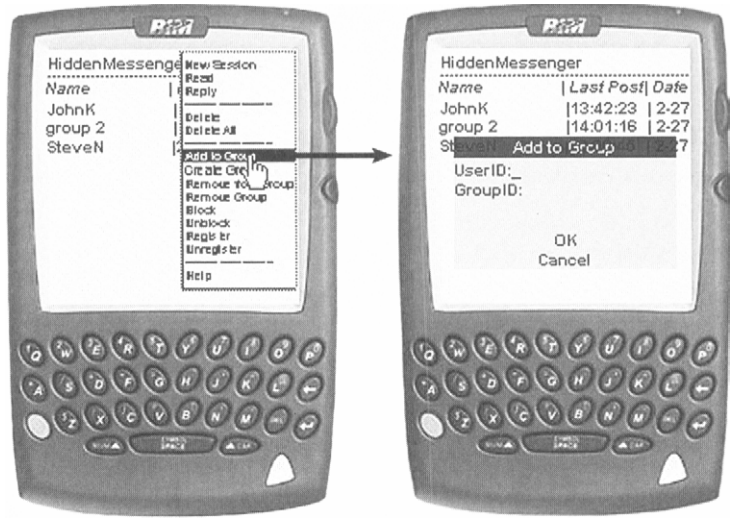

Figure 12.1 Two pages from a fixed sequence HTML prototype of a PDA application shown side-by-side. This runs in a browser, but because there are only one or two hot spots per page, in essence this prototype behaves like a slide show.

can be done using screen shots in linked HTML files as shown in Figure 12.1—naturally, if you did this for a Web site it would *be* a Web site as opposed to a slide show of a different type of interface.)

The *look* is usually in the medium-high range: high if screen shots are used, medium if wireframes are used. I've even heard of people using scanned sketches, which would of course be low. The *interaction* is only medium because you can't type, only click. The slide show usually has low to medium *depth*—the user is constrained to a subset of all the possible paths through the interface, and the controls don't work.

The main drawback to the slide-show approach is that users can't explore the interface, although you might be able to have some useful discussion based on the concepts they can see. But the lack of interactivity will limit the amount and nature of the feedback you can get—people stop exploring after the first few dead ends. You can only ask, "What do you *think* that would do?" a few times before the enthusiasm tapers off. This kind of prototype may be sufficient to expose relatively obvious errors or omissions in the design, but if you get vague positive feedback of the "Looks good!" variety, take it with a large grain of salt. The From the Field box illustrates why I believe that low-depth prototypes are more suited for demos or groups reviews than for usability testing.

From the Field: Usability Testing a Slide Show

"When I was new to the usability profession, I had a bad experience when I tried to conduct usability tests using a series of linked HTML files. The intent was to ask users what they'd click on and what they'd expect to see, and then proceed to the next page once they'd found the 'right' link. But once the users caught on that there were only one or two live links per page, the usability test quickly deteriorated into a game of 'find the hot spot.' The sessions were a complete waste of time. It really damaged the credibility of usability testing at my company, and for subsequent efforts I faced an uphill battle."

Anonymous

Paper Prototype

The *look* dimension can vary a lot for a paper prototype—hand-drawn Web pages would get a low score, grayscale printouts of professionally designed screens would be medium, and color printouts might even be high. The *interaction* factor has obvious differences—touching isn't exactly like clicking, writing isn't typing, and the human element is artificial. Although working with a paper prototype can be highly interactive, it's quite different from working with a computer, so the paper prototype gets a low score here. But what's interesting about paper prototypes is their *depth* because people take up the slack—the Computer decides what to do based on the user's inputs. Even if those inputs weren't what was expected, the Computer can (to borrow my own words from the earlier discussion of depth) gracefully prevent, detect, or explain any combination of choices that didn't make sense. So paper prototypes often have good depth.

It is worth mentioning that paper prototypes support breadth only up to a point. Although in theory you could create a 5000-page Web site out of paper, it wouldn't be practical. So if an interface (or more precisely, a particular task for that interface) requires a large number of screens, it may become overwhelming for the Computer. Although I've never counted, I estimate that most paper prototypes I've worked on have had anywhere from a few dozen to a couple hundred screens, which provided sufficient breadth and depth for the tasks the product team needed to test.

Note: Although you might hear both the PowerPoint slide show and the paper prototype referred to as "low-fidelity" prototypes, Table 12.1 reveals that they tend to have opposite characteristics. This further illustrates why the term *low-fidelity* can be misleading.

DENIM

Interestingly, there are some computer-based sketching tools that let you draw screens by hand (using a pen-and-tablet input device) and link them into functioning computer-based prototypes. One such tool is DENIM, which can be downloaded from *guir.berkeley.edu/projects/denim*. The DENIM project is headed by Dr. James Landay of the University of California at Berkeley. The idea behind DENIM is to provide Web site designers with a unified approach to various tools, such as site maps, storyboards, and individual page design. DENIM supports sketching input and a visual language—draw an X on top of a widget, and it disappears. As of 2002, DENIM does not provide support for all the widgets and interaction found in HTML forms and traditional graphic user interfaces, although enhancements are planned. See Figure 12.2 for an example of a prototype created using DENIM.

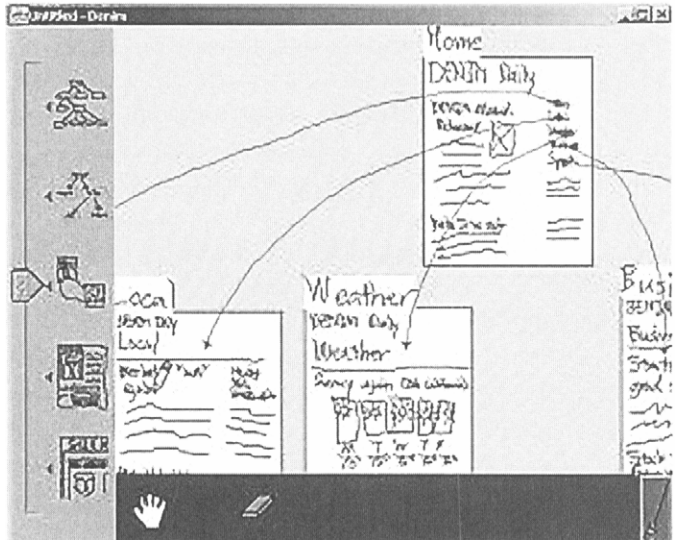

Figure 12.2 Computer-based sketching tools like DENIM allows you to draw using a pen input device and then link pages together to make a "hand drawn" prototype, which can then be tested on a computer.

Thus, a DENIM prototype has a deliberately rough *look* because its goal is to support sketching. I would rate its *interaction* capability as medium and its *depth* as low to medium because it doesn't fully support all types of interface widgets, but these scores might change as more functionality is added to DENIM.

When it comes to usability testing, a tool like DENIM may not offer any compelling benefits over a paper prototype other than removing the human Computer from the equation. And that may not even be its intent—much of DENIM's functionality is aimed toward providing a productive environment for designing, not necessarily for usability testing. To me, one of the intriguing aspects of a tool like DENIM is that it may enable researchers to tease apart the effects due to the *medium* of the prototype (paper or computer) and the *completeness* of the design.

Methods Don't Stand Alone—What Are *You* Prototyping?

Table 12.1 was created in the context of an e-commerce site, so it may or may not be relevant to what you're doing. The prototyping methods alone don't completely determine the score for each dimension—it depends on what you're prototyping. For example, you could make an HTML prototype of a handheld device as shown in Figure 12.1 and its interaction would be only medium because clicking a button with a mouse isn't the same as poking it with your finger. So if you're prototyping a handheld device, you might conclude that HTML prototypes get a medium score for interaction. On the other hand, if you're building a Web site, an HTML prototype would get a high score for interaction.

Here's an ironic example: You'd think that a paper prototype would get high marks in interaction if you were prototyping a touch screen device, but this isn't the case. Touch screens are complicated by factors such as calibration and parallax, which affect how the computer and human interpret each other's behavior. In practice, there are some important subtleties about touch screens that you can't adequately simulate with a paper prototype. Funny, huh?

So what are you prototyping? If you're feeling ambitious, you might make your own version of Table 12.1 for your interface and the prototyping methods you're considering.

Which Dimensions Matter?

As discussed in Chapter 5, when planning a usability test, you should always begin by asking what things you're most interested in learning. Table 12.2 shows some categories of questions (applicable to most software and Web sites) and which of the four prototyping dimensions are most important in answering them. Of course, you will have plenty of questions that don't appear in this table, especially if you are testing a different kind of interface—the strategy here is to think about which dimensions you need get the answers to *your* questions.

Table 12.2 Categories of Interface Questions and Paper Prototype Dimensions

Category of Questions	*Dimensions Needed*				*Examples*
	Breadth	*Depth*	*Look*	*Interaction*	
Concepts and terminology	✓	✓			◈ Do users understand the terms used in the interface? ◈ Are there concepts they gloss over or misconstrue? ◈ For new concepts, is the user able to figure them out?
Navigation, work flow, task flow	✓	✓			◈ Are users able to find their way around? ◈ Will they search, use links, or both? ◈ If there's a work flow or sequence of steps, does it match what users expect? ◈ Do you have the fields in the order that users expect? ◈ Do they have to keep flipping back and forth between screens? ◈ Does the interface ask for inputs that the users don't have, or don't want to enter?
Content	✓	✓	✓*		◈ Does the site/interface provide the right information for users to make decisions? What things do they look for? ◈ Is it useful and/or interesting to them?

Table 12.2 Categories of Interface Questions and Paper Prototype Dimensions—cont'd

Category of Questions	*Dimensions Needed*				*Examples*
	Breadth	*Depth*	*Look*	*Interaction*	
					◈ Is there extra content that they don't need, or that annoys them?
Documentation, help	✓	✓	✓*		◈ What assistance does the user need to successfully complete tasks? ◈ What's the best way to provide that information? ◈ Can users quickly find the information they need, and make sense of it?
Requirements, functionality	✓	✓	✓*	✓*	◈ Does the interface do the right set of things for its target audience? ◈ Do users have additional needs that aren't being satisfied? ◈ Is there anything you could do to make the user's life easier? ◈ Are you planning to implement something that users don't really need?
Screen layout	✓		✓		◈ Is the amount of information per screen overwhelming, not enough, or about right? ◈ Do users miss seeing something that's important? ◈ Are there elements that need to be brought out more in the visual design? Any that distract the user? ◈ Has white space been used effectively? Images? ◈ Do we have the right stuff "above the fold?"
Brand	✓	✓	✓		◈ Does the interface reflect the qualities that the company wants to convey? ◈ Does the user experience match what the designer intended?

Continued

Table 12.2 Categories of Interface Questions and Paper Prototype Dimensions—cont'd

Category of Questions	*Dimensions Needed*				*Examples*
	Breadth	*Depth*	*Look*	*Interaction*	
					◈ Are there frustrations or obstacles that can be removed? ◈ Do users like it?
Colors, fonts, and other graphic elements			✓		◈ Can users see/read everything well enough? ◈ Do the most important elements stand out? ◈ Are there any considerations pertaining to lighting, vision difficulties, or color blindness? ◈ Is the interface aesthetically appealing? ◈ Do users understand what the icons mean?
Widgets and controls				✓	◈ Do the rollover menus work for users or do they have trouble? ◈ Do users notice the status line message? ◈ Can they figure out what the cursor changes mean? ◈ Will multiple windows be a problem? ◈ Do the slider controls have the right granularity? ◈ Did we pick the best keyboard shortcuts?
Response time, performance metrics				✓	◈ Does the system respond quickly enough to satisfy users? ◈ Do the pages load fast enough? ◈ Does the display change quickly enough when the user manipulates a control? ◈ Are there any download or processing delays that users might find annoying or unacceptable? ◈ How quickly can users complete this task?

Table 12.2 Categories of Interface Questions and Paper Prototype Dimensions—cont'd

Category of Questions	*Dimensions Needed*				*Examples*
	Breadth	*Depth*	*Look*	*Interaction*	
Real-world use	All these factors, plus the real context of use				◈ How does this tool fit with others that users have? ◈ What things will annoy power users after 6 months? ◈ Which of these functions are people really going to use? ◈ What happens when the user is interrupted mid-task?

* Sometimes.

Notice that breadth appears in about half of the columns in this table, so obviously it's an important factor in a prototype—there has to be enough of the interface for users to get a realistic idea of how it works. When it comes to usability testing, breadth is essentially a function of the tasks that you have created. As mentioned earlier, breadth can be incorporated in any kind of prototype and thus it's not useful as a means of choosing among methods, except in the unusual case where you need more breadth than a paper prototype (or rather, its human Computer) can handle.

What if your team has questions in most of the categories from Table 12.2? This happens often. Does it mean that you need to wait until the real product can be tested? Usually not. It might be helpful to divide your questions into "now" and "later" piles—those that can be answered with a paper prototype now and those that require a prototyping method (or the real interface) that would take more time. If the "now" pile is big enough to make paper prototyping worthwhile, go for it. (The next chapter discusses a somewhat more quantitative way of making this decision.)

Table 12.1 illustrates that paper prototyping's main strength is depth—the human Computer can simulate much of *what* the interface will do. So if depth is an important factor in researching a question, a paper prototype is likely to provide

some answers. On the other hand, paper prototypes don't do so well in terms of interaction because it's hard for the Computer to accurately mimic exactly *how* a machine will respond. The look factor is somewhere in the middle because a paper prototype can be either hand-drawn or use highly designed screen shots.

Let's look at each of these categories—depth, look, and interaction—more closely. For purposes of this detailed discussion, I'm sticking with the example of an e-commerce site. Naturally, if you are designing a different type of interface, you should scrutinize whether the logic I'm presenting in each section applies to your situation or not.

What Paper Prototypes Will Likely Find (Depth Issues)

Looking at Table 12.2, there are several rows that have Depth as one of the factors. As a generalization, these are the types of questions that paper prototypes are well suited to answer.

Concepts and Terminology

Unclear concepts and confusing terminology are common—I don't think it's an exaggeration to say that I've seen these types of problems in every interface I've ever tested. Sometimes even one ambiguous word can prevent users from completing a task. For example, one user who was testing an e-commerce site refused to buy a cordless drill because it was described as a "kit" and he couldn't figure out what the kit included besides the drill. To probe problems pertaining to concepts and terminology, you need actual wording and often the supporting help or documentation as well, and these are depth factors. On the other hand, look and interaction usually aren't as important; if users don't understand the meaning of a field label or they need to see an example, this will be apparent regardless of whether the label is handwritten or typed.

Navigation, Work Flow, Task Flow

Breadth is an important dimension in determining whether users can navigate the interface or perform their work tasks because there is often more than one

correct path, as well as a large number of incorrect ones. One way to think of it is that you should provide enough of the interface to confuse users. (More properly, the goal is to ensure that users aren't confused, but if you make a simplified version of the interface you can't tell when you've accomplished that goal.) To evaluate navigation, it is essential for users to have the ability to go where *they* want next, which may or may not match the path you envisioned when you created the interface. Depth is important because users sometimes step off the correct path and it's useful to know whether the interface succeeds in guiding them back onto it. Or sometimes the users' path is different but also correct—with a low-depth prototype, you may not be able to watch them do it their way.

Content

Content issues pertain to interfaces where users seek information and make decisions based on it (as opposed to an interface used primarily for data input). Similar to concepts, content questions are primarily a function of breadth and depth. In usability testing, if you want to watch users make decisions, it's often important to show them the real content, not greeked or approximated by wording such as "guarantee goes here." As discussed in Chapter 7, it may be easy to draw the content by hand or print it out, so a paper prototype doesn't have any inherent advantage or disadvantage over other methods when it comes to content—it depends mostly on where your content is coming from.

Sometimes you need breadth to understand content issues. If an interface or its content is very complex or large in scope, a paper prototype may fall short of the degree of realism you need. For example, a music Web site might offer thousands of albums—a paper prototype of the site has no realistic way to represent how all this content would appear to users or how they would interact with it. Although you might learn many useful things from paper prototype testing, there may be some questions you can't answer, such as whether users in real life would prefer to search or browse to find music.

To what extent is look part of content? If you're selling flowers online, color is indeed part of the content because users care whether the roses are red or yellow. But if you're selling life insurance, color has little to do with the content of the site, although it will undoubtedly be present in the graphic design. Similar logic could be applied for every aspect of your interface's visual design—the more of these factors that pertain to the content, the more concerned you're likely to be about the look factors that are discussed later in this chapter.

Documentation/Help

The issues for documentation and help are similar to those for concepts and terminology, which is typically what the doc/help are helping *with*. When a paper prototype of an interface is being tested, it's possible to gather useful information about what questions users have, what information is needed to answer them, and equally important, what doesn't need to be explained. By listening to the terminology they use, you can glean terms for your index.

If along with the interface prototype you are also testing a prototype of the documentation, its look may matter because formatting can aid the user's ability to find and use information. But it's usually not important to duplicate the final appearance of the help system or manual—drafting the content using your favorite word processor and adding a few headings is sufficient for a paper prototype test.

Requirements/Functionality

Most functionality is expressed in terms of what the interface does, rather than how it looks while it's doing it or the specific interaction methods used (although there could be exceptions to this rule). But it's pretty much a no-brainer that if you've built the wrong thing, it's going to fail with its intended audience. Ideally, you should have a decent understanding of user requirements and functionality before you begin the design of the interface. But if you've missed a requirement or gotten one wrong, a paper prototype can help you find it.

Screen Layout

You might think I've mistakenly put this section under paper prototype's strengths because questions pertaining to screen layout seem like they'd require a prototype with a realistic look. But many questions don't—a designer first needs to understand what elements should appear on the screen and also something about their relative priority. For example, if users who sign up for a service wonder whether there's a cost, you'd want to make sure that the word "free" jumped out at them in the final design. But until you did some usability tests, you wouldn't realize users had this question.

In essence, the findings from paper prototype tests can provide useful *inputs* to the process of designing screens because they will help you determine whether you have the right set of information in the right order, whether you've forgotten

anything, and what elements need to be emphasized. On the other hand, once you're to the point of trying to determine whether you've used enough white space or if the Next button stands out enough, then you do need a more realistic visual representation.

Brand

Brand applies to the user's whole experience with the site or product, not just its appearance. A car may look great in the showroom, but you won't notice a problem with its suspension unless you drive it. Breadth and depth are often important to the brand, sometimes even more important than the look. If an interface is tastefully designed but doesn't let users do what they want, ultimately it won't create or reinforce a good impression in the user's mind.*

Although one can no longer argue, as some usability specialists did back in the mid-1990s, that graphic design is unimportant, neither can you take a flawed user experience and fix it by adding rainbows and puppy dogs. (I realize that no self-respecting graphic designer thinks this way, but occasionally I run across other people who do.)

A rough-looking paper prototype with sufficient breadth and depth can go a surprisingly long way toward revealing the ways in which the interface will support or interfere with the brand image that the company would like to convey (see the Of Interest box on p. 276). Once you have a solid foundation in terms of making the interface do what users need, then the look can build upon it in a way that supports and enhances the user experience.

What Paper Prototypes May Find (Look Issues)

When it comes to look, you first need to examine the questions you have that are specific to the visual aspects of the interface. As described in the previous section, for many types of questions, the look plays only a secondary role. But naturally many aspects of the visual design can be important depending on the circumstances. Here are three examples.

---

* Remember that famous line from the movie *2001: A Space Odyssey* where the computer HAL seizes control of the ship? It didn't matter one iota that HAL politely said, "I'm sorry Dave . . ." before dropping the bombshell " . . . but I can'let you do that."

Of Interest . . . Look and Brand—Not Inseparable

Look isn't always the most important factor in conveying a brand. I once tested two versions of a prerelease financial Web site. The first version tested had all the navigation and most of the text but lacked the graphic design—all pages had black text on a white background with no images. In other words, it was functional but ugly. The second version, which was ready a couple of months later, added the professionally designed colors, fonts, and images of smiling people.

We tested each version with half a dozen users in each of two populations (consumers and small business owners). At the end of the session, users were given a list of adjectives and asked to agree or disagree with whether they described the site. The adjectives represented the brand that the company wanted their site to have: reliable, accurate, honest, and so on. Interestingly, the brand perception didn't seem to change very much after the graphic design was added—users perceived the same flaws in the site's content and organization despite the fact that the second version looked much nicer than the first.

(Caveat: This study had a small number of users and there were other differences between the two versions besides the graphic design. In other words, this is an anecdote, not scientific proof. But it's intriguing that we saw no evidence of what we expected—that the brand perception would be improved by the addition of professional graphic design. I would love to see more formal research to help us understand when and how graphic design has the sort of positive effect that we instinctively believe it has.)

- **Alignment.** One paper prototype used a tree structure with multiple levels of indenting. Because the items in the hand-drawn paper prototype didn't line up very well, users had to work harder to understand the hierarchy that was being represented. After the test, one of the users commented that it probably would have been clearer on a computer screen, and I agreed with him.
- **Font size.** A colleague told me a funny story about font size in a Web site intended for use by teenagers. After their initial launch, the designers discovered that the large font they'd chosen could be read by a parent halfway across the room, which was unacceptable to the teenage target audience!
- **Photographs.** Some interfaces have photographs as part of the content, such as a picture of a sweater on a clothing Web site. If photographs are important, you can paste them onto an otherwise hand-drawn interface. (This is what The MathWorks did with their prototype described in Chapter 2.) On the other hand, if the interface uses photographs for a primarily decorative purpose, it's probably safe to omit them from a paper prototype.

Other things that are hard to simulate to a high degree of accuracy with a hand-drawn prototype include colors, shading, contrast, readability, white space, icons,

dense layouts, and complex graphics. Because I find that this is pretty self-evident to most people, I won't belabor the obvious by providing lots of examples. If you have important questions that *depend on* specific visual elements like these, then a hand-drawn paper prototype probably isn't going to answer them to your satisfaction. (And if there are enough differences between how your design appears on paper as compared with on the screen, maybe printed screen shots aren't good enough either; in that case, it would be important to test the design on a computer.)

What Paper Prototypes Won't Find (Interaction Issues)

Paper prototypes are a rather blunt instrument when it comes to detecting many interaction problems. Following are several types of problems I've seen in testing real interfaces that I doubt I would have found with paper. This list is by no means complete—I'm willing to bet that there are many others.

Small Changes

On some interfaces, small areas of the screen change in response to the user's actions. For example, some applications use the status line area at the bottom of the window to display messages, or an e-commerce site might update a persistent shopping cart to show the number of items in it. These changes are often subtle on a computer screen; with a paper prototype, it's much more obvious because the Computer leans over the table and sticks something on the prototype. Thus, with any sort of small or subtle change, a paper prototype can only tell you if users *understand* the change, not whether they would *notice* it—you can't draw any conclusions about the latter.

Scrolling on Web Pages

On some Web pages, users don't scroll down, and thus they miss a link or content "below the fold" that's relevant to what they're trying to do. You might try simulating the fold literally, by folding the Web page where it would fall at a given resolution. If the user says he or she would scroll down, unfold the paper so that the rest of the page can be seen. Another method is to use a cutout as shown in Chapter 4. These methods may help you determine whether you've succeeded in placing the most important information above the fold, but they're rough measures at best.

Because a folded piece of paper is more salient than the scroll bar on a Web page, if users "scroll" on the paper prototype it's risky to assume that they also will on the real site. On the other hand, if users *don't* scroll with the prototype and miss something important that's below the fold, then this perhaps does indicate a problem with the page layout that you'll want to do something about.

Long Documents and Lists

Interactions with a long document or list are difficult to simulate adequately with a paper prototype. In one usability study, I watched users navigate through a series of online documents containing industrial regulations. As long as users clicked the down arrow they were fine, but the minute they moved the scroll bar directly they'd end up in another chapter—given that there were hundreds or even thousands of pages, even a small movement of the scroll bar landed the users several dozen pages away. This problem was painfully clear from watching users use the real interface (and confirmed the team's decision to limit scrolling to just the current document rather than the whole series), but we wouldn't have seen it with a paper prototype.

Similarly, I've seen users have difficulty selecting the item they wanted from a long list. When a user scrolls through a long list by clicking repeatedly, their attention is on the items in the list, not the down (or up) arrow they're clicking. If the mouse cursor drifts off the scroll bar control, the user's next click will land either within the list (selecting an item the user didn't want) or outside of it (causing the list to disappear). Alternatively, the user might try to use the keyboard—it's common for users to type the word they're looking for, such as "fr" to find France in a list of countries. Some interface elements (such as the URL field in Internet Explorer) support this behavior, but drop-down lists don't. Many users aren't aware of the difference and are surprised when typing the R whisks them off to Romania.

The "False Top" Problem

A false top is a Web page problem that's a combination of visual design and interaction. On some Web sites, once the users scroll down, they are inhibited from scrolling back up all the way to the top because at some point the page appears as though it's already there. (The scroll bar elevator does not seem to be a sufficient clue that more of the page is out of sight above.) I call this a *false top* (see Figure

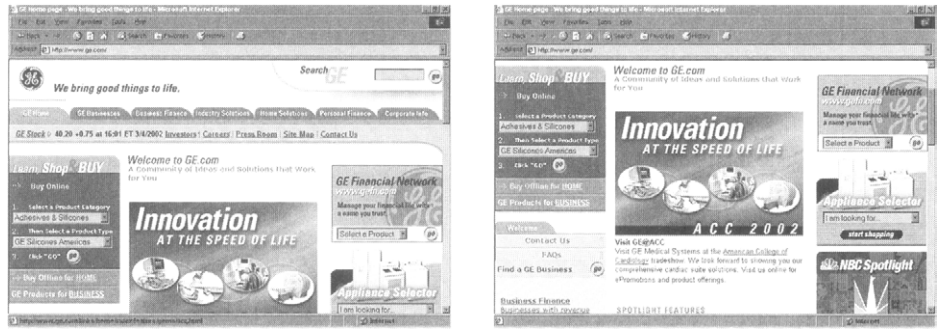

Figure 12.3 Example of a page with a false top—the left image shows the true top of the page, and the right image shows how the page also appears to be at the top when the user has scrolled down, part way up, and then stopped. You probably won't find a false top problem with a paper prototype.

12.3). A false top can cause usability problems (although usually minor ones) if the user has trouble finding top-of-page navigation that's needed for the task.

Keystroke or Mouse Errors

Although the term *user error* is no longer considered politically correct, people do hit the wrong thing sometimes even when they know perfectly well how to use the interface. Premature form submission is one example of a keystroke error—the user accidentally submits a form by hitting Enter instead of Tab. Speaking of tabs, sometimes the tab order may not match the order in which users expect to fill out fields, or auto-tab may cause data to be entered into the wrong field. Because in a paper prototype the user isn't using a keyboard, chances of finding these problems are slim.

For an example of a mouse error, look at the options in Figure 12.4 that McAfee Personal Firewall gives me when it detects an unauthorized attempt to access my computer. The "Trust this address" option, which I never use, is sandwiched right between the two options I do use, and sometimes I click it by mistake. (This type of error is sometimes called a 1-off problem.)

Size of Controls

In a paper prototype, users simply touch what they want to click and we make the assumption that they did it correctly. But in real life this assumption can be wrong, and a paper prototype won't tell you how large a button or other clickable area

Figure 12.4 As I have learned to my chagrin, it's not hard to click "Trust this address" by mistake when I'm going for "Ban this address" instead. This is another problem I wouldn't have found with a paper prototype.

needs to be. In one usability test of a Web site, I watched a user who had great difficulty clicking on anything small (for example, the state of New Hampshire in a map of the United States). I noticed that she seemed to be clicking too high, and I finally figured out that she was centering the arrow cursor over the target she wanted to click rather than putting the *tip* of the cursor in the middle of the target. When the target was small, the hot spot ended up above the thing she was aiming at.

A computer screen faces the size issue in only two dimensions—with a handheld device or manually operated equipment, the problems become more complex. Even a three-dimensional prototype may not provide enough realism to adequately investigate issues pertaining to control size, placement, operation, tactile feedback, and repetitive stress.

Mouse versus Keyboard Preference

With a paper prototype it's hard to answer the question "Do users prefer the mouse or the keyboard?" or to spot places where they're slowed down by having to switch back and forth. In watching users complete order forms on real e-commerce sites, I noticed that very few users navigated a list by using the arrow keys—almost everyone used the mouse. Thus, when a drop-down list was placed in a

form containing edit fields, the keyboard-mouse-keyboard transition slowed users down. Would I have seen this with paper? Nope, because the users' hands take the place of both mouse and keyboard.

Rollover and Cascading Menus

A *rollover menu* is one that appears when the user positions the cursor over a top-level menu option. Rollover menus can give users trouble by popping up unexpectedly (sometimes obscuring the page underneath), and some users don't understand how to use them. A *cascading menu* refers to multiple menus linked together in hierarchical fashion, like the Windows Start menu. They're difficult to use because users must move the mouse at 90-degree angles rather than taking the hypotenuse to the desired submenu choice. Because the behavior of rollover and cascading menus depends on the positioning of the mouse cursor and the timing of its movements, they're too subtle to test with a paper prototype.

Download Time or Other Response Time

Because a person simulates the behavior of the computer, the "response time" is artificial. And it can be off in either direction; on one hand, a person shuffling paper might be slower than a computer, but on the other hand a paper prototype eliminates delays due to processing, servers, and so on. Because the Computer's speed does not accurately represent the machine's, measurements of task times, error rates, and the like can be skewed by testing with a paper prototype.

Response time usually depends on technical factors, so with a paper prototype you probably can't get data about whether the interface will respond quickly enough to satisfy users. (And sometimes it's hard to answer these questions even with the real interface unless you watch users work in their own environment, doing something they care about.)

Finding Problems through Inspection

Some types of usability problems can be found simply by looking for them—the false top, cascading menu, and keyboard-mouse issues are ones I can often predict just by looking at a screen. However, there's something of a Catch-22 here: To find a problem by inspection in a paper prototype, you have to know to look for it

and that knowledge only comes from having seen examples of it in real interfaces and/or interface guidelines. To the extent that you're able to educate yourself about common usability issues, you may be able to use that expertise to partially compensate for some of paper prototyping's deficiencies. But don't expect "armchair usability" to replace the real thing because there are many, many important problems that you'll never find if you look only for the problems you know about. (If you want examples, reread Chapter 2.) Usability testing is useful way to check your blind spots.

What Usability Testing Won't Find (Real-Life Situations)

Usability testing, whether of paper prototypes or real products, has blind spots of its own. So far, this chapter has mostly been about the method of paper prototyping and what problems it will and won't find. A similar analysis could be done for other methods of prototyping, but we're still talking about how well the methods do at eliciting information *in the context of a usability test.* Although a full discussion of this topic is beyond the scope of this book, I'd like to touch on some of the things that are difficult to discover in a usability test.

- **Long-term use.** Once people learn to use it, how well can they remember how to do things? How efficiently can they get things done? How easily can they focus on the task and forget about the interaction with the system? What can be streamlined? What "little" things become huge annoyances when users encounter them every day?
- **Integration and compatibility.** How does this interface fit with others that the target market is using? Does it "play nice" with the operating system and other hardware and applications? Does it have inconsistencies with those things that bother users?
- **Real-life context and needs.** Do users' goals and tasks really correspond to those things we asked them to do in usability testing? How often do users get interrupted in the middle of using the interface or have to leave it to go look something up? How much work do they lose as a result? Do they use all the functionality? Is the product really making users' lives better in the way we envisioned? Especially if the users' goals are complex or take a long time to accomplish (for example, using chemical modeling software to create a new drug), you aren't going to get the whole picture in a 2-hour usability test.

Showstoppers can lurk beneath the surface of any of these questions, but you aren't going to find them in a lab. When I claim that paper prototyping can find many important issues, sometimes people respond "But not all of them." And that's true. But it's true of any method. Although paper prototyping and usability testing are marvelously useful techniques, they are a complement, never a substitute, for other methods of understanding users and their needs. (See the References section for more information.)

Summary

This chapter has outlined the reasons why paper prototypes are better at finding some types of problems than others. Every prototyping technique has strengths and weaknesses, and every interface has a unique set of questions that the product team would like to investigate. To summarize the strategy presented in this chapter:

1. List the kinds of questions you have about your interface in terms of the four dimensions (breadth, depth, look, and interaction).
2. Determine which dimensions are needed to address your most important questions. (Refer to Table 12.2.)
3. Choose a prototyping method that has strengths in those dimensions. (Refer to Table 12.1.)
4. Accept that one method won't find everything.
5. Reality-check your findings whenever you come into contact with users.

Think of a paper prototype as a coarse screen that you use to sift out problems. Now that you've read this chapter, you should have an idea of which problems your paper prototype will catch and which might "fall through" to be found later or with other methods.