What do Prototypes Prototype?

Stephanie Houde and Charles Hill
Apple Computer, Inc.
Cupertino, CA, USA
s.houde@ix.netcom.com, hillc@ix.netcom.com

1. INTRODUCTION

Prototypes are widely recognized to be a core means of exploring and expressing designs for interactive computer artifacts. It is common practice to build prototypes in order to represent different states of an evolving design, and to explore options. However, since interactive systems are complex, it may be difficult or impossible to create prototypes of a whole design in the formative stages of a project. Choosing the right kind of more focused prototype to build is an art in itself, and communicating its limited purposes to its various audiences is a critical aspect of its use.

The ways that we talk, and even think about prototypes, can get in the way of their effective use. Current terminology for describing prototypes centers on attributes of prototypes themselves, such as what tool was used to create them, and how refined-looking or -behaving they are. Such terms can be distracting. Tools can be used in many different ways, and detail is not a sure indicator of completeness.

We propose a change in the language used to talk about prototypes, to focus more attention on fundamental questions about the interactive system being designed: What role will the artifact play in a user's life? How should it look and feel? How should it be implemented? The goal of this chapter is to establish a model that describes any prototype in terms of the artifact being designed, rather than the prototype's incidental attributes. By focusing on the purpose of the prototype—that is, on *what it prototypes*—we can make better decisions about the kinds of prototypes to build. With a clear purpose for each prototype, we can better use prototypes to think and communicate about design.

In the first section we describe some current difficulties in communicating about prototypes: the complexity of interactive systems; issues of multidisciplinary teamwork; and the audiences of prototypes. Next, we introduce the model and illustrate it with some initial examples of prototypes from real projects. In the following section we present several more examples to illustrate some further issues. We conclude the chapter with a summary of the main implications of the model for prototyping practice.

2. THE PROBLEM WITH PROTOTYPES

Interactive computer systems are complex. Any artifact can have a rich variety of software, hardware, auditory, visual, and interactive features. For example, a personal digital assistant such as the Apple Newton has an operating system, a hard case with various ports, a graphical user interface and audio feedback. Users experience the combined effect of such interrelated features; and the task of designing—and prototyping—the user experience is therefore complex. Every aspect of the system must be designed (or inherited from a previous system), and many features need to be evaluated in combination with others.

Prototypes provide the means for examining design problems and evaluating solutions. Selecting the focus of a prototype is the art of identifying the most important open design questions. If the artifact is to provide new functionality for users—and thus play a new *role* in their lives—the most important questions may concern exactly what that role should be and what features are needed to support it. If the role is well understood, but the goal

This article is published, in a different format, as Houde, S., and Hill, C., *What Do Prototypes Prototype?*, in Handbook of Human-Computer Interaction (2nd Ed.), M. Helander, T. Landauer, and P. Prabhu (eds.): Elsevier Science B. V: Amsterdam, 1997.

of the artifact is to present its functionality in a novel way, then prototyping must focus on how the artifact will *look and feel*. If the artifact's functionality is to be based on a new technique, questions of how to *implement* the design may be the focus of prototyping efforts.

Once a prototype has been created, there are several distinct audiences that designers discuss prototypes with. They are: the intended *users* of the artifact being designed; their *design teams*; and the supporting *organizations* that they work within (Erickson, 1995). Designers evaluate their options with their own team by critiquing prototypes of alternate design directions. They show prototypes to users to get feedback on evolving designs. They show prototypes to their supporting organizations (such as project managers, business clients, or professors) to indicate progress and direction.

It is difficult for designers to communicate clearly about prototypes to such a broad audience. It is challenging to build prototypes which produce feedback from users on the most important design questions. Even communication among designers requires effort due to differing perspectives in a multidisciplinary design team. Limited understanding of design practice on the part of supporting organizations makes it hard for designers to explain their prototypes to them. Finally, prototypes are not self-explanatory: looks can be deceiving. Clarifying what aspects of a prototype correspond to the eventual artifact—and what don't—is a key part of successful prototyping.

2.1 What is a prototype?

Designing interactive systems demands collaboration between designers of many different disciplines (Kim, 1990). For example, a project might require the skills of a programmer, an interaction designer, an industrial designer, and a project manager. Even the term "prototype" is likely to be ambiguous on such a team. Everyone has a different expectation of what a prototype is. Industrial designers call a molded foam model a prototype. Interaction designers refer to a simulation of on-screen appearance and behavior as a prototype. Programmers call a test program a prototype. A user studies expert may call a storyboard, which shows a scenario of something being used, a prototype.

The organization supporting a design project may have an overly narrow expectation of what a prototype is. Shrage (1996) has shown that organizations develop their own "prototyping cultures" which may cause them to consider only certain kinds of prototypes to be valid. In some organizations, only prototypes which act as proof that an artifact can be produced are respected. In others, only highly detailed representations of look and feel are well understood.

Is a brick a prototype? The answer depends on how it is used. If it is used to represent the weight and scale of some future artifact, then it certainly is: it prototypes the weight and scale of the artifact. This example shows that prototypes are not necessarily self-explanatory. What is significant is not what media or tools were are used to create them, but *how they are used by a designer* to explore or demonstrate some aspect of the future artifact.

2.2 Current terminology

Current ways of talking about prototypes tend to focus on attributes of the prototype itself, such as which tool was used to create it (as in "C", "Director™", and "paper" prototypes); and on how finished-looking or -behaving a prototype is (as in "high-fidelity" and "low-fidelity" prototypes). Such characterizations can be misleading because the capabilities and possible uses of tools are often misunderstood and the significance of the level of finish is often unclear, particularly to non-designers.

Tools can be used in many different ways. Sometimes tools which have high-level scripting languages (like HyperCard™), rather than full programming languages (like C), are thought to be unsuitable for producing user-testable prototypes. However, Ehn and Kyng (1991) have shown that even prototypes made of cardboard are very useful for user testing. In the authors' experience, no one tool supports iterative design work in all of the important areas of investigation. To design well, designers must be willing to use different tools for different prototyping tasks; and to team up with other people with complementary skills.

Finished-looking (or -behaving) prototypes are often thought to indicate that the design they represent is near completion. Although this may sometimes be the case, a finished-looking prototype might be made early in the design process (e.g., a 3D concept model for use in market research), and a rough one might be made later on (e.g., to emphasize overall structure rather than visual details in a user test). Two related terms are used in this context: "resolution" and "fidelity". We interpret resolution to mean "amount of detail", and fidelity to mean "closeness to the eventual design". It is important to recognize that the degree of visual and behavioral refinement of a prototype does not necessarily correspond to the solidity of the design, or to a particular stage in the process.

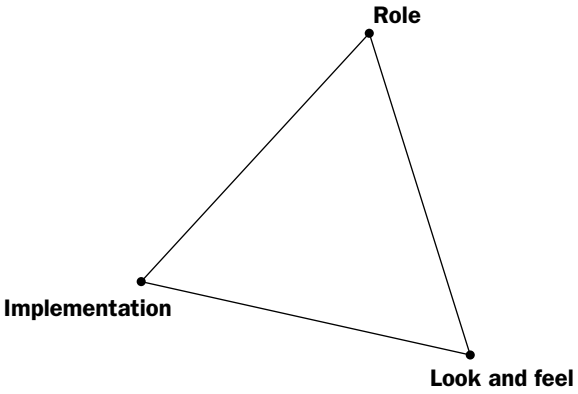


Figure 1. A model of what prototypes prototype.

3. A MODEL OF WHAT PROTOTYPES PROTOTYPE

3.1 Definitions

Before proceeding, we define some important terms. We define *artifact* as the interactive system being designed. An artifact may be a commercially released product or any end-result of a design activity such as a concept system developed for research purposes. We define *prototype* as any representation of a design idea, regardless of medium. This includes a preexisting object when used to answer a design question. We define *designer* as anyone who creates a prototype in order to design, regardless of job title.

3.2 The model

The model shown in Figure 1 represents a three-dimensional space which corresponds to important aspects of the design of an interactive artifact. We define the dimensions of the model as *role*; *look and feel*; and *implementation*. Each dimension corresponds to a class of questions which are salient to the design of any interactive system. "Role" refers to questions about the function that an artifact serves in a user's life—the way in which it is useful to them. "Look and feel" denotes questions about the concrete sensory experience of using an artifact—what the user looks at, feels and hears while using it. "Implementation" refers to questions about the techniques and components through which an artifact performs its function—the "nuts and bolts" of how it actually works. The triangle is drawn askew to emphasize that no one dimension is inherently more important than any other.

Goal of the model

Given a design problem (of any scope or size), designers can use the model to separate design issues into three classes of questions which frequently demand different approaches to prototyping. Implementation usually requires a working system to be built; look and feel requires the concrete user experience to be simulated or actually created; role requires the context of the artifact's use to be established. Being explicit about what design questions must be answered is therefore an essential aid to deciding what kind of prototype to build. The model helps visualize the focus of exploration.

Markers

A prototype may explore questions or design options in one, two or all three dimensions of the model. In this chapter, several prototypes from real design projects are presented as examples. Their relationship to the model is represented by a marker on the triangle. This is a simple way to put the purpose of any prototype in context for the designer and their audiences. It gives a global sense of what the prototype is intended to explore; and equally important, what it does not explore.

It may be noted that the triangle is a relative and subjective representation. A location toward one corner of the triangle implies simply that *in the designer's own judgment*, more attention is given to the class of questions represented by that corner than to the other two.

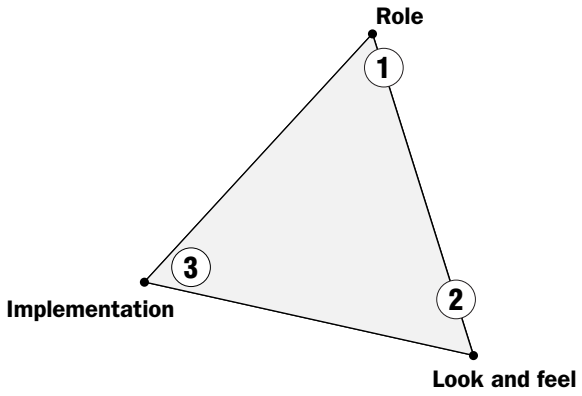
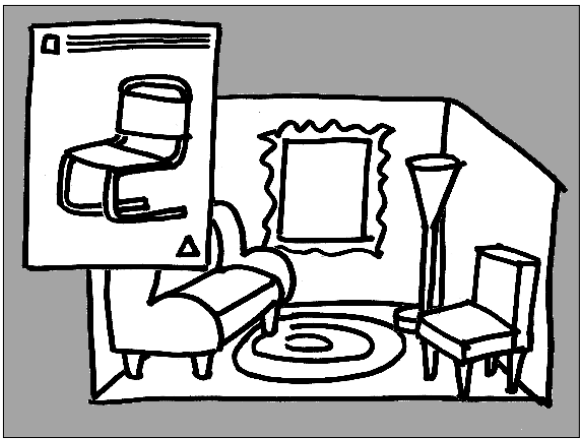


Figure 2. Relationship of three prototypes (Examples 1-3) to the model.



Example 1. Role prototype for 3D space-planning application [E1 Houde 1990].

3.3 Three prototypes of one system

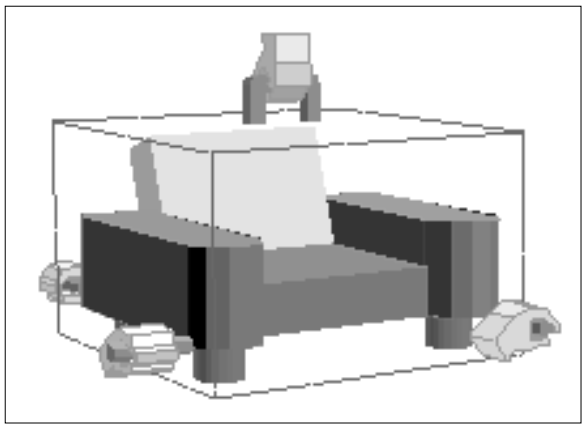
The model is best explained further through an example from a real project. The three prototypes shown in Examples 1-3 were created during the early stages of development of a 3D space-planning application (Houde, 1992).

The goal of the project was to design an example of a 3D application which would be accessible to a broad range of nontechnical users. As such it was designed to work on a personal computer with an ordinary mouse. Many prototypes were created by different members of the multi-disciplinary design team during the project.

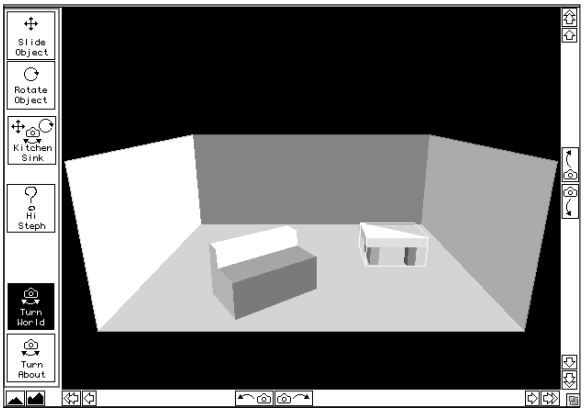
The prototype shown in Example 1 was built to show how a user might select furniture from an online catalog and try it out in an approximation of their own room. It is an interactive slide show which the designer operated by clicking on key areas of the rough user interface. The idea that virtual space-planning would be a helpful task for nontechnical users came from user studies. The purpose of the prototype was to quickly convey the proposed role of the artifact to the design team and members of the supporting organization.

Since the purpose of the prototype was primarily to explore and visualize an example of the role of the future artifact, its marker appears very near the role corner of the model in Figure 2. It is placed a little toward the look and feel corner because it also explored user interface elements in a very initial form.

One of challenges of the project was to define an easy-to-use direct manipulation user interface for moving 3D objects with an ordinary 2D mouse cursor. User testing with a foam-core model showed that the most important manipulations of a space-planning task were sliding, lifting, and turning furniture objects. Example 2 shows a picture of a prototype which was made to test a user interface featuring this constrained set of manipulations. Clicking once on the chair object caused its bounding box to appear. This “handle box” offered hand-shaped controls for lifting and turning the box and chair (as if the chair was frozen inside the box). Clicking and dragging anywhere on the box allowed the unit to slide on a 3D floor. The prototype was built using Macromedia Director (a high level animation and scripting tool.) It was made to work only with the chair data shown: a set of images predrawn for many angles of rotation.



Example 2. Look-and-feel prototype for 3D space-planning application [E2 Houde 1990].



Example 3. Implementation prototype for 3D space-planning application [E3 Chen 1990].

The purpose of Example 2 was to get feedback from users as quickly as possible as to whether the look and feel of the handle box user interface was promising. Users of the prototype were given tasks which encouraged them to move the chair around a virtual room. Some exploration of role was supported by the fact that the object manipulated was a chair, and space-planning tasks were given during the test. Although the prototype was interactive, the programming that made it so did not seriously explore how a final artifact with this interface might be implemented. It was only done in service of the look and feel test. Since the designer primarily explored the look and feel of the user interface, this prototype's marker is placed very near the look and feel corner of the model in Figure 2.

A technical challenge of the project was figuring out how to render 3D graphics quickly enough on equipment that end-users might have. At the time, it was not clear how much real time 3D interaction could be achieved on the Apple Macintosh™ IIfx computer—the fastest Macintosh then available. Example 3 shows a prototype which was built primarily to explore rendering capability and performance. This was a working prototype in which multiple 3D objects could be manipulated as in Example 2, and the view of the room could be changed to any perspective. Example 3 was made in a programming environment that best supported the display of true 3D perspectives during manipulation. It was used by the design team to determine what complexity of 3D scenes was reasonable to design for. The user interface elements shown on the left side of the screen were made by the programmer to give himself controls for demonstrating the system: they were not made to explore the look and feel of the future artifact. Thus the primary purpose of the prototype was to explore how the artifact might be implemented. The marker for this example is placed near the implementation corner (Figure 2).

One might assume that the role prototype (Example 1) was developed first, then the look and feel prototype (Example 2), and finally the implementation prototype (Example 3): that is, in order of increasing detail and production difficulty. In fact, these three prototypes were developed almost in parallel. They were built by different design team members during the early stages of the project. No single prototype could have represented the design of the future artifact at that time. The evolving design was too fuzzy—existing mainly as a shared concept in the minds of the designers. There were also too many open and interdependent questions in every design dimension: role, look and feel, implementation.

Making separate prototypes enabled specific design questions to be addressed with as much clarity as possible. The solutions found became inputs to an integrated design. Answers to the rendering capability questions addressed by Example 3 informed the design of the role that the artifact could play (guiding how many furniture objects of what complexity could be shown). It also provided guiding constraints for the direct manipulation user interface (determining how much detail the handle forms could have). Similarly, issues of role addressed by Example 1 informed the implementation problem by constraining it: only a constrained set of manipulations was needed for a space-planning application. It also simplified the direct manipulation user interface by limiting the necessary actions and therefore controls which needed to be provided.

It was more efficient to wait on the results of independent investigations in the key areas of role, look and feel and implementation than to try to build a monolithic prototype that integrated all features from the start. After sufficient investigation in separate prototypes, the prototype in Example 3 began

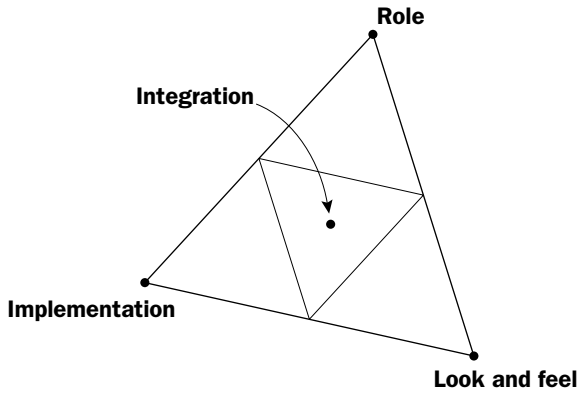


Figure 3. Four principal categories of prototypes on the model.

to evolve into an integrated prototype which could be described by a position at the center of our model. A version of the user interface developed in Example 2 was implemented in the prototype in Example 3. Results of other prototypes were also integrated. This enabled a more complete user test of features and user interface to take place.

This set of three prototypes from the same project shows how a design problem can be simultaneously approached from multiple points of view. Design questions of role, look and feel, and implementation were explored concurrently by the team with the three separate prototypes. The purpose of the model is to make it easier to develop and subsequently communicate about this kind prototyping strategy.

4. FURTHER EXAMPLES

In this section we present twelve more examples of prototypes taken from real projects, and discuss them in terms of the model. Examples are divided into four categories which correspond to the four main regions of the model, as indicated in Figure 3. The first three categories correspond to prototypes with a strong bias toward one of the three corners: *role*, *look and feel*, and *implementation* prototypes, respectively. *Integration* prototypes occupy the middle of the model: they explore a balance of questions in all three dimensions.

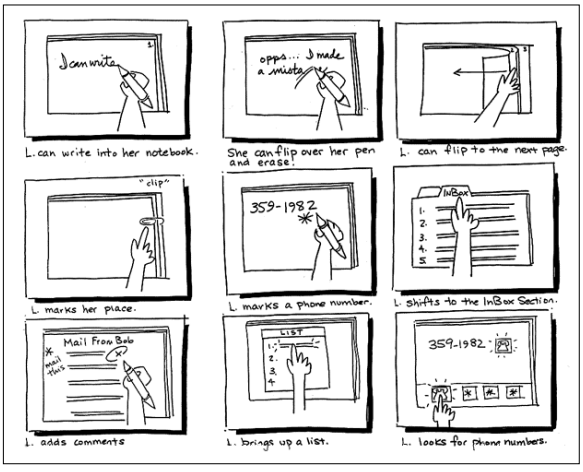
4.1 Role prototypes

Role prototypes are those which are built primarily to investigate questions of what an artifact could do for a user. They describe the functionality that a user might benefit from, with little attention to how the artifact would look and feel, or how it could be made to actually work. Designers find such prototypes useful to show their design teams what the target role of the artifact might be; to communicate that role to their supporting organization; and to evaluate the role in user studies.

A portable notebook computer

The paper storyboard shown in Example 4 was an early prototype of a portable notebook computer for students which would accept both pen and finger input. The scenario shows a student making notes, annotating a paper, and marking pages for later review in a computer notebook. The designer presented the storyboard to her design team to focus discussion on the issues of what functionality the notebook should provide and how it might be controlled through pen and finger interaction. In terms of the model, this prototype primarily explored the role of the notebook by presenting a rough task scenario for it. A secondary consideration was a rough approximation of the user interface. Its marker, shown in Figure 4, is therefore positioned near the role corner of the model and a little toward look and feel.

Storyboards like this one are considered to be effective design tools by many designers because they help focus design discussion on the role of an artifact very early on. However, giving them status as prototypes is not common because the medium is paper and thus seems very far from the medium of



Example 4. Storyboard for a portable notebook computer [E4 Vertelney 1990].

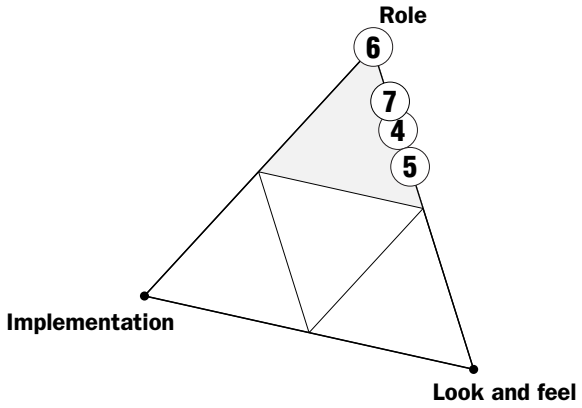
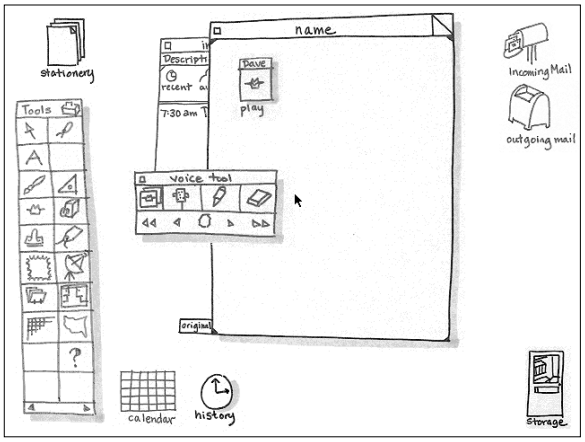


Figure 4. Relationship of role prototypes (Examples 4-7) to the model.

an interactive computer system. We consider this storyboard to be a prototype because it makes a concrete representation of a design idea and serves the purpose of asking and answering design questions. Of course, if the designer needed to evaluate a user's reaction to seeing the notebook or to using the pen-and-finger interaction, it would be necessary to build a prototype which supported direct interaction. However, it might be wasteful to do so before considering design options in the faster, lighter-weight medium of pencil and paper.

An operating system user interface

Example 5 shows a screen view of a prototype that was used to explore the design of a new operating system. The prototype was an interactive story: it could only be executed through a single, ordered, sequence of interactions. Clicking with a cursor on the mailbox picture opened a mail window; then



Example 5. Interactive story for an operating system interface [E5 Vertelney and Wong 1990].

clicking on the voice tool brought up a picture of some sound tools; and so on. To demonstrate the prototype, the designer sat in front of a computer and play-acted the role of a user opening her mail, replying to it, and so forth. The prototype was used in design team discussions and also demonstrated to project managers to explain the current design direction. According to the model, this prototype primarily explored the role that certain features of the operating system could play in a user's daily tasks. It was also used to outline very roughly how its features would be portrayed and how a user would interact with it. As in the previous example, the system's implementation was not explored. Its marker is shown in Figure 4.

To make the prototype, user interface elements were hand-drawn and scanned in. Transitions between steps in the scenario were made interactive in Macromedia Director. This kind of portrayal of on-screen interface elements as rough and hand-drawn was used in order to focus design discussion on the overall features of a design rather than on specific details of look and feel or implementation (Wong, 1992). Ironically, while the design team understood the meaning of the hand-drawn graphics, other members of the organization became enamored with the sketchy style to the extent that they considered using it in the final artifact. This result was entirely at odds with the original reasons for making a rough-looking prototype. This example shows how the effectiveness of some kinds of prototypes may be limited to a specific kind of audience.

The Knowledge Navigator

Example 6 shows a scene from Apple Computer's Knowledge Navigator™ video. The video tape tells a day-in-the-life story of a professor using a futuristic notebook computer (Dubberly and Mitch, 1987). An intelligent agent named "Phil" acts as his virtual personal assistant, finding information related to a lecture, reminding him of his mother's birthday, and connecting him with other professors via video-link. The professor interacts with Phil by talking, and Phil apparently recognizes everything said as well as a human assistant would.

Based on the model, the Knowledge Navigator is identified primarily as a prototype which describes

Example 6. Knowledge Navigator™ vision video for a future notebook computer [E6 Dubberly and Mitch '87].

the role that the notebook would play in such a user's life. The story is told in great detail, and it is clear that many decisions were made about what to emphasize in the role. The video also shows specific details of appearance, interaction, and performance. However, they were not intended by the designers to be prototypes of look and feel. They were merely placeholders for the actual design work which would be necessary to make the product really work. Thus its marker goes directly on the role corner (Figure 4).

Thanks to the video's special effects, the scenario of the professor interacting with the notebook and his assistant looks like a demonstration of a real product. Why did Apple make a highly produced prototype when the previous examples show that a rapid paper storyboard or a sketchy interactive prototype were sufficient for designing a role and telling a usage story? The answer lies in the kind of audience. The tape was shown publicly and to Apple employees as a vision of the future of computing. Thus the audience of the Knowledge Navigator was very broad—including almost anyone in the world. Each of the two previous role design prototypes was shown to an audience which was well informed about the design project. A rough hand-drawn prototype would not have made the idea seem real to the broad audience the video addressed: high resolution was necessary to help people concretely visualize the design. Again, while team members learn to interpret abstract kinds of prototypes accurately, less expert audiences cannot normally be expected to understand such approximate representations.

The Integrated Communicator

Example 7 shows an appearance model of an Integrated Communicator created for customer research into alternate presentations of new technology (ID Magazine 1995). It was one of three presentations of possible mechanical configurations and interaction designs, each built to the same high finish and accompanied by a video describing on-screen interactions. In the study, the value of each presentation was evaluated relative to the others, as perceived by study subjects during one-on-one interviews. The prototype was used to help subjects imagine such a product in the store and in their homes or offices, and thus to evaluate whether they would purchase such a product, how much they would expect it to cost, what features they would expect, etc.

The prototype primarily addresses the role of the product, by presenting carefully designed cues which imply a telephone-like role and look-and-feel. Figure 4 shows its marker near the role corner of the model. As with the Knowledge Navigator, the very high-resolution look and feel was a means of making the design as concrete as possible to a broad audience. In this case however it also enabled a basic interaction design strategy to be worked out and demonstrated. The prototype did not address implementation.

Example 7. Appearance model for the Integrated Communicator [E7 Udagawa 1995].

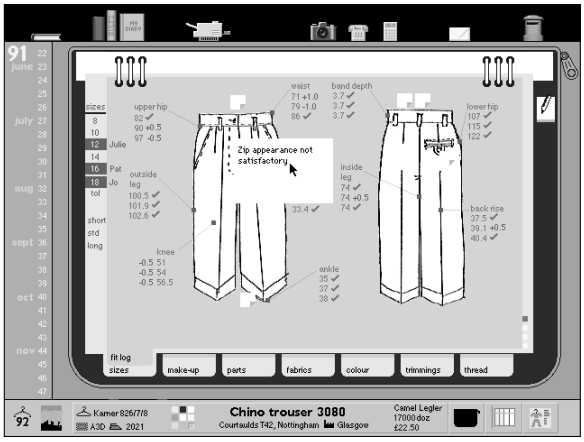
The key feature of this kind of prototype is that it is a concrete and direct representation, as visually finished as actual consumer products. These attributes encourage an uncoached person to directly relate the design to their own environment, and to the products they own or see in stores. High quality appearance models are costly to build. There are two common reasons for investing in one: to get a visceral response by making the design seem "real" to any audience (design team, organization, and potential users); and to verify the intended look and feel of the artifact before committing to production tooling. An interesting side-effect of this prototype was that its directness made it a powerful prop for promoting the project within the organization.

4.2 Look and Feel prototypes

Look and feel prototypes are built primarily to explore and demonstrate options for the concrete experience of an artifact. They simulate what it would be like to look at and interact with, without necessarily investigating the role it would play in the user's life or how it would be made to work. Designers make such prototypes to visualize different look and feel possibilities for themselves and their design teams. They ask users to interact with them to see how the look and feel could be improved. They also use them to give members of their supporting organization a concrete sense of what the future artifact will be like.

A fashion design workspace

The prototype shown in Example 8 was developed to support research into collaboration tools for fashion designers (Hill et al., 1993; Scaife et al, 1994). A twenty-minute animation, it presented the concept design for a system for monitoring garment design work. It illustrated in considerable detail the translation of a proven paper-based procedure into a computer-based system with a visually rich, direct manipulation, user interface. The prototype's main purposes were to confirm to the design team that an engaging and effective look and feel could be designed for this application, and to convince managers of the possibilities of the project. It was presented to users purely for informal discussion.



Example 8. Animation of the look and feel of a fashion design workspace [E8 Hill 1992].

This is an example of a look and feel prototype. The virtue of the prototype was that it enabled a novel user interface design to be developed without having first to implement complex underlying technologies. While the role was inherited from existing fashion design practice, the prototype also demonstrated new options offered by the new computer-based approach. Thus, Figure 5 shows its marker in the look and feel area of the model.

One issue with prototypes like this one is that inexperienced audiences tend to believe them to be more functional than they are just by virtue of being shown on a computer screen. When this prototype was shown, the designers found they needed to take great care to explain that the design was not implemented.

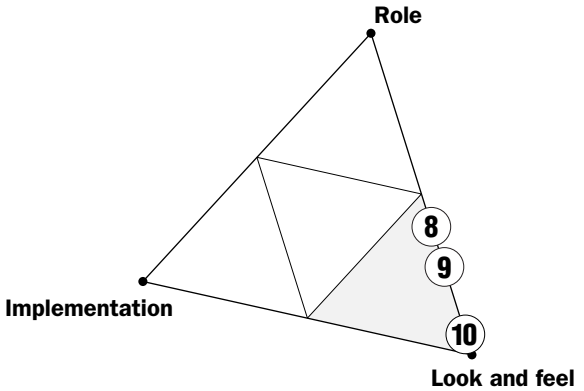
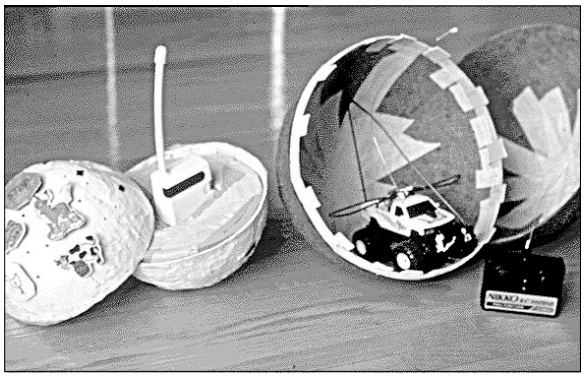


Figure 5. Relationship of the look and feel prototypes (Examples 8-10) to the model.

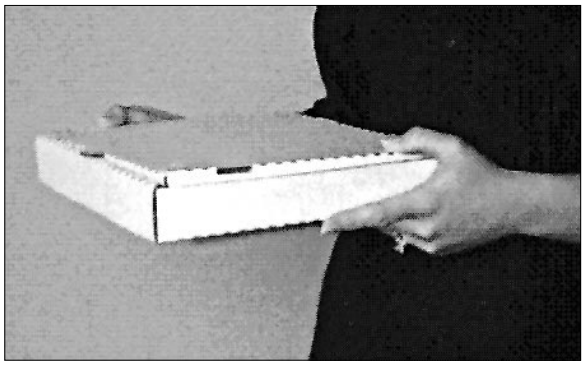
A learning toy

The "GloBall" project was a concept for a children's toy: a ball that would interact with children who played with it. Two prototypes from the project are shown, disassembled, in Example 9. The design team wanted the ball to speak back to kids when they spoke to it, and to roll towards or away from them in reaction to their movements. The two prototypes were built to simulate these functions separately. The ball on the left had a walkie-talkie which was concealed in use. A hidden operator spoke into a linked walkie-talkie to simulate the ball's speech while a young child played with it. Similarly, the ball on the right had a radio-controlled car which was concealed in use. A hidden operator remotely controlled the car, thus causing the ball to roll around in response to the child's actions.

As indicated by the marker in Figure 5, both prototypes were used to explore the toy's look and feel from a child's viewpoint, and to a lesser extent to evaluate the role that the toy would play. Neither seriously addressed implementation. The designers of these very efficient prototypes wanted to know how a child would respond to a toy that appeared to speak and move of its own free will. They managed to convincingly simulate novel and difficult-to-implement technologies such as speech and automotion, for minimal cost and using readily available components. By using a "man behind the curtain" (or "Wizard of Oz") technique, the designers were able to present the prototypes directly to children and to directly evaluate their effect.



Example 9. Look and feel simulation prototypes for a child's toy [E9 Bellman et al, 1993].



Example 10. Pizza-box prototype of an architect's computer [E10 Apple Design Project, 1992].

An architect's computer

This example concerned the design of a portable computer for architects who need to gather a lot of information during visits to building sites. One of the first questions the designers explored was what form would be appropriate for their users. Without much ado they weighted the pizza box shown in Example 10 to the expected weight of the computer, and gave it to an architect to carry on a site visit. They watched how he carried the box, what else he carried with him, and what tasks he needed to do during the visit. They saw that the rectilinear form and weight were too awkward, given the other materials he carried with him, and this simple insight led them to consider of a softer form. As shown by its marker, this is an example of a rough look and feel prototype (Figure 5). Role was also explored in a minor way by seeing the context that the artifact would be used in.

The pizza box was a very efficient prototype. Spending virtually no time building it or considering options, the students got useful feedback on a basic design question—what physical form would be best for the user. From what they learned in their simple field test, they knew immediately that they should try to think beyond standard rectilinear notebook computer forms. They began to consider many different options including designing the computer to feel more like a soft shoulder bag.

4.3 Implementation Prototypes

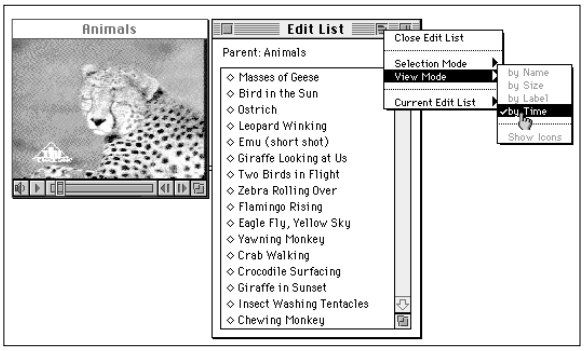
Some prototypes are built primarily to answer technical questions about how a future artifact might actually be made to work. They are used to discover methods by which adequate specifications for

the final artifact can be achieved—without having to define its look and feel or the role it will play for a user. (Some specifications may be unstated, and may include externally imposed constraints, such as the need to reuse existing components or production machinery.) Designers make implementation prototypes as experiments for themselves and the design team, to demonstrate to their organization the technical feasibility of the artifact, and to get feedback from users on performance issues.

A digital movie editor

Some years ago it was not clear how much interactivity could be added to digital movies playing on personal computers. Example 11 shows a picture of a prototype that was built to investigate solutions to this technical challenge. It was an application, written in the C programming language to run on an Apple Macintosh computer. It offered a variety of movie data-processing functionality such as controlling various attributes of movie play. The main goal of the prototype was to allow marking of points in a movie to which scripts (which added interactivity) would be attached. As indicated by the marker in Figure 6, this was primarily a carefully planned implementation prototype. Many options were evaluated about the best way to implement its functions. The role that the functions would play was less well defined. The visible look and feel of the prototype was largely incidental: it was created by the designer almost purely to demonstrate the available functionality, and was not intended to be used by others.

This prototype received varying responses when demonstrated to a group of designers who were not



Example 11. Working prototype of a digital movie editor [E11 Degen, 1994].

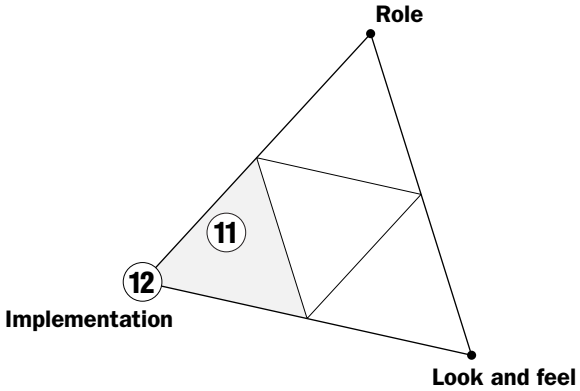


Figure 6. Relationship of implementation prototypes (Examples 11 and 12) to the model.

members of the movie editor design team. When the audience understood that an implementation design was being demonstrated, discussion was focused productively. At other times it became focused on problems with the user interface, such as the multiple cascading menus, which were hard to control and visually confusing. In these cases, discussion was less productive: the incidental user interface got in the way of the intentional implementation.

The project leader shared some reflections after this somewhat frustrating experience. He said that part of his goal in pursuing a working prototype alone was to move the project through an organization that respected this kind of prototype more than "smoke and mirrors" prototypes—ones which only simulate functionality. He added that one problem might have been that the user interface was neither good enough nor bad enough to avoid misunderstandings. The edit list, which allowed points to be marked in movies, was a viable look and feel design; while the cascading menus were not. For the audience that the prototype was shown to, it might have been more effective to stress the fact that look and feel were not the focus of the prototype; and perhaps, time permitting, to have complemented this prototype with a separate look and feel prototype that explained their intentions in that dimension.

A fluid dynamics simulation system

Example 12 shows a small part of the C++ program listing for a system for simulating gas flows and combustion in car engines, part of an engineer-

```
IntList& IntList::operator=(const IntList& oldList)
{
register long n = oldList.size;
  if (n != size) setSize(n);
  register int* newPtr = &values[n];
  register int* oldPtr = &oldList.values[n];
  while (n--) *--newPtr = *--oldPtr;
  return *this;
}
```

Example 12. C++ program sample from a fluid dynamics simulation system [E12 Hill, 1993].

ing research project (Hill, 1993). One goal of this prototype was to demonstrate the feasibility of object-oriented programming using the C++ language in place of procedural programs written in the older FORTRAN language. Object-oriented programming can in theory lead to increased software reuse, better reliability and easier maintenance. Since an engine simulation may take a week to run on the fastest available computers and is extremely memory-intensive, it was important to show that the new approach did not incur excessive performance or memory overheads. The program listing shown was the implementation of the operation to copy one list of numbers to another. When tested, it was shown to be faster than the existing FORTRAN implementation. The prototype was built primarily for the design team's own use, and eventually used to create a deployable system. The marker in Figure 6 indicates that this prototype primarily explored implementation.

Other kinds of implementation prototypes include demonstrations of new algorithms (e.g., a graphical rendering technique or a new search technology), and trial conversions of existing programs to run in new environments (e.g., converting a program written in the C language to the Java language).

Implementation prototypes can be hard to build, and since they actually work, it is common for them to find their way directly into the final system. Two problems arise from this dynamic: firstly, programs developed mainly to demonstrate feasibility may turn out in the long term to be difficult to maintain and develop; and secondly, their temporary user interfaces may never be properly redesigned before the final system is released. For these reasons it is often desirable to treat even implementation prototypes as disposable, and to migrate successful implementation designs to a new integrated prototype as the project progresses.

4.4 Integration prototypes

Integration prototypes are built to represent the complete user experience of an artifact. Such prototypes bring together the artifact's intended design in terms of role, look and feel, and implementation. Integrated prototypes help designers to balance and resolve constraints arising in different design dimensions; to verify that the design is complete and coherent; and to find synergy in the design of the integration itself. In some cases the integration design may become the unique innovation or feature of the final artifact. Since the user's experience of an artifact ultimately combines all three dimensions of the model, integration prototypes are most able to accurately simulate the final artifact. Since they may need to be as complex as the final artifact, they are the most difficult and time consuming kinds of prototypes to build. Designers make integration prototypes to understand the design as a whole, to show their organizations a close approximation to the final artifact, and to get feedback from users about the overall design.

The Sound Browser

The "SoundBrowser" prototype shown in Example 13 was built as part of a larger project which investigated uses of audio for personal computer users (Degen et al, 1992). The prototype was built in C

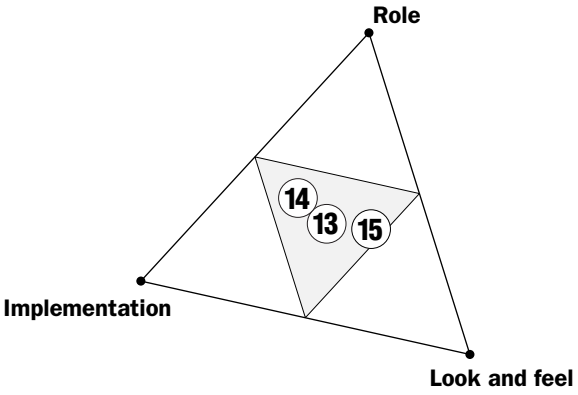
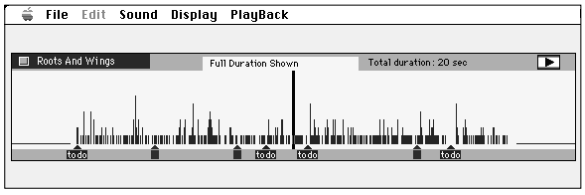


Figure 7. Relationship of integration prototypes (Examples 13-15) to the model.

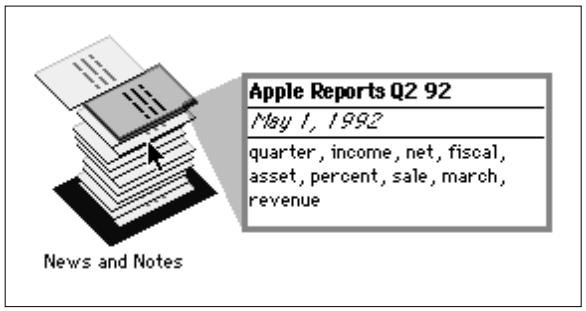
to run on a Macintosh. It allowed a user to browse digital audio data recorded on a special personal tape recorder equipped with buttons for marking points in the audio. The picture shows the SoundBrowser's visual representation of the audio data, showing the markers below the sound display. A variety of functions were provided for reviewing sound, such as high-speed playback and playback of marked segments of audio.

This prototype earns a position right in the center of the model, as shown in Figure 7. All three dimensions of the model were explored and represented in the prototype. The role of the artifact was well thought-out, being driven initially by observations of what users currently do to mark and play back audio, and then by iteratively designed scenarios of how it might be done more efficiently if electronic marking and viewing functions were offered. The look and feel of the prototype went through many visual design iterations. The implementation was redesigned several times to meet the performance needs of the desired high-speed playback function.

When the SoundBrowser was near completion it was prepared for a user test. One of the features which the design team intended to evaluate was the visual representation of the sound in the main window. They wanted to show users several alternatives to understand their preferences. The programmer who built the SoundBrowser had developed most of the alternatives. In order to refine these and explore others, two other team members copied screen-shots from the tool into a pixel-painting application, where they experimented with modifications. This was a quick way to try out different visual options, in temporary isolation from other aspects of the artifact. It was far easier to do this in a visual design tool than by programming in C. When finished, the new options were programmed



Example 13. Integrated prototype of a sound browser [E13 Degen 1993].



Example 14. Integration prototype of the "pile" metaphor for information retrieval [E14 Rose, 1993].

into the integrated prototype. This example shows the value of using different tools for different kinds of design exploration, and how even at the end of a project simple, low-fidelity prototypes might be built to solve specific problems.

The Pile Metaphor

The prototype shown in Example 14 was made as part of the development of the "pile" metaphor—a user interface element for casual organization of information (Mander et al, 1992, Rose et al, 1993). It represented the integration of designs developed in several other prototypes which independently explored the look and feel of piles, "content-aware" information retrieval, and the role that piles could play as a part of an operating system. In the pile metaphor, each electronic document was represented by a small icon or "proxy", several of which were stacked to form a pile. The contents of the pile could be quickly reviewed by moving the arrow cursor over it. While the cursor was over a particular document, the "viewing cone" to the right displayed a short text summary of the document.

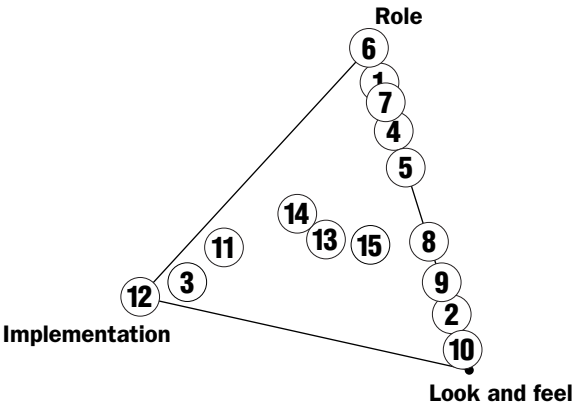
This prototype was shown to designers, project managers, and software developers as a proof of concept of the novel technology. The implementation design in this prototype might have been achieved with virtually no user interface: just text input and output. However, since the prototype was to be shown to a broad audience, an integrated style of prototype was chosen, both to communicate the implementation point and to verify that the piles representation was practically feasible. It helped greatly that the artifact's role and look and feel could be directly inherited from previous prototypes. Figure 7 shows its marker on the model.

A garment history browser

The prototype in Example 15 was a working system which enabled users to enter and retrieve snippets of information about garment designs via a visually rich user interface (Hill et al, 1993; Scaife et al, 1994). The picture shows the query tool which was designed to engage fashion designers and provide memorable visual cues. The prototype was designed for testing in three corporations with a limited set of users' actual data, and presented to users in interviews. It was briefly demonstrated, then users were asked to try queries and enter remarks about design issues they were currently aware of.

This prototype was the end-result of a progression from an initial focus on role (represented by verbal usage scenarios), followed by rough look and feel prototypes and an initial implementation. Along the way various ideas were explored, refined or rejected. The working tool, built in Allegiant SuperCard™, required two months' intensive work by two designers. In retrospect the designers had mixed feelings about it. It was highly motivating to users to be able to manipulate real user data through a novel user interface, and much was learned about the design. However, the designers also felt that they had to invest a large amount of time in making the prototype, yet had only been able to support a very narrow role compared to the breadth shown in the animation shown in Example 8. Many broader design questions remained unanswered.

Example 15. Integrated prototype of a garment history browser [E15 Hill and Kamlish 1992].



1. *3D space-planning (role)*
2. *3D space-planning (look and feel)*
3. *3D space-planning (implementation)*
4. *Storyboard for portable notebook computer*
5. *Interactive story, operating system user interface*
6. *Vision video, notebook computer*
7. *Appearance model, integrated communicator*
8. *Animation, fashion design workspace*
9. *Look and feel simulation, child's toy*
10. *Pizza-box, architect's computer*
11. *Working prototype, digital movie editor*
12. *C++ program listing, fluid dynamics simulation*
13. *Integrated prototype, sound browser*
14. *Integrated prototype, pile metaphor*
15. *Integrated prototype, garment history browser*

Figure 8. Relationship of all examples to the model.

5. SUMMARY

In this chapter, we have proposed a change in the language used by designers to think and talk about prototypes of interactive artifacts. Much current terminology centers on attributes of prototypes themselves: the tools used to create them, or how refined-looking or -behaving they are. Yet tools can be used in many different ways, and resolution can be misleading. We have proposed a shift in attention to focus on questions about the design of the artifact itself: What role will it play in a users life? How should it look and feel? How should it be implemented? The model that we have introduced can be used by designers to divide any design problem into these three classes of questions, each of which may benefit from a different approach to prototyping.

We have described a variety of prototypes from real projects, and have shown how the model can be used to communicate about their purposes. Several practical suggestions for designers have been raised by the examples:

• *Define "prototype" broadly.* Efficient prototypes produce answers to their designers' most important questions in the least amount of time. Sometimes very simple representations make highly effective prototypes: e.g., the pizza-box prototype of an architect's computer [Example 10] and the storyboard notebook [Example 1]. We define a prototype as any representation of a design idea—regardless of medium; and designers as the people who create them—regardless of their job titles.

• *Build multiple prototypes.* Since interactive artifacts can be very complex, it may be impossible to create an integrated prototype in the formative stages of a project, as in the 3D space-planning example [Examples 1, 2, and 3]. Choosing the right focused prototypes to build is an art in itself. Be prepared to throw some prototypes away, and to use different tools for different kinds of prototypes.

• *Know your audience.* The necessary resolution and fidelity of a prototype may depend most on the nature of its audience. A rough role prototype such as the interactive storyboard [Example 4] may work well for a design team but not for members of the supporting organization. Broader audiences may require higher-resolution representations. Some organizations expect to see certain kinds of prototypes: implementation designs are often expected in engineering departments, while look-and-feel and role prototypes may rule in a visual design environment.

• *Know your prototype; prepare your audience.* Be clear about what design questions are being explored with a given prototype—and what are not. Communicating the specific purposes of a prototype to its audience is a critical aspect of its use. It is up to the designer to prepare an audience for viewing a prototype. Prototypes themselves do not necessarily communicate their purpose. It is especially important to clarify what is and what is not addressed by a prototype when presenting it to any audience beyond the immediate design team.

By focusing on the purpose of the prototype—that is, on what it prototypes—we can make better decisions about the kinds of prototypes to build. With a clear purpose for each prototype, we can better use prototypes to think and communicate about design.

6. ACKNOWLEDGMENTS

Special thanks are due to Thomas Erickson for guidance with this chapter, and to our many colleagues whose prototypes we have cited, for their comments on early drafts. We would also like to acknowledge S. Joy Mountford whose leadership of the Human Interface Group at Apple created an atmosphere in which creative prototyping could flourish. Finally, thanks to James Spohrer, Lori Leahy, Dan Russell, and Donald Norman at Apple Research Labs for supporting us in writing this chapter.

6.1 Prototype Credits

We credit here the principal designer and design team of each example prototype shown.

[E1] Stephanie Houde, [E2] Stephanie Houde, [E3] Michael Chen. © Apple Computer, Inc., 1990. Project team: Penny Bauersfeld, Michael Chen, Lewis Knapp (project leader), Laurie Vertelney and Stephanie Houde.

[E4] Laurie Vertelney. © Apple Computer Inc., 1990. Project team: Michael Chen, Thomas Erickson, Frank Leahy, Laurie Vertelney (project leader).

[E5] Laurie Vertelney and Yin Yin Wong. © Apple Computer Inc., 1990. Project team: Richard Mander, Gitta Salomon (project leader), Ian Small, Laurie Vertelney, Yin Yin Wong.

[E6] Hugh Dubberly and Doris Mitch. © Apple Computer, Inc., 1987. The Knowledge Navigator. (Video.)

[E7] Masamichi Udagawa. © Apple Computer Inc., 1995. Project team: Charles Hill, Heiko Sacher, Nancy Silver, Masamichi Udagawa.

[E8] Charles Hill. © Royal College of Art, London, 1992. Project team: Gillian Crampton Smith, Eleanor Curtis, Charles Hill (Royal College of Art), Mike Scaife (Sussex University), and Philip Joe (IDEO, London).
[E9] Tom Bellman, Byron Long, Abba Lustgarten. University of Toronto, Apple Design Project. © Apple Computer Inc., 1993.

[E10] Apple Design Project. © Apple Computer, Inc., 1992.

[E11] Leo Degen. © Apple Computer Inc., Project team: Leo Degen, Stephanie Houde, Michael Mills (team leader), David Vronay.

[E12] Charles Hill. Doctoral thesis. Project team: Charles Hill, Henry Weller. © University of London, 1993.

[E13] Leo Degen. © Apple Computer Inc., 1193. Project team: Leo Degen, Richard Mander, Gitta Salomon (team leader), Yin Yin Wong.

[E14] Daniel Rose. © Apple Computer, Inc., 1993. Project team: Penny Bauersfeld, Leo Degen, Stephanie Houde, Richard Mander, Ian Small, Gitta Salomon (team leader), Yin Yin Wong

[E15] Charles Hill and Stephen Kamlish. © Royal College of Art, London, 1992. Project team: Gillian Crampton Smith, Eleanor Curtis, Charles Hill, Stephen Kamlish (Royal College of Art), and Mike Scaife (Sussex University).

6.2 References

Degen, L., Mander, R., Salomon, G. (1992). *Working with Audio: Integrating Personal Tape Recorders and Desktop Computers*. Human Factors in Computing Systems: Proc. CHI'92. New York: ACM, pp. 413-418.

Dubberly, H. and Mitch, D. (1987). *The Knowledge Navigator*. Apple Computer, Inc. videotape.

Ehn, P., Kyng, M. (1991). *Cardboad Computers: Mocking-it-up or Hands-on the Future*. Design at Work: Cooperative Design of Computer Systems (ed. Greenbaum, J., and Kyng, M.). Hillsdale, NJ: Lawrence Erlbaum. pp. 169-195.

Erickson, T. (1995). *Notes on Design Practice: Stories and Prototypes as Catalysts for Communication.* "Envisioning Technology: The Scenario as a Framework for the System Development Life Cycle" (ed. Carroll, J.). Addison-Wesley.

Hill, C. (1993). *Software Design for Interactive Engineering Simulation.* Doctoral Thesis. Imperial College of Science, Technology and Medicine, University of London.

Hill, C., Crampton Smith, G., Curtis, E., Kamlish, S. (1993). *Designing a Visual Database for Fashion Designers*. Human Factors in Computing Systems: Adjunct Proc. INTERCHI'93. New York, ACM, pp. 49-50.

Houde, S. (1992). *Iterative Design of and Interface for Easy 3-D Direct Manipulation. Human Factors in Computing Systems*. Proc. CHI'92. New York: ACM, pp. 135-142.

I.D.Magazine. *Apple's Shared Conceptual Model*. The International Design Magazine: 41st. Annual Design Review, July-August 1995. USA. pp. 206-207

Kim, S. (1990). *Interdisciplinary Collaboration*. The Art of Human Computer Interface Design (ed. B. Laurel). Reading, MA: Addison-Wesley. pp.31-44.

Mander, R., Salomon, G., Wong, Y.Y. (1992). *A 'Pile' Metaphor for Supporting Casual Organization of Information*. Human Factors in Computing Systems: Proc. CHI'92. New York: ACM, pp. 627-634.

Rose, D.E., Mander, R., Oren, T., Ponceleón, D.B., Salomon, G., Wong, Y. (1993). *Content Awareness in a File System Interface: Implementing the 'Pile' Metaphor for Organizing Information*. Research and Development in Information Retrieval: Proc. SIGIR. Pittsburgh, PA: ACM, pp. 260-269.

Scaife, M., Curtis, E., Hill, C. (1994). *Interdisciplinary Collaboration: a Case Study of Software Development for Fashion Designers*. Interacting with Computers, Vol. 6. no. 4, pp. 395-410

Schrage, M. (1996). *Cultures of Prototyping. Bringing Design to Software* (ed. T. Winograd). USA: ACM Press. pp. 191-205.

Wong, Y.Y. (1992). *Rough and Ready Prototypes: Lessons from Graphic Design*. Human Factors in Computing Systems: Proc. CHI'92, Posters and Short-Talks, New York: ACM, pp. 83-84.