CHAPTER

Prototyping

20

Highlights

- Depth and breadth of prototypes.
- Fidelity of prototypes.
- Wireframe prototypes.
- Wireflow prototypes.
- Building up increasing levels of fidelity.
- Specialized prototypes.

20.1 INTRODUCTION

20.1.1 You Are Here

We begin each process chapter with a "you are here" picture of the chapter topic in the context of The Wheel, the overall UX design lifecycle template (Fig. 20-1). In this chapter, we describe how to perform the Prototype Candidates UX design lifecycle activity.

This is the chapter describing types of prototypes and how to make wireflow and wireframe prototypes within the Prototype Candidates lifecycle activity.

20.1.2 Prototyping Intertwines with Other UX Activities

Back in Chapter 5, the *Prelude to the process chapters*, we said that we have to describe each lifecycle activity separately in its own chapter(s), but they actually all go together and happen in combination—closely intertwined and interleaved throughout the lifecycle.

Prototyping is a good example of this intertwining. See Chapter 14 on generative design where prototyping occurs right from the start of design creation in the form of sketches and continues to occur as wireframes and other forms throughout much of the remaining design process.

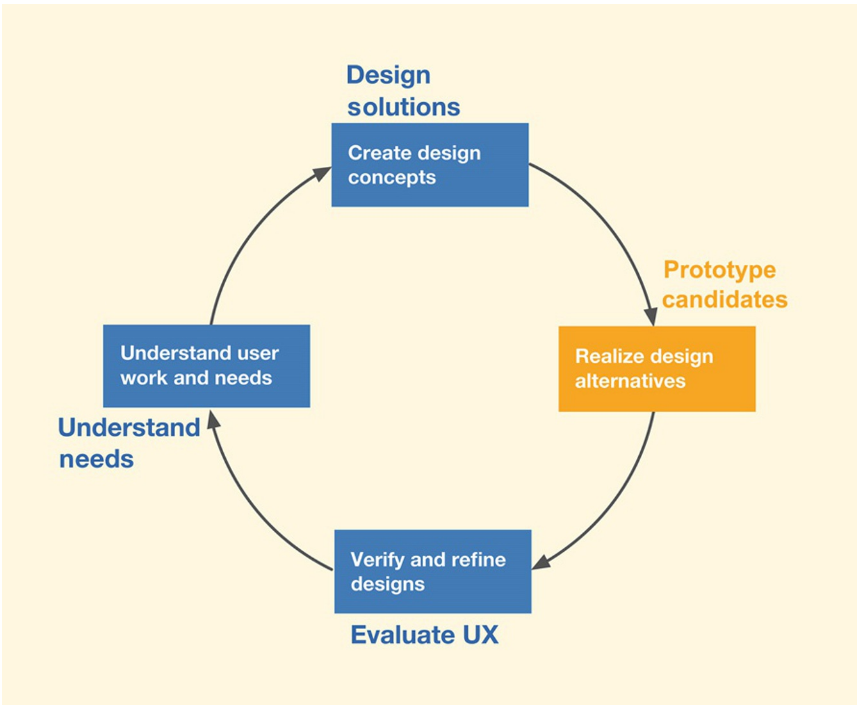


Fig. 20-1

You are here in the chapter on the Prototype Candidates lifecycle activity in the context of the overall Wheel UX lifecycle process.

20.1.3 A Dilemma and a Solution

The only way to be sure that your design is the right design and the best design it can be is to subject it to UX evaluation. However, at some early point, you will have a design but no product or system yet to evaluate. But if you wait until after it is implemented, changes are much more difficult and expensive to make.

A prototype gives you something to evaluate before you have to commit resources to build the real thing. Prototypes allow you to fail faster, learn sooner, and succeed earlier.

20.1.4 Advantages of Prototyping

Prototypes have these advantages:

- Provide a platform to support UX evaluation with users.
- Offer concrete baseline for communication between users and designers.
- Provide a conversational "prop" to support communication of concepts not easily conveyed verbally.

- Allow users to "take the design for a spin" (who would buy a car without taking it for a test drive or buy a stereo system without first listening to it?).
- Give project visibility and buy-in within customer and developer organizations.
- Encourage early user participation and involvement.
- Give the impression that design is easy to change because a prototype is obviously not finished.
- Afford designers immediate observation of user performance and consequences of design decisions.
- Help sell management on an idea for new product.
- Help affect a paradigm shift from an existing system to a new system.

Paradigm

A model, pattern, template, or intellectual perception or view guiding a way of thinking and doing. Historically, with respect to a field of thought and work, it is thought of as coming in waves over time (Section 6.3).

20.1.5 Universality of Prototyping

The idea of prototyping is timeless and universal. Automobile designers build and test mockups, architects and sculptors make models, circuit designers use "bread-boards," artists work with sketches, and aircraft designers build and fly experimental designs. Even Leonardo da Vinci and Alexander Graham Bell made prototypes.

Thomas Edison is famous for making thousands of prototypes before getting just the right design. In each case, the concept of a prototype was the key to affording the design team and others an early ability to observe something about the final product—evaluating ideas, weighing alternatives, and seeing what works and what does not.

Alfred Hitchcock, master of dramatic dialogue design, is known for using prototyping to refine the plots of his movies. Hitchcock would tell variations of stories at cocktail parties and observe the reactions of his listeners. He would experiment with various sequences and mechanisms for revealing the storyline. Refinement of the story was based on listener reactions as an evaluation criterion. The movie *Psycho* is a notable example of the results of this technique.

20.1.6 Scandinavian Origins of Prototyping

Like a large number of other parts of this overall lifecycle process, the origins of prototyping in UX, especially low-fidelity prototyping, go back to the Scandinavian work activity theory research and practice of Ehn, Kyng, and others (Bjerknes, Ehn, & Kyng, 1987; Ehn, 1988) as well as participatory design work (Kyng, 1994). These formative works emphasized the need to foster early and detailed communication about design and participation in understanding the requirements for that design.

Activity theory

An abstract adaptation for HCI of a theoretical descriptive framework based on the Soviet psychological activity theory as adapted by Scandinavian researchers in the mid-1980s (Section 11.3.1).

20.2 DEPTH AND BREADTH OF A PROTOTYPE

The idea behind prototypes is to provide fast and easily changed early views of an envisioned UX design. *Because it must be quickly and easily changed, a prototype is a design representation that is in some way(s) less than a full implementation.* The choices for your approach to prototyping are about *how* to make it less. One way you can make it less is by focusing on just the breadth or just the depth of the system.

When you slice the features and functionality of a system by breadth, you get a horizontal prototype. And when you slice by depth, you get a vertical prototype (Hartson & Smith, 1991). In his usability engineering book, Nielsen (1993) illustrates the relative concepts of horizontal and vertical prototyping, which we show as Fig. 20-2.

20.2.1 Horizontal Prototypes

A horizontal prototype (top “bar” in Fig. 20-2) is very broad in the features it incorporates, but offers less depth in its coverage of how that functionality works. A horizontal prototype is a good place to start with your prototyping, as it provides an overview on which you can base a top-down approach. A horizontal prototype is effective in demonstrating the product concept and for conveying an early product overview to managers, customers, and users (Kensing & Munk-Madsen, 1993). However, because of the lack of details in depth, horizontal prototypes usually do not support complete workflows, and user experience evaluation with this kind of prototype is generally less realistic. For these reasons, prototyping in the early funnel tends to be horizontal in nature.

Early funnel

The part of the funnel (agile UX model) for large-scope activity, usually for conceptual design, before syncing with software engineering (Section 4.4.4).

20.2.2 Vertical Prototypes

A vertical prototype (upright “bar” in Fig. 20-2) contains more depth of detail for some functionality, but only for a narrow selection of features. A vertical prototype allows testing a limited range of features but those functions

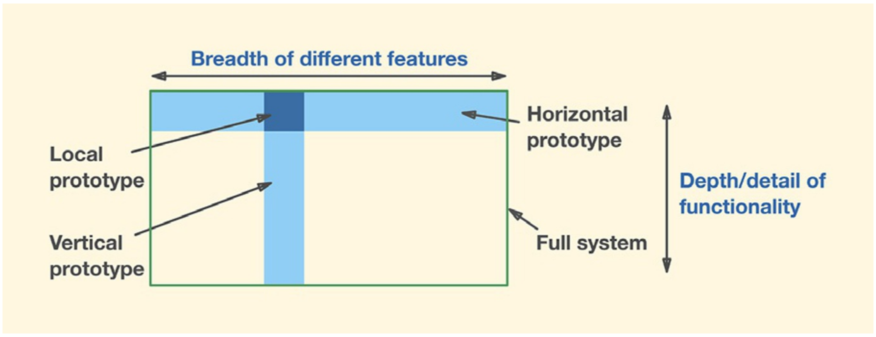


Fig. 20-2

Horizontal and vertical prototyping concepts, adapted from Nielsen (1993), with permission.

that are included are evolved in enough detail to support realistic user experience evaluation. Often the functionality of a vertical prototype can include a stub for or a connection to an actual working backend database.

A vertical prototype is ideal for times when you need to represent completely the details of an isolated part of an individual interaction workflow in order to understand how those details play out in actual usage. For example, you may wish to study a new design for the checkout part of the workflow for an e-commerce website. A vertical prototype might show that one task sequence and associated user actions, in depth. Because vertical prototypes are usually about individual features, they are most commonly used in the late funnel part of the process.

Late funnel

The part of the funnel (agile UX model) for small-scope activity and for syncing with agile software engineering sprints (Section 4.4.3).

20.2.3 Local Prototypes

Sometimes you need a "local prototype," a prototype that is narrow in both dimensions, limiting its focus to a localized interaction design issue. A local prototype is used to evaluate design alternatives for particular isolated interaction details, such as one dialogue box, the appearance of an icon, the wording of a message, or the behavior of an individual interaction object.

A local prototype is the solution for those times when your design team encounters an impasse in design discussions where, after a while, there is no agreement and people are starting to repeat themselves. Perhaps usage research data are not clear on the question and further arguing is a waste of time. It is time to put the specific design issue on a list for testing, letting the user or customer speak to it in a kind of "feature face-off" to help decide among the alternatives.

Because of their role in deciding specific issues, local prototypes are used independently from other prototypes and are temporary and disposable, having short life spans.

20.2.4 "T" Prototypes

A "T" prototype combines advantages of both the horizontal and vertical prototypes (see the T-shaped shaded area of Fig. 20-2), offering a good compromise for design evaluation. Much of the feature breadth is realized at a shallow level (the top of the T), but a few parts are done in depth (the vertical part of the T).

In the early going, we recommend the T prototype because it provides a nice balance between the two extremes, giving you some advantages of each. Once you have established a system overview in your horizontal prototype, as a practical matter the T prototype is the next step toward achieving some depth. In time, the horizontal foundation supports evolving vertical growth across the whole prototype.

20.3 FIDELITY OF PROTOTYPES

In addition to depth and breadth, the level of fidelity of a prototype is another dimension along which there are tradeoffs with respect to completeness and cost/time. *The fidelity of a prototype reflects how "finished" it is perceived to be by customers and users* (Tullis, 1990). Being "finished" applies to completeness of content and functionality as well as how refined it is in appearance.

In general, lower fidelity prototypes are less finished but more flexible and can be constructed more rapidly and at less cost. But, as you progress through stages of development in your project, your need for fidelity in prototypes increases. The level of fidelity to aim for depends on your current stage of progress in the project and the purpose for which you plan to use the prototype.

There are many ways that levels of fidelity in a prototype have been described and used in the past. Here we focus on the practical role they play in an agile UX process. In Section 20.5 we describe these levels of prototype fidelity in the context of their various purposes.

20.4 WIREFRAME PROTOTYPES

Wireframes are now the go-to prototyping technique in UX practice. The bulk of the wireframe prototypes will be made during interaction design creation (Chapter 14).

20.4.1 What is a Wireframe?

A wireframe is a sketch, image, or prototype of a single interaction page or screen (in the broadest sense of "screen").

As we said in Section 17.5, wireframes are described as two-dimensional sketches or drawings consisting of lines, arcs, and vertices (thus the name wireframe), plus some text for labels, representing the layout of an interaction design for a page or screen. These wireframes are best generated with a software tool (such as Sketch).[1]

20.4.2 Wireframe Design Elements

Low-fidelity wireframes usually do not have graphical design elements such as images or specific colors or typography. Typical elements represented in a wireframe can include:

[1]https://www.sketchapp.com/

- Header.
- Footer.
- Content areas.
- Labeling.
- Menus.
- Tabs (possibly with drop-downs).
- Buttons.
- Icons.
- Pop-ups.
- Messages.
- Navigation bar, navigation links.
- Placeholders for logo and branding images.
- Search field.

Draw on everything you have worked on so far for the design. Use your conceptual design, design scenarios, ideation, personas, storyboards, and everything else you have created in working up to this first real materialization of your design ideas.

20.4.3 Wireflow Prototypes

The most common term UX professionals use in the context of prototyping using boxes, arrows, and other simple shapes is a wireframe. Even though in industry practice, the plural form of that term, *wireframes*, is used to denote flows and sequences of individual wireframes, the more accurate term is a wireflow.

20.4.3.1 What is a wireflow prototype?

Simply put, a wireflow[2] prototype (Fig. 20-3), or wireflow for short, is a prototype that illustrates navigational flow within an interaction design. Structurally, a wireflow is a directed graph in which:

- The nodes are wireframes.
- The arcs are arrows representing navigational flow among the wireframes.

As we will describe soon in more detail, a wireflow prototype is, at its base, a state diagram of user workflow.

Notice that the flow arrows go from interaction objects (such as a button or icon) that users can act upon (e.g., click) within one wireframe to navigate to subsequent wireframes.

Storyboard

A visual scenario in the form of a series of sketches or graphical clips, often annotated, in cartoon-like frames, illustrating the interplay between a user and an envisioned ecology or device (Section 17.4.1).

State diagram (in UX)

A directed graph in which nodes are states which correspond to screens (in the broadest sense), and arcs (or arrows) are transitions between states resulting from user actions or system events. Used in wireflow and wireframe prototypes to show navigation among screens (Sections 9.7.6 and 20.4.4.2).

[2] http:/nform.com/cards/wireflow/, https:/www.nngroup.com/articles/wireflows/

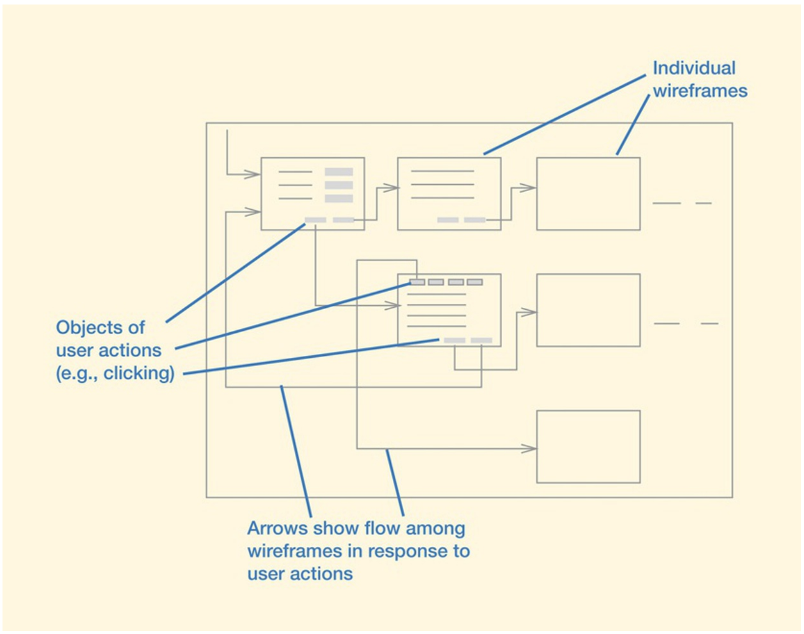

Individual wireframes
Objects of user actions (e.g., clicking)
Arrows show flow among wireframes in response to user actions

Fig. 20-3

Generic wireflow diagram.

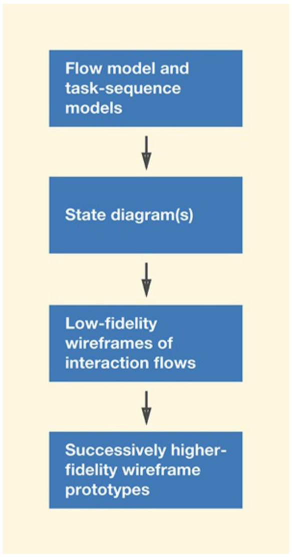

Flow model and task-sequence models
State diagram(s)
Low-fidelity wireframes of interaction flows
Successively higher-fidelity wireframe prototypes

Fig. 20-4

The general evolutionary process of going from usage research models to wireframes.

Most of the configurations of wireframes that serve as UX prototypes are actually wireflow prototypes, which include both the individual wireframes and the navigational arrows connecting them. To simplify the terminology, we will mostly use the industry term “wireframes” or “wireframe deck” instead of “wireflow” to refer to the whole prototype.

20.4.4 General Process of Representing Interaction

The evolutionary process of establishing the interaction design for the set of related tasks of a feature is summarized in Fig. 20-4.

20.4.4.1 Focus on user workflow

Start by looking in detail at user workflow and navigation (in the most general sense). The flow model (a simple graphical representation giving an overview of how information, artifacts, and work products flow among user work roles and parts of the product or system, as the result of user actions, Section 9.5) is a starting point. More detailed task sequence models will fill in specifics about user actions and resulting navigation paths (top of Fig. 20-4).

20.4.4.2 Represent flow and navigation with state diagrams

Using your task-sequencing models, the next step in prototyping task sequencing and navigation is the creation of one or more state diagrams (next node down in Fig. 20-4), which take us a step closer to design by helping us represent details of flow and navigation in the interaction view of design.

Start with the main navigational paths, the essence of the flow, and initially leave out unnecessary detail, special conditions, and edge cases, such as error checking, confirmation dialogue, etc.

Example: State Diagram for Bundling Network Services[3]

The Network Infrastructure and Services group, part of the IT organization at Virginia Tech, is responsible for providing network and communications services to faculty, staff, and students. They maintain systems for ordering, billing, and maintaining services such as wireless Internet access for buildings and classrooms and wired ethernet for faculty offices and laboratories. Services also include all campus telephones and cable TV for classrooms.

As part of their mission, hardware, software, and UX people develop systems to support customers in ordering of network services. This example is about a web-based feature for bundling services together in popular configurations. To understand this example, you need to know that each service can be offered in a "plan" (e.g., a choice of ethernet connection speed) and each plan can have what they call "add ons," which are additional related features. A specialist in the Network Infrastructure and Services group is authorized to create these bundles, which can then be ordered by customers.

The system feature that supports bundle creation was the target of agile UX design. Starting with raw usage research data from interviews with these specialists, we created a flow model and several task sequence models. These were used to create an early state diagram (Fig. 20-5) to illustrate the main sequencing of workflow for this user task.

Starting at the "Bundles Landing Page," users can view, delete, or edit an existing bundle or create a new bundle. Editing a bundle includes editing bundle attributes (e.g., name, cost) and editing bundle contents (e.g., services, plans, and add ons in the bundle). Once those changes are saved, the user is taken back to the "Bundles Landing Page." It looks straightforward in this diagram and maybe that is a testimony to the power of a good state diagram because it took

[3]Thanks for permission to use this example to Joe Hutson and Mathew Mathai, Network Infrastructure and Services, Virginia Tech.

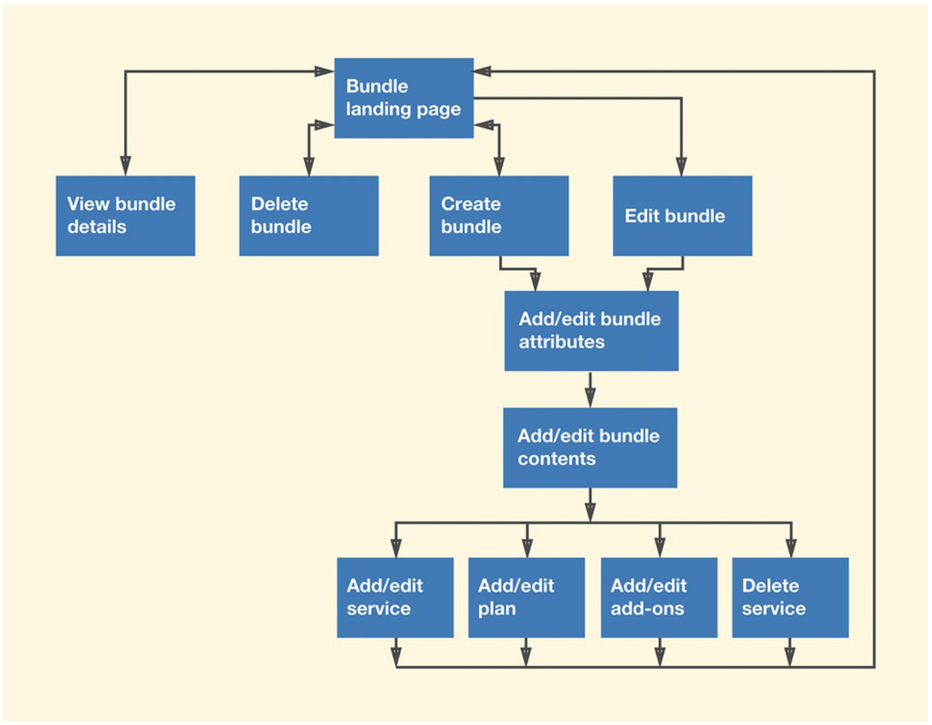


Fig. 20-5

Early state diagram for service bundling (Thanks for permission to use this example to Joe Hutson and Mathew Mathai, Network Infrastructure and Services, Virginia Tech).

quite a bit of analysis and synthesis of the disparate pieces of usage research data to get to this tidy representation.

In early design, your state diagrams can be translated almost directly into the structure of a wireframe deck (next section).

20.4.5 Create a Wireframe for Each State

Each state of the state diagram or box in the wireflow diagram becomes a low-fidelity wireframe design for one "screen." Within that screen, you are designing for things that "live" in that state, including the work spaces and dialogue to support the related task. Arcs of the state diagram will guide you in adding the controls (e.g., buttons) to support navigation among the wireframes.

As you work, you will probably have to add new states and the corresponding navigation to handle nonmainstream dialogue (e.g., to handle confirmation requests such as "Are you sure you want to delete this service?").

After you have fleshed out the details of the individual screens, you get a higher-fidelity version to see the wireframes in the navigational context (Fig. 20-6). Space limitations prevent showing this large enough to read the text, but you get the idea.

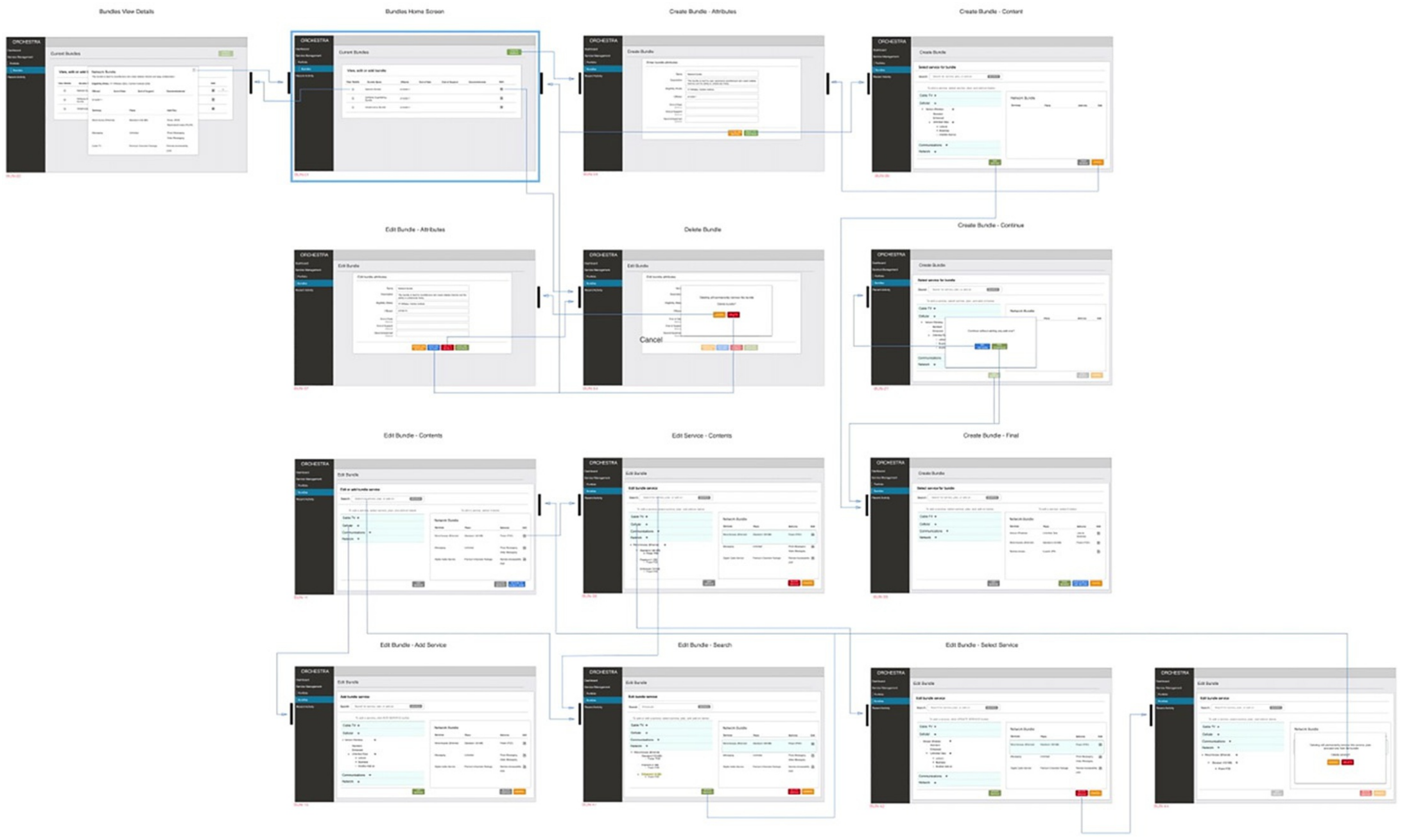


Fig. 20-6

High-fidelity wireframes showing navigation (Thanks for permission to use this example to Joe Hutson and Mathew Mathai, Network Infrastructure and Services, Virginia Tech).

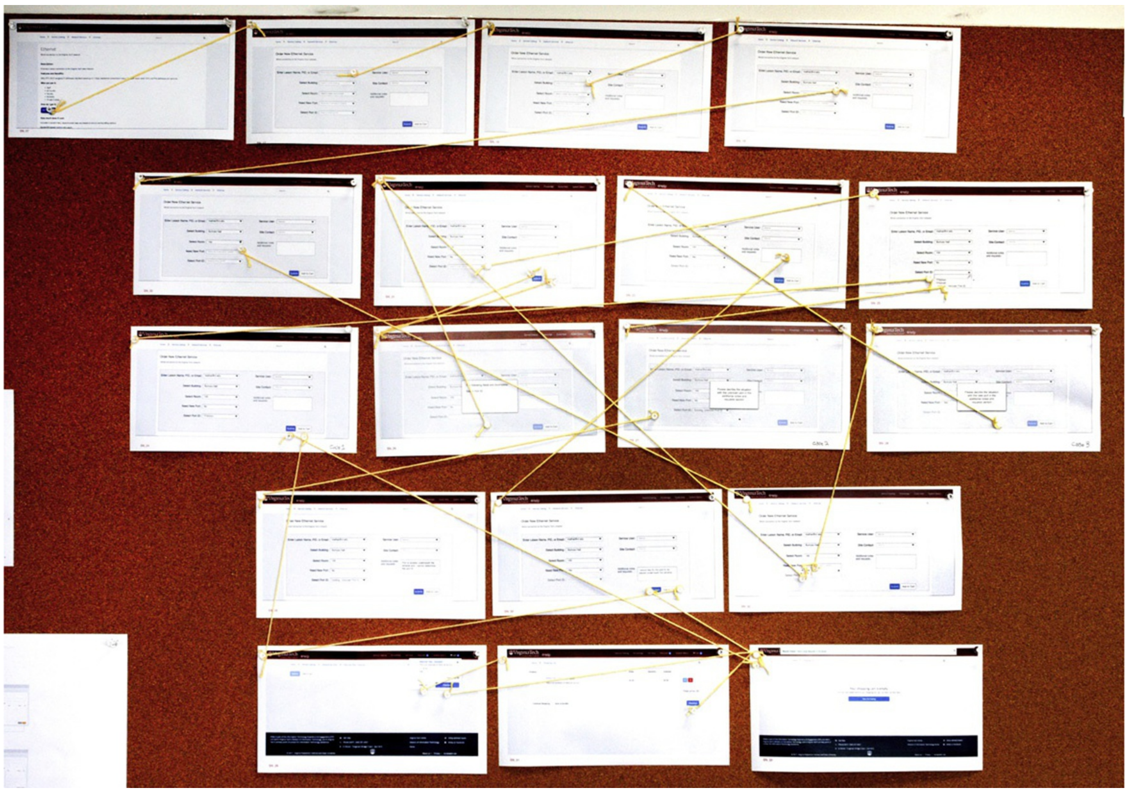


Fig. 20-7

Posting a readable wireflow diagram using colored yarn to represent navigational connections between individual wireframes (Thanks for permission to use this example to Joe Hutson and Mathew Mathai, Network Infrastructure and Services, Virginia Tech).

We posted this high-fidelity wireframe deck in our UX studio as a work artifact around which we had design discussions. To print the diagram large enough for everyone to read, we printed each wireframe on a sheet of paper and taped them together.

Another technique we have used is to print each wireframe on a separate sheet of paper, post them on a cork board, and represent the navigational connections with yarn held in place by push pins (Figs. 20-7 and 20-8).

20.5 BUILD UP PROTOTYPES IN INCREASING LEVELS OF FIDELITY

In this section we develop a sequence of the kind of prototypes we would use in a typical UX project to illustrate an increasing level of fidelity in the context of a UX design.

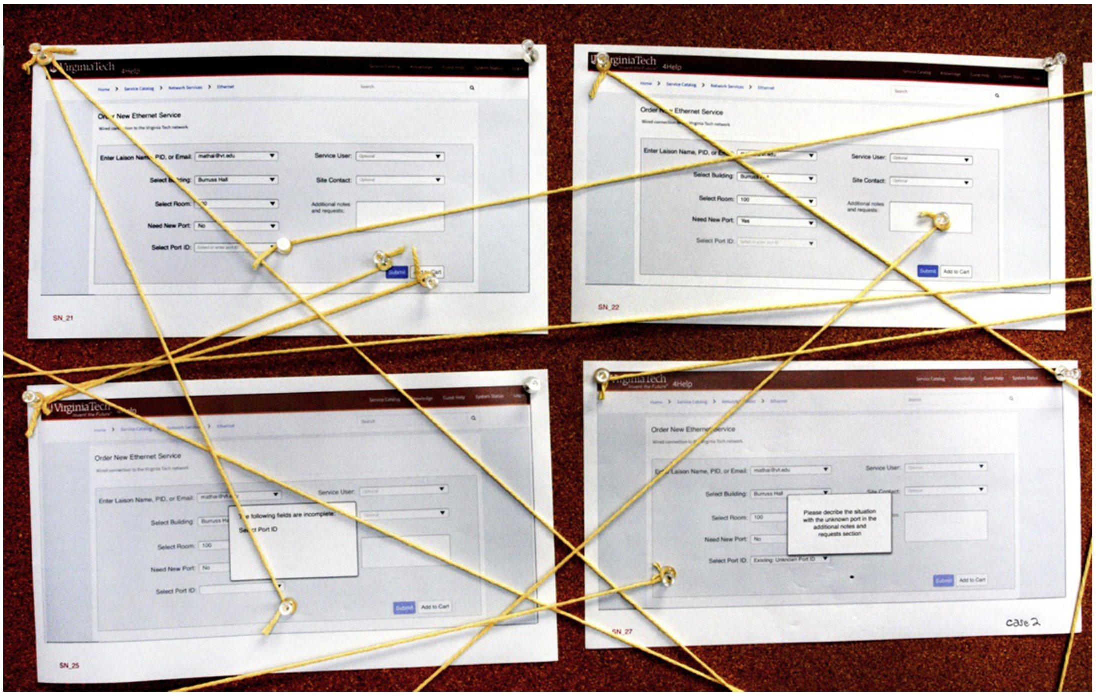


Fig. 20-8

Close-up of wireflow diagram using colored yarn to represent navigational connections (Thanks for permission to use this example to Joe Hutson and Mathew Mathai, Network Infrastructure and Services, Virginia Tech).

20.5.1 High-Level Task Context

In the previous section, we talked about state diagrams and interaction flow wireframes. Such broad representations of the design are helpful to envision high-level conceptual design and the breadth of the emerging design. This will take the user from the "home page" or beginning of an application down into the task hierarchy to the context of where they will be performing individual tasks of the feature being designed. This part in itself has no detailed task representation, just context.

In much of what we do with adding to or updating existing systems, this is unnecessary because the context is already well established.

20.5.2 Very Low-Fidelity Wireframe Sketches to Support Design Idea Exploration in Generative Design

20.5.2.1 The nature of low-fidelity prototypes

Low-fidelity prototypes are, as the term implies, prototypes that are not faithful representations of the details of look, feel, and behavior, but instead give rather high-level, more abstract impressions of the intended design. The low-fidelity versions usually do not have set graphical design elements such as images, color, or typography. Low-fidelity prototypes are appropriate when design details have not been decided or when they are likely to change. They are flexible and easy to change, costing almost nothing to iterate. Therefore, these are essentially sketches that come and go as part of ideation, sketching, and critiquing.

20.5.2.2 The first level of fidelity

At the first level of fidelity in prototypes are very-low-fidelity wireframe sketches to support ideation and sketching as the UX design team explores design ideas in generative design. These quick and "dirty" design representations are almost always just hand sketches. They are disposable and short lived and are changing and evolving rapidly.

20.5.2.3 Decks of wireframes

If you group related wireframes in a sequence, we call this a *deck* of wireframes. When demonstrating the flow of this design, you can move through the deck one wireframe at a time, simulating a potential scenario by pretending to click on interaction widgets on the screen. These page sequences can represent the flow of user activity within a scenario, but cannot show all possible navigational paths.

A designer can narrate a design scenario where user actions cause the navigation to progress through the corresponding images in a deck.

20.5.3 Static Low-Fidelity Wireframes to Summarize and Solidify Design with UX Team

Once you get through initial generative design for a set of screens and have at least initially settled on the most promising candidate design, you create a deck of slightly higher fidelity wireframes that represents where you think you are with that design, to summarize and clarify it for yourselves in the UX team. To simulate navigation, you can just say "click here" and move manually to the next wireframe on your laptop.

While these are a little higher fidelity than the design exploration sketches, these low-fidelity wireframes are still just static sketches of screens in that there is no interaction or navigation.

Sometimes it is convenient to use paper printouts of wireframes and move them around on the tabletop to show the design ideas to the UX team and others. During your early design reviews and walk-throughs, a prototype "executor" can manipulate the paper pages to respond to simulated user actions. If you show your stakeholders the prototype on paper (which we often do), UX team members can write redesign suggestions directly on the paper.

As an alternative, you can go directly to showing the screens on a laptop displayed for discussion via a projector.

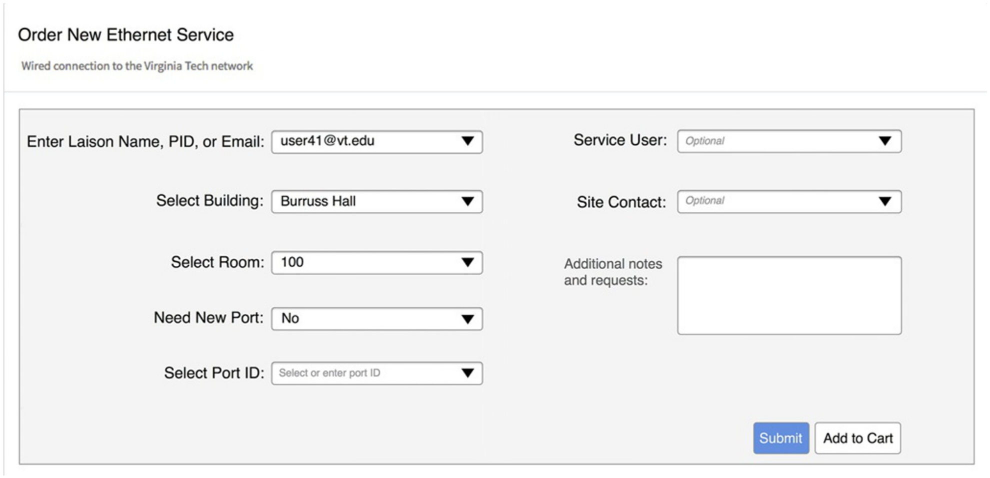
20.5.3.1 Lower fidelity means initial cost effectiveness

It is at the lowest end of the fidelity spectrum where you often get the most value from evaluation in terms of user experience gained per unit of effort expended. For finding and fixing most of the really obvious usability and UX problems early on, we believe this kind of paper wireframe prototype can be enormously effective. A low-fidelity prototype is much less evolved and therefore far less expensive. It can be constructed and iterated in a fraction of the time it takes to produce a good high-fidelity prototype. And yet, while there can be a big difference between a prototype and the finished product, low-fidelity prototypes can be surprisingly effective at finding UX problems in the design.

But, because of the low-fidelity prototype's simplicity and obvious lack of fidelity, many UX designers overlook its potential. The facility of wireframe prototypes enables you to create a design for a set of related user tasks, implement them in a low-fidelity wireflow prototype, evaluate with users, and modify the whole design—all within a day or so.

Example: Low-Fidelity Sketched Wireframes for Ordering Ethernet Service

Fig. 20-9 shows a low-fidelity sketched wireframe for a new design to order ethernet service, designed by the Network Infrastructure and Services UX team at Virginia Tech.


Order New Ethernet Service

Wired connection to the Virginia Tech network

Enter Laison Name, PID, or Email: user41@vt.edu

Select Building: Burruss Hall

Select Room: 100

Need New Port: No

Select Port ID: Select or enter port ID

Service User: Optional

Site Contact: Optional

Additional notes and requests:

Fig. 20-9

Low-fidelity sketched wireframe for ethernet service ordering (Thanks for permission to use this example to Joe Hutson and Mathew Mathai, Network Infrastructure and Services, Virginia Tech).

20.5.4 Increased Fidelity Wireframes for Subsequent Design Reviews and Walkthroughs

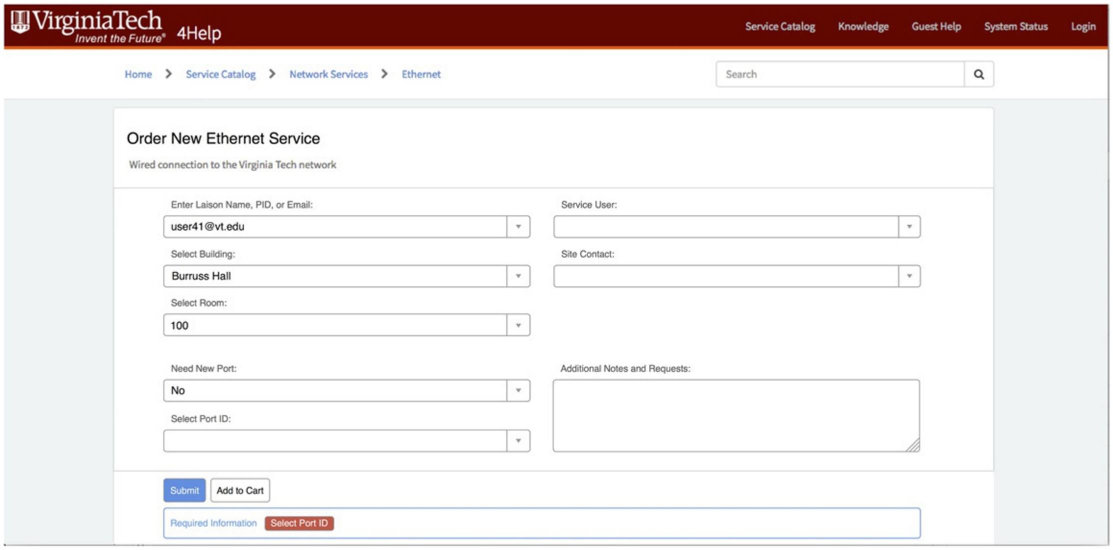
As you work with the design and go back to stakeholders with more refined versions, you will want to steadily increase the fidelity of the wireframes, which you can accomplish by adding more screens and more detail to each screen. You can add some color and some appearance of branding. Add increased fidelity by using a static image (e.g., JPEG) as a basic template containing background, color schemes, and styles in a fairly high-fidelity appearance.

Example: Increasing Fidelity of a Wireframe for Ordering Ethernet Service

The static image used in Fig. 20-10 is an example of an increased fidelity version of the same screen design as shown in Fig. 20-9, containing the background, colors, and styles that are starting to approach the desired look and feel. In this case, this design style template was obtained from people higher up in the organization who were responsible for making the official design appearance. We took a JPEG screenshot of that template, opened it in the Sketch app, and saved it as a background image to be used for all screens. We used the Sketch drawing tool to duplicate the colors, background, and styles in wireframe designs.

Fig. 20-10

Higher-fidelity wireframe for ethernet service ordering showing expected colors and style (Thanks for permission to use this example to Joe Hutson and Mathew Mathai, Network Infrastructure and Services, Virginia Tech).



Order New Ethernet Service

Wired connection to the Virginia Tech network

Enter Laison Name, PID, or Email: user41@vt.edu

Select Building: Burruss Hall

Select Room: 100

Need New Port: No

Select Port ID:

Service User:

Site Contact:

Additional Notes and Requests:

SN_21_HF

Work fast and efficiently. Reuse widgets by establishing a "library" of reusable forms and templates and predefined UI objects. You can even set yourself up with a library of particular styles—for example, Windows or Mac styles or your own branded style. In fact, you don't even have to use "real" widgets, just pick something that looks close from your library of UI objects.

The goal is for the client and users to see some of the UX design very quickly, usually by way of the design review and walkthrough evaluation techniques. Because of the low fidelity, UX people will have to do the "driving" to operate the prototypes. The wireframe identifiers are essential for "connecting" user actions on interaction objects to subsequent wireframes in the workflow. The UX team can annotate the paper wireframes with UX problems observed in the early rounds of evaluation. You can even hand sketch possible design solutions directly on the prototype and get feedback on those from the users.

Always trade off fidelity (when it is not needed) for efficiency (that is always needed). As an example, if it is not important to have the days and dates on your calendar correspond to the real-world calendar, you can even reuse the calendar grid with the dates already on it.

20.5.4.1 Establish a library of templates for interaction objects in your sketching tool

You can add efficiency as you build up the fidelity of look and feel through tools with libraries of interaction and UI styles and UI objects such as icons, buttons, menu bars, pull-down menus, pop ups, etc. Don't waste time by repeatedly building these interaction objects.

Reusing templates of design objects from a library can help you with consistency across designs and certainly increases your efficiency. Tools such as Sketch allow you to build libraries of such widgets, called symbols, and reuse them. Symbols can be nested and defined to specify aspects that can be overridden when instantiated. This allows designers, for example, to create a button symbol with a generic text label, and override that label every time that symbol is instantiated on the screen. For example, two instances of the same button symbol can have two different labels: "Save changes" and "Discard changes." Furthermore, updating a symbol will automatically update all instances of that symbol in the wireframe deck. For example, if the shape or color of that button symbol is updated, those changes are automatically propagated to all instances of that symbol.

Once you define your symbol library for your target platform, or choose to use one of the popular ones already built and available as a download, you can be very efficient and quick in generating detailed wireframe decks. Even large-scale changes to large wireframe decks can be made rather quickly because of the capabilities of modern tools such as Sketch.

20.5.5 Medium-Fidelity Wireframes with Some Navigational Behavior to Support Early Design Reviews and Walkthroughs

When you are ready for your earliest design reviews with the client, users, and other stakeholders, you connect these low-fidelity wireframes in a deck by way of some initial "hotspots," links or active buttons that allow sequencing through screens by clicking to demonstrate interaction flow by simulating navigational behavior. These prototypes do not usually have more functionality than that. With this added capability, these wireframes are called *click-through prototypes*, which are fast and easy to create and modify and, because they are machine-readable, they are easily sharable.

Click-through wireframe prototypes are used for evaluation via design reviews and walkthroughs with UX team and other stakeholders. This is really still just moving from one static visual wireframe to another but, because some links now work, the prototype can be operated on a laptop by anyone in the group, projecting it onto a screen or TV for group viewing and discussion.

The addition of links takes more effort to create, and even more to maintain as the deck changes, but the added realism is worth it. We have had users and clients exclaim that these are so realistic that they look just like the real thing.

These earliest design reviews usually result in going back to the UX studio, modifying the design, and returning to the same audience for confirmation and/or more feedback.

20.5.6 Medium- to High-Fidelity Click-Through Prototypes to Support Empirical Evaluation

Once you have the layout of each wireframe settled and have decided on the colors, with all the text, labels, boxes, widgets, and links in place, you might be ready to consider moving on from design review and walkthrough evaluations to empirical testing with real users who will do the driving and operate the prototype.

These days a user-based empirical evaluation is not something you always have time to do but, if you need to, you should have the right prototype to support it. To support user-based empirical evaluation or analytic evaluation, you may need to move to a medium- to high-fidelity programmed prototype with more detailed representations of designs, including details of appearance and interaction behavior and possibly even connections to system functionality. HTML5 and

CSS3 are common technologies for programmed prototypes and are often developed by a dedicated "prototyper" embedded in the UX team.

You can use prototypes containing scripts (written in scripting languages) to give a set of wireframes more ability to respond to user actions, such as links to pieces of real or simulated functionality. This added behavior is limited only by the power of the scripting language, but now you are getting into prototype programming, usually not a cost-effective path to follow very far.

Scripting languages can be relatively easy to learn and use and, being high-level languages, can be used to produce some kinds of behavior, mostly navigational behavior, very rapidly. But they are still not effective tools for implementing much functionality.

Preparing for an evaluation session with users will probably require elaborating all the states of the design relevant to the workflow that is the focus of the evaluation.

In projects requiring high rigor and where the stakes for getting it right are high, a high-fidelity prototype, although more expensive and time consuming, can lead to the insight you need; it is still less expensive and faster than building the final product. High-fidelity prototypes can also be useful as advance sales demos for marketing and even as demos for raising venture capital for the company. Aside from these possibilities, high-fidelity prototypes are beyond what you need in most projects, especially in agile development projects.

Horizontal prototype

A prototype incorporating very broad features, offering only shallow depth in its coverage of functionality (Section 20.2.1).

Benchmark task

A task description devised for a participant to perform during UX evaluation so that UX measures such as time on task and error rates can be obtained and compared to a baseline value across the performances of multiple participants (Section 22.6).

20.5.6.1 Include "decoy" user interface objects

If the goal is empirical evaluation with real users, you need to make sure the prototypes are representative of the real design. This means they have horizontal coverage. If you include only user interface objects needed to do your initial benchmark tasks, it may be unrealistically easy for users to do just those tasks. Doing user experience testing with this kind of initial interaction design does not give a good idea of the ease of use of the design compared to when it is complete. It also contains many user interface objects to choose from and many other choices to make during a task.

Therefore, you should include many other "decoy" buttons, menu choices, etc., even if they do not do anything—so that participants see more than just the "happy path" for their benchmark tasks. Your decoy objects should look plausible and should, as much as possible, anticipate other tasks and other paths. Users performing tasks with your prototype will be faced with a more realistic array of user interface objects about which they will have to think as they make choices about what user actions are next. When a user clicks on a decoy object, you get to use your "not implemented" message (next section). It's also an opportunity to

Participant

A participant, or user participant, is a user, potential, or user surrogate who helps evaluate UX designs for usability and user experience. These are the people who perform tasks and give feedback while we observe and measure. Because we wish to invite these volunteers to join our team and help us evaluate designs (i.e., we want them to participate), we use the term "participant" instead of "subject" (Section 21.1.3).

probe users on why they clicked on that object when it is not part of your envisioned task sequence.

20.5.6.2 Make a "this feature not yet implemented" message

This is the prototype's response to a user action that was not anticipated or that has not yet been included in the design. You will be surprised how often you may use this in user experience evaluation with early prototypes. See Fig. 20-11.

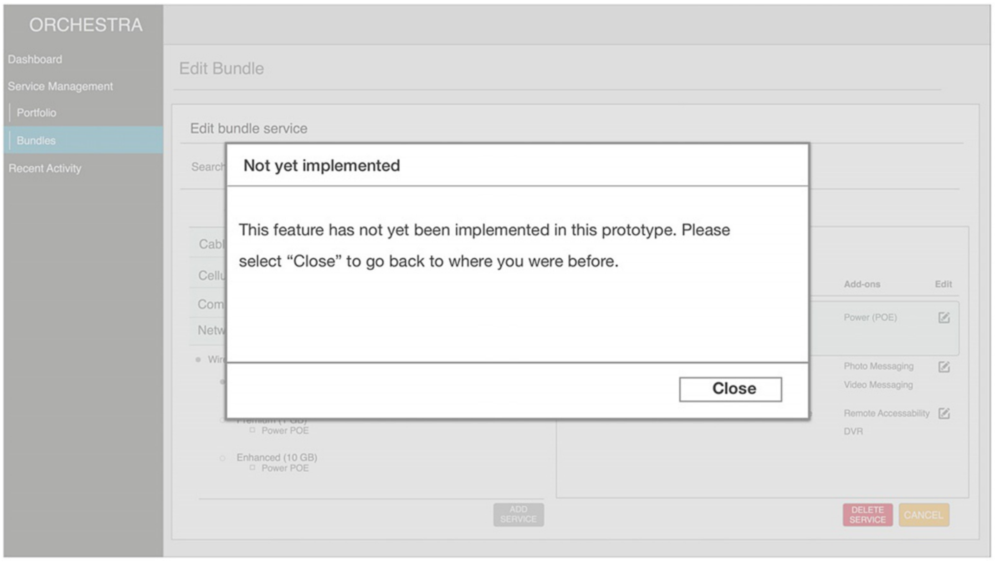
20.5.7 Medium- to High-Fidelity Prototypes Refined Through Evaluation and Iteration to Hand Off to Software Developers

Finally, after the design ideas are iterated and agreed upon by relevant stakeholders, wireframes (and in rare cases programmed prototypes) can be used as a part of interaction design specifications and for design production. Annotate them with details to describe the different states of the design and widgets, including mouse-over states, keyboard inputs, and active focus states. Edge cases and transition effects can now also be described. The goal here is completeness to enable a developer to implement the designs without the need for any interpretation.

If the developers don't have control over color and branding schemes, such UX design specifications can be accompanied by high-fidelity visual comps (pixel-perfect application "skins") from graphic designers (Section 12.5.5).

Fig. 20-11

"Not yet implemented" message (Thanks for permission to use this example to Joe Hutson and Mathew Mathai, Network Infrastructure and Services, Virginia Tech).



Not yet implemented

This feature has not yet been implemented in this prototype. Please select "Close" to go back to where you were before.

When you hand this prototype off to the software developers for implementation, the process requires collaboration across the UX-SE (software engineering) aisle. The UX people "own" the UX design but not the software that implements it. It is a little like owning a home but not the land it sits on. The "hand-off" point is a serious nexus in the two lifecycles (Chapter 29).

This is the time to go over it with a developer. This prototype:

- Acts as a kind of "contract" for what is to be built.
- Provides the mock-up for:
  - Discussing implementation feasibility.
  - Checking against software platform constraints.
  - Checking for consistency with other features already implemented.

Make it clear that you will be back to check the implemented version against this design.

20.5.7.1 Do not think the UX team is now done

Because preserving a hard-earned quality user experience in the design is not in the purview of the SE people, the UX people have a strong responsibility to ensure that their UX design comes through with fidelity. This checking of the implementation against the final design is something for which we borrowed the term quality assurance (QA), an essential part of the agile UX lifecycle in the late funnel. If the SE people are the sole interpreters of this process, there is no way to predict whether the user experience you worked so hard to achieve will still be there in the final product.

20.5.8 Visually High-Fidelity Prototypes to Support Graphic Design

It is usually not important for even high-fidelity wireframes to have exactly the right graphical appearance and exact colors at this point. Most of the time the UX team will already have worked out the colors and templates necessary for visual comps for the various styles used in the design. Wireframes usually don't have to be produced to those standards.

Visual comp

A pixel-perfect mockup of the graphical "skin," including objects, colors, sizes, shapes, fonts, spacing, and location, plus visual "assets" for user interface elements (Section 12.5.5).

Exercise 20-1: Building a Low- to Mid-Fidelity Wireframe Prototype Deck for Your System

Goal: To obtain experience with rapid construction of a low- to mid-fidelity wireframe prototype deck for some selected user tasks in your system.

Activities: This should be one of your most fun exercises, but it can also be quite a bit of work.

Make sure that the prototype will support at least the benchmark tasks (You might have to read ahead in Section 22.6 to learn a little bit about benchmark tasks for this).

Add in some other "decoy" interaction design "features," widgets, and objects so that the prototype does not look tailored to just your benchmark tasks.

Hints and cautions: It is normal for you to have to do more design work during this exercise to complete details that were not fully designed in previous exercises.

Remember: You are learning the process, not creating a perfect design or prototype.

Assuming you are doing this as a team: Get everyone on your team involved in drawing or using the wireframe sketching tool, not just one or two people. You will be done much faster if everyone pitches in.

If you are sketching the wireframes by hand, this is not art class so don't worry too much about straight lines, exact details, etc.

Pilot test to be sure it will support your benchmark tasks for evaluation.

Deliverables: A right, smart, "executable" wireframe prototype deck that will support your benchmark tasks in user experience testing.

Schedule: It could take several hours, but it is essential for the exercises that follow.

20.6 SPECIALIZED PROTOTYPES

In addition to the wireframe prototypes that are the bread and butter of today's UX design, for completeness we describe a few other kinds of prototypes to consider in specialized situations.

20.6.1 Physical Mockups for Physical Interactivity

A physical mockup is a tangible, three-dimensional prototype or model of a physical device or product, often one that can be held and often crafted rapidly out of materials at hand, used during exploration and evaluation to at least simulate physical interaction.

Physicality

Referring to real direct physical interaction with real physical (hardware) devices like in the grasping and moving of knobs and levers (Section 30.3.2.4).

If a primary characteristic of a product or system is physicality, such as you have with a handheld device, then an effective prototype will also offer the same kind of physicality in its interaction. Programming new applications on physical devices with real software means complex and lengthy implementation on a challenging hardware and software platform. Physical prototypes are an inexpensive way to afford designers and others insight into the product look and feel.

Some products are "physical" in the sense that they are a tangible device that users might hold in their hands. A physical prototype for such products goes beyond screen simulation on a computer, encompassing the whole device. Pering (2002) describes an older case study of such an approach for a handheld communicator device that combines the functionality of a PDA and a cellphone. If the product is to be handheld, make a prototype from cardboard, wood, or metal that can also be handheld.

Or a system might be "physical" like a kiosk. The TKS kiosk is an ideal candidate for physical prototyping. Brainstorm the physical design through ideation and sketches and then build some cardboard mockups that sit on the floor or the ground, add some physical buttons, and have a cutout for the screen about head height. After homing in on the overall look and feel with cardboard, you can make a wooden version to be sturdier, then add physical buttons and attach a touchscreen (e.g., an iPad or a detachable laptop touchscreen) from the inside to fill the cutout and allow some real interaction.

You can use materials at hand and/or craft physical prototypes with realistic hardware. Start off with glued-on shirt buttons and progress to real push-button switches. Scrounge up hardware buttons and other controls that are as close to those in your envisioned design as possible: push buttons, sliders (for example, from a light dimmer), knobs and dials, a rocker switch, or a joystick from an old Nintendo game.

Even if details are low fidelity, these prototypes are higher fidelity in some ways because they are typically three-dimensional, embodied, and tangible. You can touch them and manipulate them physically. If they are small, you can hold them in your hands. Also, physical prototypes are excellent media for supporting evaluation of emotional impact and other user experience characteristics beyond just usability.

Designers of the original Palm PDA carried around a block of wood as a physical prototype of the envisioned personal digital assistant. They used it to explore the physical feel and other requirements for such a device and its interaction possibilities (Moggridge, 2007, p. 204). Fig. 20-12 shows an example of a rough physical mockup of a design for a "rickshaw"-style cart for transporting people in developing countries.

Physical prototyping is now being used for cellphones, consumer electronics, and products beyond interactive electronics, employing found objects, "junk" (paper plates, pipe cleaners, and other playful materials) from the recycle bin, thrift stores, dollar stores, and school supply shops (Frishberg, 2006). Perhaps IDEO[4] is the company most famous for its physical prototyping for product

Embodied interaction

Interaction with technology that involves a user's body in a natural and significant way, such as by using gestures (Section 6.2.6.3).

Tangible interaction

Interaction involving physical actions between human users and physical objects. A key area of focus in Industrial design, pertaining to designing objects and products to be held, felt, and manipulated by humans. Closely related to embodied interaction (Section 6.2.6.3).

Emotional impact

An affective component of user experience that influences user feelings. Includes such effects as enjoyment, pleasure, fun, satisfaction, aesthetics, coolness, engagement, and novelty and can involve deeper emotional factors such as self-expression, self-identity, a feeling of contribution to the world, and pride of ownership (Section 1.4.4).

[4]http://www.ideo.com

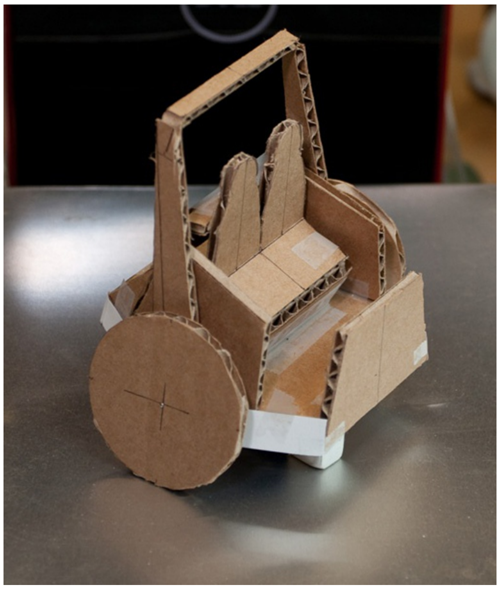


Fig. 20-12

Example of a rough physical mockup (courtesy of Akshay Sharma, Virginia Tech Department of Industrial Design).

ideation; see their shopping cart project video (ABC News Nightline, 1999) for a good example.

Wright (2005) describes the power of a physical mockup that users can see and hold as a real object over just pictures on a screen, however powerful and fancy the screen graphics. Users get a real feeling that this *is* the product. The kind of embodied user experience projected by this approach can lead to a product that generates user surprise and delight, product praise in the media, and must-have cachet in the market.

Heller and Borchers (2012) created a physical mechanism for individual electric consumers to be visually aware of their levels of electric consumption at any moment as a simple but compelling example of a physical mockup. By integrating a display where the power is consumed, right at the household outlet, they fashioned a display that is there all the time, not something the consumer has to choose to use or must attach.

Iterating through a series of software and hardware prototypes, the final step was to add a DIY hardware design printed circuit board behind the outlet, making it a fully functional model. The amount of power being consumed at any moment is displayed very simply by a band of color surrounding the outlet socket—green for low consumption, yellow for medium, and red for high.

See more about physical mockups as embodied sketches (physical embodiments of design ideas) in Section 14.3.3.

20.6.2 Paper-in-Device Mockup Prototype, Especially for Mobile Applications

Prototypes can be very effective for apps on mobile devices, but a paper prototype of a mobile app needs an "executor," a person playing computer to change screens and do all the other actions of the system in response to a user's actions. This role of mediator between user and device will necessarily interfere with the usage experience, especially when a large part of that experience involves holding, feeling, and manipulating the device itself.

Bolchini, Pulido, and Faiola (2009) and others devised a solution by which they placed the paper prototype inside the device, leveraging the advantages of paper prototyping in evaluating mobile device interfaces with the real physical device. They drew the prototype screens on paper, scanned them, and loaded them into the device as a sequence of digital images that the device can display. During evaluation, users can move through this sequential navigation by making touches or gestures that the device can already recognize.

(This kind of realism can be achieved without needing a sequential series of screens by click-through wireframe prototypes where hotspots on the screen take you to different screens. The prototype is then loaded as a pdf file into a mobile device.)

This is an agile and inexpensive technique, and the authors reported that their testing showed that even this limited amount of interactivity generated useful feedback and discussion with evaluation users. Also, by adding page annotations about user interactions, possible user thoughts, and other behind-the-scenes information, the progression of pages can become a design storyboard.

Storyboard

A visual scenario in the form of a series of sketches or graphical clips, often annotated, in cartoon-like frames, illustrating the interplay between a user and an envisioned ecology or device (Section 17.4.1).

20.6.3 Animated Prototypes

Most prototypes are static in that they depend on user interaction to show what they can do. Video animation can bring a prototype to life for concept demos, to visualize new UX designs, and to communicate design ideas. While animated prototypes are not interactive, they are at least active. *An animated prototype is a prototype in which the interaction objects are brought to life via animation, usually in video, to demonstrate dynamically and visually what the interaction looks like.*

Löwgren (2004) shows how video animations based on a series of sketches can carry the advantages of low-fidelity prototypes to new dimensions where a static paper prototype cannot tread. Animated sketches are still "rough" enough to invite engagement and design suggestions but, being more like scenarios or storyboards, animations can convey flow and sequencing better in the context of usage.

HCI designers have been using video to bring prototypes to life as early as the 1980s (Vertelney, 1989). A simple approach is to use storyboard frames in a flip book-style sequence on video or, if you already have a fairly complete low-fidelity prototype, you can film it in motion by making a kind of "claymation" frame-by-frame video of its parts moving within an interaction task.

20.6.4 Experience Prototyping, the Goal of High-Fidelity Physical Prototyping

As Buchenau and Suri (2000) point out, if you are told something, you forget it. If you see something for yourself, you remember it. But if you do something for yourself, you understand it.

For some domains, then, for participants to understand the design situation and context well enough to give effective feedback, the prototype they use must allow them to actually do the activity being designed for, to become engaged and immersed in the subjective experience. To appreciate what users will feel, participants need to get beyond being passively exposed to a demo or walkthrough of a prototype and become actively engaged (Buchenau & Suri, 2000, p. 425).

A very good example that everyone can understand is a full flight simulator for a specific airplane. It's not enough to just look at screens and have someone tell you what would be happening. The pilot in training must experience flight situations in as close a way as possible to the real thing.

But aircraft flight simulators are a special case that can be almost as complex as the aircraft themselves. Buchenau and Suri are talking about experience prototypes that succeed in domains not nearly as expensive or as complex. For example, in a project to design for an Internet-enabled cardiac telemetry system that included a device to deliver a defibrillating shock to heart patients in the field (Buchenau & Suri, 2000, p. 426), participants had to simulate usage of several device designs set in the usage context to give full contextual feedback.

In this case, an experience prototype was used in support of usage research. Participants were given a pager to carry with them on weekends. Getting a page on this device represented a patient getting a rather large electrical shock, with the aim of stopping fibrillation that had been detected by the remote device. They were also given a camera to photograph their immediate surroundings at the time of the "shock" and a notebook in which to describe the experience, what were they doing, and what it would have been like to be stunned by a real defibrillating shock at that exact moment in time.

Participants quickly understood the necessity for getting a warning before such a shock is administered—if only for safety (in case they were holding a baby or operating a power tool, for example) and to prepare themselves psychologically. Plus, a way was needed to explain the patient's condition to bystanders. In this approach, "high fidelity" means bringing the participant close to the experience of the real thing.

20.6.5 "Wizard of Oz" Prototypes

Even though we rarely see this type of prototyping in practice these days, we cover it here for completeness. The Wizard of Oz prototyping technique is a deceptively simple approach that gives the *appearance* of a high degree of interactivity. It can be a rapid way to produce highly flexible prototype behavior in complex situations where user inputs are unpredictable. The setup requires two connected computers, each in a different room. The user's computer is connected as a "slave" to the evaluator's computer. The user makes input actions on one computer, which are sent directly to a human team member at the evaluator's computer, hidden in the second room. The human evaluator sees the user inputs on the hidden computer and sends appropriate simulated output back to the user's computer.

This approach has the advantage of an apparently high level of interactivity as seen by the user. It is especially effective when flexible and adaptive "computer" behavior is of the essence, as with artificial intelligence and other difficult-to-implement systems. Within the limits of the cleverness of the human evaluator, the "system" should never break down or crash.

In one of the earliest uses of the Wizard of Oz technique that we know of, Good, Whiteside, Wixon, and Jones (1984) designed empirically a command-driven email interface to accommodate natural novice user actions. Users were given no menus, help, documentation, or instruction.

Users were unaware that a hidden operator was intercepting commands when the system itself could not interpret the input. The design was modified iteratively so that it would have recognized and responded to previously intercepted inputs.

The design progressed from recognizing only 7% of inputs to recognizing about 76% of user commands.

The Wizard of Oz prototyping technique is especially useful when your design ideas are still wide open and you want to see how users behave naturally in the course of simulated interaction. It could work well, for example, with a kiosk.

You would set up the general scope of usage expected and let users at it. You will see what they want to do. Because you have a human at the other end, you do not have to worry about whether you programmed the application to handle any given situation.

20.7 SOFTWARE TOOLS FOR MAKING WIREFRAMES

Wireframes can be sketched using any drawing or word processing software package that supports creating and manipulating shapes. While many applications suffice for simple wireframing, we recommend tools designed specifically for this purpose. We use Sketch, a drawing app, to do all the drawing. Craft is a plug-in to Sketch that connects it to InVision, allowing you to export Sketch screen designs to InVision to incorporate hotspots as working links.

In the “Build mode” of InVision, you work on one screen at a time, adding rectangular overlays that are the hotspots. For each hotspot, you specify what other screen you go to when someone clicks on that hotspot in “Preview mode.” You get a nice bonus using InVision: In the “operate” mode, you, or the user, can click anywhere in an open space in the prototype and it highlights all the available links. These tools are available only on Mac computers, but similar tools are available under Windows.

Beyond this discussion, it’s not wise to try to cover software tools for making prototypes in this kind of textbook. The field is changing fast and whatever we could say here would be out of date by the time you read this. Plus, it wouldn’t be fair to the numerous other perfectly good tools that didn’t get cited. To get the latest on software tools for prototyping, it’s better to ask an experienced UX professional or to do your research online.