User Experience Direct
(UX Direct)

FAQ: How to Prototype the User Interface

Disclaimer

The following is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, functionality, or service and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described for Oracle’s products remains at the sole discretion of Oracle. This document contains preliminary images.
FAQ: How to Prototype the User Interface

User Experience Direct (UX Direct) is an Oracle Applications User Experience (UX) program that provides user experience expertise to Oracle customers and partners for their implementations, customizations, and usage of Oracle enterprise applications. The goal of this program is to enhance end user experiences during and after customer implementations and improve user adoption of Oracle’s enterprise applications.

Overview

This UX Direct collateral describes user interface prototypes: what they are, their benefits, tools you can use to create them, and an example of a low-fidelity prototype. It also integrates prototyping into a user-centered design process for application software implementation. The audiences for this collateral are the customer team responsible for customizing an Oracle application and consultants who are hired to assist in implementations.

1. What is a prototype?

A prototype is a mock up or a simplified representation of an object. For example, a prototype of an airplane design is often used to test its aerodynamics before building the real plane. Software user-interface prototyping, refers to the activity of creating prototypes of software applications, i.e., incomplete versions of the pages displayed by an application being developed or modified. A prototype typically simulates only a few aspects of the final solution. For example, it may only show the home page of an application with only the navigation links active.

2. Why create a prototype?

A question sometimes asked about prototypes is: Why create a prototype that is then thrown away when you can code a solution, adjust it, and then you already have the released product? The answer is that moving to code too early results in products that need to undergo costly re-coding and results in difficult to learn and use products. Before coding takes place developers need to gather and understand the users’ requirements, create user profiles, determine the critical and frequent tasks users perform, and use prototypes to obtain feedback from users on design solutions. That user-centered design process is how Oracle develops its core applications. The UX Direct program works with customers to apply those same practices to the customization of applications.

Prototyping has the following benefits:
- It saves time and money. It allows the implementation team to explore a design early in its evolution. For example, the team analyzing their end users’ work process may propose that the process can be implemented with a wizard. But there are design issues to be resolved, such as how many steps are needed, how to distribute the work over the steps, and how to make the process clear to users. A prototype of the wizard can be created in a few hours and shown to end users for their feedback long before the application is coded.
- It requires that the team work out the steps in the design so that there is no uncertainty about what it will do and whether it meets any stated requirements.
- It makes it easy to obtain user feedback on the design. The prototyped pages can be shown to end users who are asked to complete one or more tasks with them. User’s hesitations and errors help to identify usability problems. Because usability problems are identified based on data from users rather than opinions debated among team members, the need to fix them is much clearer.
- It makes it easy and quick to make changes. Because a prototype is not the application code, designers are less likely to resist making changes to their designs. If end users are confused, the mocked-up screens can be changed in a few minutes and shown to them again or to different users to verify that the solution fixed the problems.
- Its cost in dollars and labor is low. As we will see below, paper prototypes or low-fidelity software prototyping tools are cheap, easy to learn, and can be used to quickly mockup work processes. When you compare the cost of prototyping early in development against the cost of modifying code, the savings are substantial.

3. How close is a prototype to the final product?

It depends. Prototypes differ from the final product in two ways: their fidelity and their scope. Fidelity has to do with how closely the prototype looks like the final product. It is convenient to think about prototype fidelity as having two levels:

1. Low fidelity prototypes may look very different from a finished product. An example of a low fidelity prototype is a paper prototype – see the following figure. Low-fidelity representations, such as sketches, differ from the final product in interaction style, visual appearance, and/or level of detail.
Paper prototypes are used in the early stages of development to explore user interface concepts. They are easy and quick to create and to modify.

As the figure shows, a paper prototype can be interactive in the sense that a user can indicate what actions they would take on a screen and the paper version can be used to show the results. A fuller description of paper prototyping can be found at [http://www.paperprototyping.com/index.html](http://www.paperprototyping.com/index.html). As we will see below, you also can create low-fidelity prototypes using software tools.

2. High-fidelity prototypes look close to the finished product and are highly interactive. While there may be little behind them, they look and act like a real interface. They include some visual design and a high level of detail. They always are created with software tools. High-fidelity prototypes typically are used to evaluate a small part of an application that is nearing release.

Prototypes also differ in their scope. Some common types are:

- Horizontal prototypes provide a broad view of an application or website, focusing on user interaction more than low-level detail. They can be useful for probing the effectiveness of navigation through work processes and the overall structure of a design.
- Vertical prototypes are more complete elaborations of a single process. For example, a designer might prototype only a wizard or other guided step-by-step process.

The two dimensions of a prototype then determine its type. A low fidelity, horizontal prototype could be constructed with a simple-to-use software tool to mimic the navigation flow of pages used in accomplishing a business process. Only the navigation links or buttons on the pages need be active. A low-fidelity vertical prototype could also be constructed with a simple-to-use software tool and show the screens used to accomplish a task using a wizard. All the necessary fields and links could be active, but the prototype would not go beyond the scope of the wizard.

For prototyping screens during the implementation of an Oracle application, it is not necessary to use high-fidelity prototypes. Consequently, we will focus in this collateral on the low-fidelity ones.

4 What tools are available for low fidelity prototyping?

In addition to paper prototyping, there are perhaps a dozen software tools that have been used for prototyping. But most of them were not designed for creating prototypes; they have been adopted for that purpose. Table 1 shows the tools that a sample of Oracle User Experience (UX) designers have used and recommended for low-fidelity prototyping. It describes the advantages and limitations of each tool.

The first item in the list is paper prototyping. It requires only a minimal ability to draw and very little practice to create prototypes with some interactivity. Paper prototyping is the tool that Oracle UX designers use most frequently.

The second tool in the list is Balsamiq. It is by far the most recommended software tool for low-fidelity prototyping. It was invented to make it easy to create low-fidelity prototypes. It can be learned in a few hours and, with a little practice, a mockup of a page can be created in 10-15 minutes. A license costs about $80. In the next section we will show an example of a prototype created with Balsamiq.

The other two tools described in the table have the advantage of being readily available in most companies. Because of their familiarity, many people know how to use them. But they were not designed for creating prototypes so their limited flexibility makes it harder to simulate interactive software pages.

There are some other tools available that are sometimes used for prototyping: Axure, Dreamweaver, Fireworks, Flash, Flex, Illustrator, Microsoft Expression Blend, Microsoft Expression Design, OmniGraffle, Photoshop, and Thermo. However, these tools were not built for creating low-fidelity prototypes. Most were made for creating polished looking, detailed images. Consequently, they work best for high-fidelity prototyping.
<table>
<thead>
<tr>
<th>Tool Name</th>
<th>URL</th>
<th>Advantages</th>
<th>Limitations</th>
</tr>
</thead>
</table>
| Paper Prototyping         | www.paperprototyping.com/       | • Usability problems detected at a very early stage  
• Minimal resources and expertise required  
• Quick to build and refine, supporting iterative design  
• Stimulates team creativity  
• Making changes faster than with computer-based prototypes | • Only limited interactions can be simulated  
• Evaluation of detailed design elements not supported  
• Cannot reliably simulate system response times, error rates, etc  
• Rapid navigation between screens and pop ups difficult |
| Balsamiq                  | www.balsamiq.com                | • Tool built for creating low-fidelity prototypes  
• A wide range of controls that look hand drawn yet allowing functions like copy/paste, undo/redo.  
• Mastered in a few hours  
• Pages designed in minutes  
• Designs edited in real time during the course of a team meeting | • Detailed designs to achieve a higher level of fidelity not supported  
• Cannot reliably simulate system response times, error rates, etc. |
• Pages linked together and exported as web pages  
• Allows creating a background and foreground  
• Available within most organizations | • Not designed for creating low-fidelity prototypes resulting in functionality limitations  
• Only available on Windows platform  
• Many hours to become proficient enough for creating prototypes |
| Microsoft PowerPoint      | http://office.microsoft.com/en-us/powerpoint/ | • Free tool for Windows users  
• Many people are familiar with it  
• Basic functions are easy to learn | • Not designed for creating low-fidelity prototypes resulting in functionality limitations  
• Functions needed for quick prototyping are difficult to learn  
• Only available on Windows platform |
5 How do I create a low-fidelity prototype?

We have created an example of a prototype using Balsamiq to illustrate the prototyping process. But before we discuss it, it is important to understand the context into which a prototype fits. Before you can decide what to prototype you must first understand the relevant business processes of the customer, who the end users are, their requirements, and the critical and frequent tasks they perform within business processes. All of these elements are included a user profile (see the UX Direct collateral User Profile Checklist).

The User Profile describes the skills and experience of end users, the business processes they work within and their critical and frequent tasks. Those tasks become candidates for prototypes. What do end users spend 80% of their time doing? Which tasks are key?

The hypothetical application we have chosen to illustrate the prototyping process allows potential employees to apply for open jobs within ABC corporation. We chose this simple application because it does not require specialized knowledge to understand the task. We don’t want the details of a complex task to get in the way of understanding the prototype and how it can be used to create a mockup of a simple task.

Let’s look at the first page of our mockup.

It shows a page in the Careers section of a hypothetical application to apply for a job. This page was created in about 10 minutes. Notice that it looks like a hand-drawn page, which is the look that Balsamiq purposely uses. It’s meant to look like a mockup not a polished, finished page. The mockup illustrates a design concept. Balsamiq pages are not interactive in the sense that you can click on a button and automatically move to the appropriate page. But end users can indicate what they would do on the page and they are then shown the next one.

This first page shows the user the scope of the application and allows them to either search for an open job or to pick from a list of new job openings. It’s quite complex in what it simulates. There are vertical and horizontal scroll bars. A left-hand navigation pane that can be expanded or contracted. It has text entry areas and text display areas. It has buttons and links. The text in the text areas can be pasted from a text file you have created and can look like text on a screen. The text areas on this page have explanatory text that says what would appear. But you could display realistic text if you wanted to simulate that.

By showing this page to users, you can ask them not only what they would do to move forward to apply for a job, but also if they understand the labels of buttons and links and the names of page sections. For example, the term Apply beside the job description. Apply is commonly used in applications to mean take an action but leave me on this page instead of what it’s meant to mean here “apply for this job.” Is a better term possible? Users might be asked what they would expect to see if they selected My Job Basket and whether that function would be of value to them.

Let’s assume that the user wanted to look at a new job description. He or she would click on the Apply button beside Job Description 1. The following page would be displayed.

Again, it has a hand-drawn look. There are links at the top. On this page, the user could navigate to the previous page or to other pages. Users might be asked if the ability to email the
description would be needed and, if so, whether the “to Friend” is needed on the Email button.

The blank job description field could show real text or it could indicate the components that make up a description.

Because the task asks the user to apply for a job, the user would choose Apply Now button and see the following page.

From the designers point of view, this page is the beginning of a sequence of pages, like a wizard or a train of pages. The question is whether users will see it that way. There is a checkbox and, at the top, the first item of a breadcrumb trail. The presence of the Next button should indicate to users that this is the beginning of a process but that needs to be confirmed by users. The user would agree to the terms and select the Next button.

We have created this page to illustrate several ways that users can be asked to answer questions: a spin box, checkboxes, and radio buttons. Balsamiq provides a list of widgets to select from. If the wording of questions was an issue to explore, the text of a question and its options could be displayed so that end users could comment on them.

Another issue on this page is what happens if the user decides to exit. Will their resume be saved? Whatever that decision is, there should be a dialogue asking users if they really want to exit and, perhaps, if they want their resume keep on file. The design of that dialogue could be simulated to obtain users’ feedback on it.

Again, the breadcrumb trail has an additional item. To move on, the user would select the Next button.
On this page we illustrate the use of tabs and fields to which text can be added. Note that the inactive tabs are grayed out. Users could be asked if they need more guidance about what goes into the text boxes and whether items are required or optional.

To move on, the user would select the Next button.

This page displays a status section showing the parts of the application that have been completed. Note also that we have added a question mark Help icon. The Help icon can be placed anywhere on any of the pages and users can be asked what they would expect the Help to tell them. On this page the user can go back to the previous page to add Education Information. Also by watching users and listening as they talk out loud, it can be determined if they see the View as PDF link below the status box. To move on, the user would select the Next button.

This is the last step in the sequence. By asking users what they could do on this page, you could determine if they realize that this is the end. There could be a Finish button here, then a submit button on the next page. But we saved a step by allowing the user to submit the application on this one. This is an example of a dual action button that saves the application and submits it. The button could say Save and Submit Application if users have any doubt about whether their application has been saved. To move on, the user would select that button.

This page shows an information dialogue that pops up over the previous page. By selecting Back to Home, the user moves
back to the page that started the process. Users might want more places to jump to from here. In addition, they might require that they receive some indication about when they would hear that their application has been accepted. To end our example task, the user selects Exit.

This ends our example of a walkthrough of an typical task for a typical user of this hypothetical application. The example illustrates how a prototype focuses on a user profile and a specific task. It shows that many alternative design elements can go into even a simple task and how obtaining user feedback can indicate which alternatives work best. With a low-fidelity prototyping tool, alternatives can be explored quickly and little is lost if a design solution needs to be discarded.

6. How do we evaluate a prototype?

As we mentioned above, one of the main advantages of a prototype is that it allows an implementation team to show their design to end users to evaluate it. The evaluation is called usability testing, which we have described in a separate UX Direct collateral: *Usability Testing Checklist*.

In the example walkthrough we illustrated in the previous section, there were numerous design alternatives that were explored by having users attempt the task and to comment as they worked on their understanding of what they were seeing. That is how a usability test works and why it is so valuable.

Briefly, the testing process for a prototype would follow these steps:

- The image of the prototype pages could be captured and pasted into a file to show users or the images could be printed out.
- The task that the prototype was based on would be turned into a scenario for the user to attempt “You want to apply for a job with ABC corporation. Use these pages to complete the application process.”
- The test administrator would find a small sample of 3-5 end users.
- The administrator would then ask each user individually to attempt the task with the pages while talking out loud as they work. The users would point at the page to indicate what action they would take. Then the administrator would move to the appropriate page. In cases for which there is more than one design alternative, users could be shown both and asked to say which one they prefer and why.
- The administrator would note where users hesitate, make mistakes, or seem confused. The administrator would probe to understand what problems in the design are causing the behaviors.
- The team would then design a fix for the problems and create a new mockup in a few minutes.
- The administrator would then either go back to the previous users or ask a new sample to attempt the task with the new pages to confirm that the problems have been eliminated.

As you can see, prototyping makes the iterative evaluation of a design easy to accomplish.

7. Are prototypes necessary or just nice to have?

By its nature, prototyping changes the way software applications are designed. Creating and evaluating application page designs with a prototype is a collaborative process. Designers work together to mock up pages and then end users get involved by attempting tasks with those mockups. The more the designers know about who the users are, what their wants and needs are, and what key tasks they perform within a business process, the better the mockups are to start with. Bringing the end users in early to evaluate those mockups then improves the design further.

Prototyping is an essential step in a user-centered development process because it is the vehicle through which users get involved in evaluating the user-interface design. It is the key to catching design flaws early, long before coding begins and changes become costly.