UX Prototypes: Low Fidelity vs. High Fidelity

Kara Pernice
December 18, 2016

Summary: Clickable or static? Axure or paper? No matter which prototyping tools you use, the same tips apply to preparing a user interface prototype for the most effective user research.

In This Article:

What Is a Testable Prototype?

A user interface prototype is a hypothesis — a candidate design solution that you consider for a specific design problem. The most straightforward way to test this hypothesis is to watch users work with it.

There are many types of prototypes, ranging anywhere between any of these pairs of extremes:

- Single page vs. multipage with enough menus, screens, and click targets that the user can completely finish a task
- Realistic and detailed vs. hand-sketched on a piece of paper
- Interactive (clickable) vs. static (requiring a person to manipulate different pages and act as a “computer”)

The choice of prototype will vary greatly depending on goals of the testing, completeness of the design, tools used to create the prototype, and resources available to help before and during the usability tests. But, whatever prototype you may use, testing it will help you learn about users' interactions and reactions, so you can improve the design.

Why Test a Prototype?

Ripping up code is very expensive. Ripping up a prototype is not, especially if it's just a piece of paper.

Let's first consider common arguments for **not** testing a prototype. These are:

- Waiting for a design to be implemented before you test it means that the design actually works, and test participants can use it in a natural way.
- There is no adjustment to be made to the Agile or Waterfall processes to accommodate UX and iterative design.
- Some Lean proponents say that, with no prototype testing, there is no prototype to throw away when it (inevitably) tests badly, so there is no waste.

These arguments may seem good at first glance. But in reality, **testing final products is uninformed and risky**. Enlightened teams create prototypes, have users test them, and iterate the design until it's good enough. (Note: We also test final products to benchmark the usability at the end of a project or at the beginning of a new one, to assess ROI, to run competitive studies, and to do a final check and make small tweaks.)

Interactive vs. Static Prototypes

Work needs to be done to bring a prototype to life for usability testing. To make it respond to user actions, we can spend time implementing the interaction before the test or we can "fake" the interaction during the test. Both methods have benefits and drawbacks.

Interactive (Clickable) Prototypes

With interactive prototypes, the designer must set a response for each possible user action before the test happens. Even with the right tool, building an interactive prototype can be time consuming: you have to get all the click targets right, and make the system respond only when the user interacts with a clickable target.

Static Prototypes

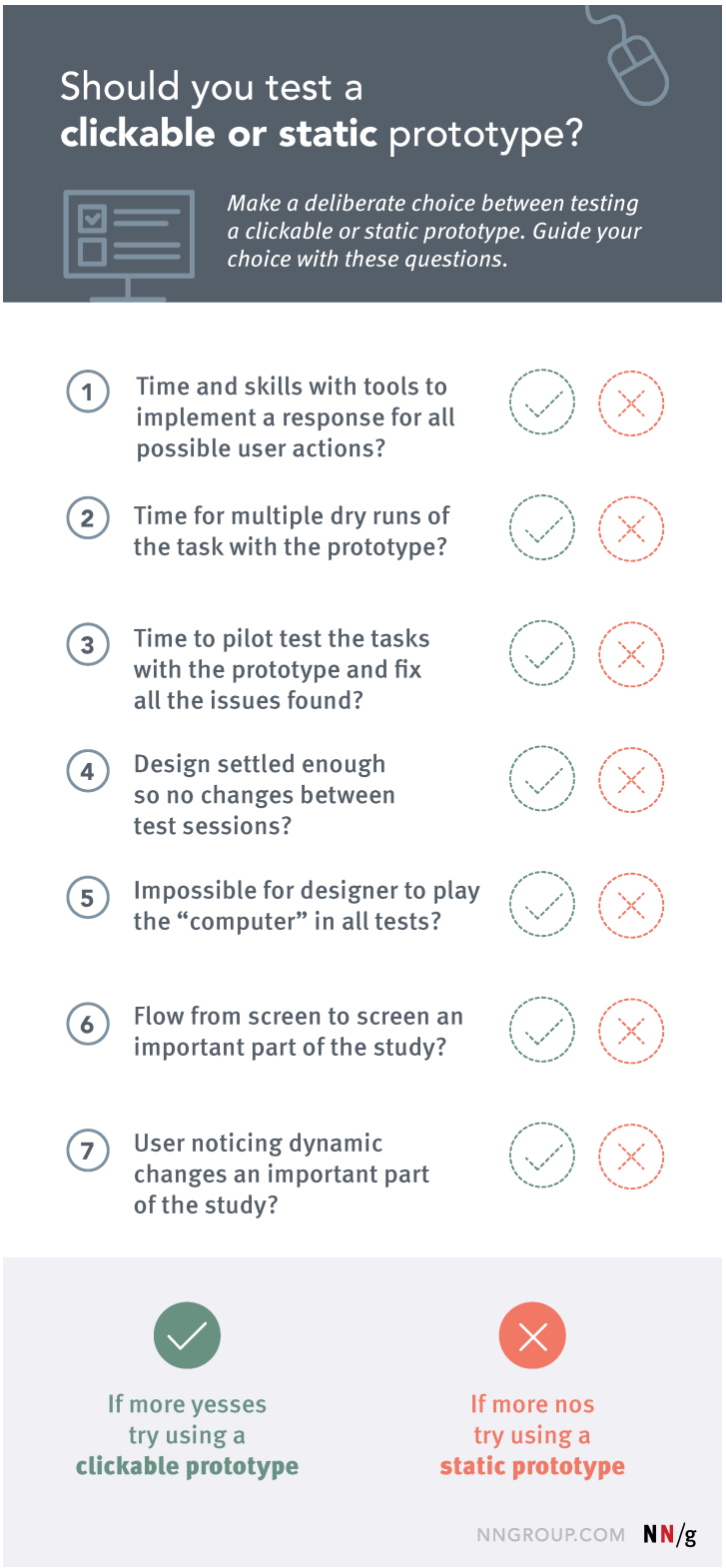
With static prototypes, the responses to users' actions are given in real-time during the test by a person who is familiar with the design. There are several methods that can be used with static prototypes:

- **Wizard of Oz.** This method is named after the famous Frank Baum book (and more famous movie) with the same name. (If you're not familiar with the book or the movie: in it, the great Wizard of Oz is impersonated by an ordinary human hiding behind a curtain.) The "wizard" (the designer or someone else familiar with the design) controls the user's screen remotely from another room. None of the user's clicks or taps really do anything. Rather, when the user clicks, the "wizard" decides what should come next, then serves up that page to the user's screen. The "wizard" may even create the page on-the-fly and serve it up. Users don't know what produces the response. In fact, they are usually told very little beyond that the system is "unfinished and slow."
  Wizard of Oz testing is particularly useful for testing AI-based systems before you have implemented the artificial intelligence. The human who controls the computer can simulate the AI responses based on natural intelligence.

- **Paper-Prototype "Computer."** The design is created on paper. A person who knows the design well plays the part of the "computer" and lays the papers out on a table, near the user's test table but not in her line of sight. As the user taps with the finger on a paper "screen" in front of her, the "computer" picks up the page (or modular part) representing the response and places it in front of the user. (In this article, we use the notation of "computer" to refer to the human who's implementing the user interface during the test session.)
  TIPS:

  1. The "computer" should indicate to the users when "it" has finished working and they can proceed with the interaction. This can be done either by using a designated gesture consistently (e.g., hands folded in front of you) or by using a special "Working" or hourglass-icon printout that is shown to the users while the "computer" is looking for the appropriate response and that is removed as soon as the "computer" has finished working.
  2. The facilitator should avoid overexplaining the design elements or the process.
- **Steal-the-Mouse "Computer."** This method is a version of the Wizard of Oz technique in which the "wizard" is in the same room with the user. (The "wizard's" role could be played by the facilitator.) The prototype is shown to the user on a computer screen. As the user clicks, the facilitator asks the user to look away from the monitor for a moment and the "wizard" navigates to the page that should appear next. The user is then prompted to look at the monitor and continue.

Criteria to help you decide which type of prototype is right for your project:



The *fidelity* of the prototype refers to *how closely it matches* the look-and-feel of the final system. Fidelity can vary in the areas of:

- Interactivity
- Visuals

- Content and commands

A prototype may have high or low fidelity in all or some of the above 3 areas. The table below explains what high and low fidelity mean in each of these areas.

	HIGH-FIDELITY PROTOTYPE	**LOW-FIDELITY PROTOTYPE**
Interactivity		
Clickable links and menus	Yes: Many or all are clickable.	No: Targets do not work.
Automatic response to user's actions	Yes: Links in the prototype are made to work via a prototyping tool (e.g., InVision, PowerPoint).	No: Screens are presented to the user in real time by a person playing "the computer."
Visuals		
Realistic visual hierarchy, priority of screen elements, and screen size	Yes: Graphics, spacing, and layout look like a live system would look (even if the prototype is presented on paper).	No: Only some or none of the visual attributes of the final live system are captured (e.g., a black-and-white sketch or wireframe, schematic representation of images and graphics, single sheet of paper for several screenfuls of information). Spacing and element prioritization may or may not be preserved.
Content and Navigation Hierarchy		

Content	Yes: The prototype includes all the content that would appear in the final design (e.g., full articles, product-description text and images).	No: The prototype includes only a summary of the content or a stand-in for product images.

Benefits of High-Fidelity Prototypes

1. **Prototypes with high-fidelity interactivity have realistic (faster) system response during the test.** Sometimes it can take extra time for the person playing the computer, whether online or on paper, to find the right screen and respond to a user's click. Too long of a lag between user's action and the "computer's" response can break the users' flow and make them forget what they did last or expected to see next.
   A delay also gives users extra time to study the current page. So, with a slow prototype, usability-test participants may notice more design details or absorb more content than they normally would with a live system.

   TIP: If the page supposed to appear next is hard to find in a paper prototype or slow to load in a clickable prototype, take away the current screen the user is looking at, so she is instead looking at a blank page or area. When the next page is ready, first display the previous page for a few moments again so the user can get her bearings, then replace that screen with the next one. The test facilitator can help this process by saying just a few words to help the user recover the context — for example, "Just a recap, you clicked *About Us."*

2. **With high-fidelity interactivity and/or visuals, you can test workflow, specific UI components (e.g. mega menus, accordions), graphical elements** such as affordance, page hierarchy, type legibility, image quality, as well as engagement.
3. **High-fidelity prototypes often look like "live" software to users.** This means that test participants will be more likely to behave realistically, as if they were interacting with a real system, whereas with a sketchy prototype they may have unclear expectations about what is supposed to work and what isn't. (Though it's amazing how strong the suspension of disbelief is for many users in test situations where not everything is real.)
4. **High-fidelity interactivity frees the designer to focus on observing the test instead of thinking about what should come next.** Nobody needs to worry during the test about making the prototype work.

5. **Interactive-prototype testing is less likely to be affected by human error.** With a static prototype, there is a lot of pressure on the "computer" and a fair chance of making a mistake. Rushing, stress, nerves, paying close attention to user clicks, and navigating through a stack of papers can all make the "computer" panic or just slip during the test.

Benefits of Low-Fidelity Prototypes

1. **Less time to prepare a static prototype, more time to work on design, before the test.** Creating a clickable prototype takes time. Without having to make the prototype work, you can spend more time on designing more pages, menus, or content. (You still need to organize pages before the test so the "computer" can easily find the right one to present. But doing this is usually a lot faster than preparing a clickable prototype.)
2. **You can make design changes more easily during the test.** A designer can sketch a quick response, and erase or change part of design between test sessions (or during a session) without worrying about linking the new page in the interactive prototype.
3. **Low-fidelity prototypes put less pressure on users.** If a design seems incomplete, users usually have no idea whether it took a minute or months to create it. They may better understand that you are indeed testing the design and not them, feel less obliged to be successful, and be more likely to express negative reactions.
4. **Designers feel less wedded to low-fidelity prototypes.** Designers are more likely to want to change a sketchy design than one with full interaction and aesthetics. Once we invest more time and sweat in a design, it's harder to give it up if it does not work well.
5. **Stakeholders recognize that the work isn't finished yet.** When people see a rough prototype, they don't expect it to ship tomorrow. Everybody on the team will expect changes before the design is finalized. (In contrast, when a design looks very polished, it's easy for an executive to fall into the trap of saying, "this looks good, let's make it go live now.")

Interaction with the User During Any Prototype Test

In prototype tests, facilitators often talk a bit more with participants than they do in tests of live systems, mainly for the following good reasons:

- They need to explain the nature of the prototype medium (not how the design itself works) to the user, before the study starts.
- They occasionally may need to explain the state of the system (e.g., "This page doesn't work yet") and ask "What were you expecting to happen?"
- They may have to find out whether users who sit idle are waiting for a response (from a slow system) or think that the task is completed.

Even with the above necessary interactions between the test facilitator and the user, the test facilitator's ultimate goal should be to ***quietly observe*** the person interacting with the design, not to have a conversation with the participant.

TIPS:

1. If users click an item for which there is no prepared response yet:
    - **Say: "** That isn't working."
    - **Ask:** "What were you expecting to happen when you clicked that?"
    - Present the second-best page if there is one, and say something as an explanation. For example, "You clicked the *compact cars* link. We don't have those screens today. So please pretend you clicked *midsize cars.* Okay?" After the user confirms, present the midsize-cars page. Then say as little as possible, and stay neutral.
2. If the wrong page appears after the user clicks, the "computer" should take that page away as soon as possible and revert to the previous page. The facilitator should tell the user immediately that the page was wrong, then repeat what the user did on the current page, "You tapped *About Us."* Then the "computer" presents the right page.

"Computer" Errors Have a Negative Impact

Note that "computer" errors can seriously impact the test. As screens appear, users form a mental model of how the system and the research method work. If the wrong page appears, don't assume that you can make users forget what they just saw. (Brain wipes only work in Star Trek.) Even if you take the screen away or try to explain the error, users may infer that the wrong screen is related to the task or get some more knowledge from your explanation, and may be later influenced by that information. Seeing the wrong page also breaks the users' flow, and can confuse them. Finally, later in the test, if a screen appears that is unexpected, they might think the prototype is just malfunctioning again. This impacts the users' expectations, trust in the research method, and ability to form a consistent mental model.

Since computer errors can negatively impact the study, take the time to pilot test and fix issues with the prototype before any sessions occur.

Conclusion

Make no mistake: You cannot afford to not test prototypes. Your design will be tested, whether you plan for it or not. Once your system goes live and people begin to use it, they are testing it. And rather than collecting feedback in a low-risk research setting, where you can learn, and then react and change the design, you'll have actual unhappy customers on your hands. With them will come a litany of problems such as lost sales, abandoned orders, lack of understanding of content and products, alienation due to poor tone of voice, returned

products, increased support calls, increased training costs, social shares of bad experiences, low Net Promoter Scores, and brand abandonment. The business will have to figure out how to fix all these. The development team will react by scrambling to fix the design, taking out working code, visuals, and content, and trading it for rushed, only marginally better replacements. All will come at a great cost. Redesigning, taking code out, coding again with the new design, quality testing that code, and, if applicable, changing marketing and documentation materials, is far more expensive than discarding a prototype.

Test prototypes, whether clickable or static, whether high- or low-fidelity. Aim to learn how to change and improve your design. That way, your customers will never see your design failures.