Creating Task Scenarios

What is a Scenario? ................................................................. 172

What Makes a Good Scenario? ............................................. 173
  Short.................................................................................. 173
  In the user’s words, not the product’s .................................. 174
  Unambiguous—so all participants will understand it. .............. 174
  Enough information to do the task ..................................... 175
  Directly linked to your tasks and concerns ......................... 176

Do You Always Give Participants Written Scenarios? .............. 177

How Do You Divide Up the Tasks and Scenarios for Participants? 179

How Do You Make Participants Stop Between Tasks? ............. 181
Once you have the list of tasks for the test, you have to decide how to present those tasks to the participants. One way that works well is to give the participants "scenarios"—situations in which the task is embedded in a reasonable and very short story. In this chapter, we look at the art of creating interesting and informative scenarios. We discuss these questions:

- What is a scenario?
- What makes a good scenario?
- Do you always give participants written scenarios?
- How do you divide up the tasks and scenarios for participants?
- How do you make participants stop between tasks?

### What Is a Scenario?

You use scenarios to tell participants what you want them to do during the test. Scenarios describe the tasks in a way that takes some of the artificiality out of the test. Here are two examples of scenarios:

**Scenario 1:**
You've just bought a new combination telephone and answering machine. The box is on the table. Take the product out of the box and set it up so that you can make and receive calls.

**Scenario 2:**
You can save numbers in the telephone's memory and then call those numbers without dialing all the digits each time. Your best friend's number is 212-555-1234. Put your friend's number in the telephone's memory.

Here's an example of a scenario for our usability test of the electronic mail program:

**Scenario 1:**
You have successfully completed installing the compiler. Test it with the program DEMO.FTN

**Scenario 2:**
You need to make a presentation to your manager about the month's sales figures. Copy the spreadsheet SALES1.WKS into the SALES table.

As you can see, a scenario makes the task more realistic. In a scenario, you give the goal and whatever information a user would actually have when coming to do this task. You do not give the steps. The point of the test is to see if a typical user can figure out the steps that this product requires.

### What Makes a Good Scenario?

A good scenario is:

- short
- in the user's words, not the product's
- unambiguous—so all participants will understand it

A good scenario:

- gives participants enough information to do the task
- is directly linked to your tasks and concerns

### Short

Time is precious in a test. You do not want participants to spend more time than necessary reading the scenarios.

People also read at different rates. Because you are timing the task, you do not want these different reading speeds to unduly influence the task times. The way to keep that difference small is to write short scenarios.

Depending on the software or other means you are using to time the tasks, you may or may not be able to separate the time participants spend reading the scenario from the time they spend doing the task. See Chapter 15 on "Preparing the Test Environment" for more information on measuring time.

Most scenarios can be kept very short. If you set up the entire test so that the participant has a consistent and plausible "role," you can write each scenario to fit into that role.

Here are a few more examples of scenarios:

**Scenario:**
There have been some staff changes in your office.
Set up a new account for E. Dickenson.

**Scenario:**
You have successfully completed installing the compiler.
Test it with the program DEMO.FTN

**Scenario:**
You need to make a presentation to your manager about the month's sales figures.
Copy the spreadsheet SALES1.WKS into the SALES table.
In the User's Words, Not the Product's

The whole point of usability testing is to predict what will happen when people use the product on their own—without a developer or usability specialist looking over their shoulders or answering their questions.

If you are testing a menu-driven product, one of the concerns you are probably testing is whether users will choose the right menu option. If you are testing a graphical user interface with icons, one of your concerns is likely to be whether users will select the right icon. If you are testing a device with labeled buttons, you are likely to be concerned about whether users will know which button does what.

Don't give that information away in your scenarios. If you do, the product may do well in testing and still fail in the marketplace.

Suppose you are testing a product that lets users save their mail in "logs." The menu choice for this is "log." If you write a scenario that says, "Now log the mail that you just read," the participants may successfully do the task without even understanding what task they are doing. When other users get the product in their offices, they may never store their mail because they make no connections between the task as they say it to themselves and the menu choice "log."

To test this task realistically, you must write a scenario that describes the task the way users will say it to themselves. That might be: "Now store the mail you just read." Or it might be: "You don't want that message to sit in your list of mail forever, but you may want to read it again later. Save it so that you could get it back to look at later." Now you have a realistic test of whether users recognize the product's word "log" as the right one for the task they represent to themselves as "storing" or "saving."

This issue of making sure the scenarios are worded well is very important. However, don't go overboard. If you have done a lot of previous usability work to be certain that you are using plain English, users' terms, don't make up new, unusual terms for the scenarios.

Developers may have a hard time seeing this problem because the words in the product are so familiar to them that they cannot imagine users not understanding the terms. A usability specialist should either write the scenarios or review them carefully, looking for words and phrases that are product words and might not be the ones users would look for.

Unambiguous — So All Participants Will Understand It

You're trying to see how easy the product is to use. You don't want misunderstandings about the scenarios to interfere with learning as much as you can about the product.

When writing scenarios, watch out for these common pitfalls:

- not being clear about when the task is over
- not giving the information people need to do the task

In this scenario, for example, where are you supposed to stop?

Ambiguous scenario:
You have a message from Jane Jones about the Fourth Quarter Budget. Read it and write a response in which you answer her question.

Should you send the reply or just compose it and wait for further instructions?

If any of the scenarios are ambiguous, you are likely to see the problem in pilot testing. This is one reason for conducting a pilot test, as we explain in Chapter 17.

Enough Information to Do the Task

In addition to telling participants the general task or goal that you want them to accomplish, you may need to give users some data to work with. If, in a real situation, they would have information about a specific case when they come to do the task, you have to supply the information for such a case.

Suppose you are testing a new touchscreen product for retailers. You've set up a general scenario for your participants so they are in the role of a salesperson in the linen department. One of the tasks you are concerned about is whether they can handle a sale that includes multiple items. You have to give them a situation in which that task will occur.

If you are giving the scenarios in writing, you might create one like this:

Scenario:
A customer comes up to your station with several items. She wants to buy them on her store charge card. She gives you her store charge card (Account #9-80-786-5). And she gives you the items, which are

- two twin-bed-size sheets. Each sheet is item #347689.
- a package of pillow cases. The package is item #346988.
- two pillows. Each pillow is item #456897.

Ring up the purchase for her.

Note: For this test, a better way to deliver the scenarios might be to have team members pretend to be customers. For this scenario, someone would walk up to the participant/salesperson with the items and a store credit card. See the section, "Do You Always Give
pants Written Scenarios? later in this chapter. You still have to plan all this information and make sure you are using for the test has that customer and those items in it. Otherwise what you are really testing is how users deaf with the messages they'll get when they try to enter a nonexistent account number or incorrect item numbers.

Note that you haven't told the participants to do the task. You've only given them the same information that they would have if they were doing such a task in real life. That's the key to a useful scenario: the participants should feel as if the scenario matches what they would have to do and what they would know when they are doing that task in their actual jobs.

You have to give the participants enough data to be able to do the task. Don't give them extra data unless they are likely to actually have that data when doing the task and you are specifically testing a concern about whether they will know which data to use in the task.

You don't always have to spell out all the data. For example, if you are testing an editing program and you want the participants to write a note, you don't have to give them the exact wording of the note, but you should give them some information to write about.

We find that if you give nonspecific instructions, such as “Write a note to Brett Jones at Headquarters,” some people will spend a great deal of time deciding what to write about. That throws off your comparison of time for the task you are testing. We also find that if you give participants all the words to type, however, the task is boring. What works best is to tell them something like this: “You need to let Brett Jones at Headquarters know about the staff meeting next Friday at 2 p.m. in the main conference room.” With this much information, you will get to observe how each participant goes about addressing and composing a note—whether you want to know—and yet not find large disparities in the time they spend deciding what to say—which you do not want to know.

**Directly linked to Your Tasks and Concerns**

Each scenario tests one or more of the tasks on the list that you decided to include in the test. Each of those tasks, in turn, is directly linked to one or more of your goals or concerns.

You don’t give these task names to the participants. They get only the scenarios. But the team should know exactly what each scenario is testing. Sometimes the task-scenario link is obvious from the wording of the scenario, but sometimes it is not.

The following is an example in which the link is obvious:

**Concern:** Can users figure out how to use the remote control to turn off the television set?

**Task description:**

You've finished watching the TV for now. Turn it off.

**Scenario:**

Participants have been told to use the remote control for all tasks.

Turning off the TV.

You've finished watching the TV for now. Turn it off.

In this second example, the link between the task and the scenario is not obvious from the wording of the scenario:

**Concern:** Will users understand the messages on the small one-line screen of the photocopier?

**Test setup:**

The photocopier has only two sheets of paper in the bin that will be used in this task. Packages of different size paper are available on a supply table nearby, but the participant has not been told that they will be needed.

**Task description:**

Figuring out what is wrong; adding paper correctly.

**Scenario:**

Please make one copy of this five-page paper.

---

**Do You Always Give Participants Written Scenarios?**

No, you don't always give the scenarios in writing. You want to make the test as realistic as possible, and another mode of delivering the scenarios may be more realistic than having the participants read them. It might be appropriate to

- have test team members pretend to be “customers,” “supervisors,” or “colleagues” and walk into the test room to deliver the scenarios in person
- use the product to deliver the scenarios

For example, in testing telephones, the scenarios would
probably include telephone calls to the participant as well as calls the participant makes.
bullet mix the modes; give participants scenarios in writing, but then interrupt with calls or visits.

Whatever mode you use to present the scenarios, however you must write them out as part of your test plan. Even if the participant never sees the written scenario, you must be sure that each participant gets the same scenario delivered in the same way. Therefore, if test team members are play-acting scenarios over the phone or in person they have to use the exact same scenarios with each participant. That may mean memorizing the situation and words as if this were theater.

The following are some examples of test situations that have lent themselves to different modes for delivering the scenarios:

- To test a new part of the airlines reservation and check-in system, the test team at American Airlines set up a ticket counter in the usability laboratory. The participants were representative gate agents. They stood behind the counter as they do on their jobs. The scenarios were delivered by test team members acting as travelers.

- To test a new multifunction telephone, the test team at AIR set up the usability laboratory to look like an office. The participants were representative office workers. They were each given a set of written scenarios, but the test team knew that not all the scenarios were included in the participant’s package. The test team’s version of the package included some scenarios that the test participant did not see, such as: “Call the participant on the phone and say: . . . .”

- To test an objective that users be able to easily suspend what they are doing and turn to a different task, the team testing an electronic mail package decided to interrupt participants in the middle of writing a memo. The participants started what they thought from the written scenario was a task to send a memo on a particular topic to a particular person. A person on the test team had instructions to interrupt over the intercom after the participant had addressed the memo and typed about two lines of the message. The prescribed interruption gave the participants a plausible reason to want to stop working on the memo they were writing and instead immediately send a different message to a different person.

- AIR conducted a test of a monitor that nurses would use at a central nursing station on a hospital floor. One of the concerns was whether nurses would know what to do when two alarms went off at the same time. The nurses got a scenario that told them to monitor their patients’ status. While they were doing that, two alarms went off. Preparing for that scenario required writing software to make the alarms sound during that task.

Match the mode of presentation to the situations in which the product you are testing will be used.

**How Do You Divide Up the Tasks and Scenarios for Participants?**

If you have a list of 15 tasks that you want to test, you may have exactly 15 task scenarios for the participants, but you may have more or less. You do not have to have a one-to-one correspondence between tasks on your list and your scenarios.

You may have more scenarios than tasks if you want to test the same task more than once. One of your concerns may be how quickly users will learn to do a particular task. You may hypothesize that users will make a few wrong icon choices the first time they try something, but then, after that, they’ll know which icon to select and won’t make any errors the second time they need to use that icon. In that case, you might have two or three scenarios for the same task in your test.

You may have fewer scenarios than tasks if you combine two or three tasks and have participants do them together.

In deciding how to match up scenarios and tasks, the major issue you have to consider is whether you want separate measurements of time, errors, or other codes for particular tasks. If the software program that you are using to measure the participants’ performance can automatically give you time only between the code for “start task” and the code for “stop task,” you have to give each task for which you want a separate time as a separate scenario. An example may help to explain this point more clearly:

Let’s say you are testing a project management program. You are concerned that it will be accepted by the company’s managers only if it is faster to use than keeping their manual records. You know that speed in doing the tasks is dependent not only on the system’s response time, but also on how quickly managers find the right menu choices and fill in or change. Time is one appropriate measure of the ease of use of this product.

Your task list for this test includes these three tasks that project managers have to do:
- get a project’s file
- make changes to the schedule
- add a new person to the project team
You could write a scenario that had your project manager participants do all three of these at once. In that case, you would automatically know only the total time, total errors, and total frustrations for the combined set of three tasks.

However, you might want to know how much time it takes the participants to do each part, how many errors they make doing each part, and whether one of these tasks is more frustrating than the others. In that case, you would create three scenarios. Participants would stop at the end of each one. The data-logging software would give you the time, error count, and frustration count for each task, that is for each scenario, separately.

In the first case, the one in which you have the participants do all three tasks as part of one scenario, you would give the participants a page that might look like Figure 12-1.

Task 1

You are the project manager for the BOOK project. You’ve just found out that there’s been a change in schedule and staffing for the project and you have to make the computer files reflect those changes.

- Get the BOOK project on your screen so that you can change it.
- Deliverable #3 is going to be two weeks late. Change the file to reflect that.
- To make even that new deadline, your boss has authorized you to add another designer, Jed Brown, full time for two weeks starting tomorrow. Jed earns the same salary as Betsy Moore. Add Jed to the project staff.

Figure 12-1. Three tasks as one scenario, timed together

In the second case, in which you want to know separate times and other counts for each task, you would give the participants the same instructions, but you would separate them into three tasks, giving each task as a separate scenario on a separate page. It might look like Figure 12-2.

Task 1

You are the project manager for the BOOK project. You’ve just found out that there’s been a change in schedule and staffing for the project and you have to make the computer files reflect those changes.

Get the BOOK project on your screen so that you can change it.

Task 2

Deliverable #3 is going to be two weeks late.

Change the file to reflect that.

Task 3

To make even that new deadline, your boss has authorized you to add another designer, Jed Brown, full time for two weeks starting tomorrow.

Jed earns the same salary as Betsy Moore.

Add Jed to the project staff.

Figure 12-2. The same three tasks as separate scenarios, timed separately

How Do You Make Participants Stop Between Tasks?

If you want to time the tasks separately, you have to get the participants to stop between each task. In some organizations, the usability testers give participants the tasks one at a time. That way,
they always know when the participant has finished one task and when the participant begins the next task. If you run your test that way, you can also ask questions about the task, conducting a mini-interview after each one.

However, if your test consists of many short tasks, going in to the participant between each one may be disruptive. In a typical test in the AIR lab, we give the participants a booklet of the scenarios that we want them to do during the test. Each scenario that we want to time separately is on a separate page. Because “task” is an easier and clearer word for participants than “scenario,” we label the pages as “Task 1,” “Task 2,” and so on.

To remind participants to stop at the end of each task, each page includes two sentences near the bottom:

Please tell us when you have finished this task.
Please wait for us to tell you to turn the page.

At the beginning of the test, the person who interacts with the participant, the “briefer,” also reminds the participant:

- to wait for the briefer to say when to begin the first task
- to say out loud when the task is done
- to wait again between each task until the briefer says to go ahead

Putting the scenario for each task that you want to time separately on a separate page is one way to get participants to stop between each task. The entire test team will appreciate having a few moments between each of the participant’s tasks to finish their own work and be ready to observe again.

When we want to get participants’ reactions to each task, we include a short posttask questionnaire in the booklet after each task. You can see an example of a posttask questionnaire in Chapter 14 on “Preparing Test Materials.” A posttask questionnaire, like the mini-interview that some teams do between tasks, allows you to get the participant’s immediate reaction to each task. Answering a few questions after each task also gives the participant something to do while the test team finishes logging their own comments.

Now that you have turned your task list into the specific scenarios that you’ll give to the participants, we’ll turn to the question of what you want to observe and measure while participants are working with the product: