• a recruiting questionnaire and the recruiter’s notes
• a nondisclosure form
• a pretest questionnaire
• the task scenarios
• a posttest questionnaire
• notes the test team makes during the test
• the debriefer’s notes from the posttest interview
• a printout of the test log
• a receipt for the participant’s incentive

On your checklists, you should have items for putting the number of the test participant on all printed materials.

If you make notes on what is happening during the test, you must put the participant’s number on them. If you do not, you will soon forget which participant the notes are for.

Keep a file folder for each participant and put all of the materials into that folder.

---

**Observing Problems and Creating a Problem List**

As with any empirical method, the keys to success in a usability test are the observations and measurements you make during it. In order to keep your observations organized, you need some means for tabulating events, either a software-based or a manual datalog. But you also need to record what you are seeing at another level. When participants have problems of any kind with the product, you need to record that problem and, in some cases, its frequency. We recommend that from the first minute of the first participant to the last minute of the last participant, the person responsible, usually test administrator or the briefer, keep a running list of the problems they see. When other members of the test team see problems, they should have the responsible person add them to the list.

In our tests, the test administrator has a pad of paper with the pages still attached. The participant’s number goes on the top of page 1, and the administrator keeps a running list of problems as the participant proceeds through the tasks.

What we record on this list are observations (what we actually see) and inferences (our hypotheses and interpretations). What we want to understand at the end of a usability test are the real, underlying problems—the ones that have to be fixed. The behaviors we observe may be symptoms of these problems. Our hypotheses and interpretations help us understand the scope and severity of these problems—and give us insights for recommending good solutions.
Amanda Prail of Hewlett-Packard has captured many of these points very well in an article in the newsletter *Common Ground* (Prail, 1991). Figure 19-1 quotes Prail’s suggestions on collecting observational data.

**Suggestions on collecting observational data**

**Before observing**
- Know the product that the user is using.
- Know the goals of the task the user is performing.
- Get into the observer role:
  - Respect the user.
  - Assume there are lots of opportunities to improve the design.
  - Be very curious.
  - Become detached, uninvested in the outcome.
  - Think of yourself as an anthropologist.

**While observing**
- Keep 100% of your attention on the user.
- Observe actively. Think of all potential user actions and construct hypotheses about observed actions.
- Write down both observations and your hypotheses and interpretations.
- If design ideas occur to you, write them down, but save design discussion for after the session.
- Stay away from your own thoughts. Censor any tendencies to rationalize or defend the design or to dismiss problems that “can’t be fixed.”
- Record all the user problems, whether they are part of your design or not.

**After observing**
- Discuss your interpretations of user problems with others to discover differences.
- Ask questions of the user during debriefing to confirm unsupported or debated interpretations.
- Make sure you have support from the user for all of your interpretations before examining potential design solutions.

We recommend that if you are new to testing, you should focus on making observations first and leave the interpretations until after the participant is done and you have time to think and talk about what you have seen.

With practice, you will be able to move from observations to problems more quickly, and you will spend less time in your end-of-day discussion of the day’s problems. Having an understanding
of the context of research and experience with human-computer interaction helps you to interpret the meaning of observations.

Eventually, you want to get to the basic causes of the problems you see. But you must become skilled at making the observations before you learn how to interpret them. For example, as you watch a participant try to complete a task, you see that he says that he does not understand the word toggle in an instruction and makes an error by following the wrong procedure. Your observation, then, is the fact that there was a statement by the participant about not understanding the word toggle and that the wrong procedure was followed.

What interpretations and hypotheses can you make from this observation? There are several that are possible, and part of the skill of a good tester is seeing the range of possibilities and then deciding which one explains the most observations.

For example, this observation may only be the first of a series of observations about the wording of messages. There may be several other cases that would eventually lead you to believe that the interface is filled with jargon. You might hypothesize from this first instance that there may be jargon in other messages. You would then be on the lookout for future observations about problems participants have with words.

You might also hypothesize that the messages were written by software engineers not writers, and that there may be other evidence of this fact, such as awkwardly written messages and online help text. You would want to look for these instances and ask the developers who wrote the messages.

Experienced testers can write down observations and also some hypotheses or even keep the hypotheses in their heads. They also know that there is a danger in forming hypotheses early, because hypotheses can bias observations when they influence what is observed, and because they can make you miss observations that you would otherwise see. In our example, testers may be more likely to see difficulties with wording when there really are none, or they may be so focused on language that they miss problems with graphics.

As we will see in the next chapter, the problem list you create during the test becomes a tool for structuring the way you analyze the data. Some test teams take the first day’s hurriedly written notes and create a more organized, formal list of problems. After the second day, the team adds any new problems to the formal list. They also look to see whether the second day’s participants had any of the same problems as the first day’s participants. If so, the team increases the frequency of the first day’s problems by one.

When you keep this organized problem list up to date, you have a major part of your data analysis done when you have finished conducting the test. Keeping a formal problem list is a good practice when you are new to testing.
interacting with Participants

In the last chapter, “Caring for the Test Participants,” we discussed how the briefer gets the participant settled into the test and some of the problems that may arise in testing, so that you can develop policies for handling those situations.

In this chapter, we are concerned with who interacts with the participant and how to deal with normal interactions with the participant during the test. In particular, we discuss five issues that you may need to think about as you conduct the test:

- Who interacts with the participant during the test?
- When and how should you interrupt the participant?
- How do you avoid biasing test participants?
- Who decides when a participant has finished a task?
- What makes an effective help desk in a usability test?

Who Interacts with the Participant During the Test?

Even if you have several people on your test team, you want to keep the number of people with whom the participant has to interact to a minimum.

If you have a test team member acting as the help desk, that person talks to the participant—if and only if the participant calls.

If you have different people delivering the scenarios as if they were customers, either in person or over the phone, of course, you’ll have more people interacting with the participant. That is a fairly unusual circumstance, however.

In most tests in which there are separate observation and test rooms, the participant works alone in the test room. Other than calls to the help desk, only the briefer interacts with the participant during the test. While the participant is working on the tasks, the briefer does not go back into the room. The briefer can talk to the participant over the intercom when one is available. In tests in which there is a member of the test team in the room with the participant, that person will be the one who interacts with the participant.

When and How Should You Interrupt the Participant?

When to interrupt a test participant is a matter of some controversy. Some organizations keep interruptions to a minimum, while others talk freely with the participant. The difference between the two styles depends, to some extent, on the goals of the test. If it is important that the test simulate a situation in which users will be working alone without help, keep interruptions to a minimum. For example, for some organizations it is important to find problems that will cause users to call the company’s customer support line. In tests for these
organizations, therefore, the test team takes a hands-off approach when participants have problems with the product. The test team provides help to the participant only through a help line that the participant calls from the test room.

In tests of early prototypes, on the other hand, the designers usually want to get as much diagnostic information as possible. In these tests, there is more interaction with the participants. For example, the briefer may ask participants to repeat a task using a different method or to describe why they did a task the way they did.

In general, you want to let the participants do each task at their own pace and in their own way. That’s what you are there to observe.

It may be very difficult to sit behind the one-way mirror and watch participants struggle. You may find yourself desperately wanting to help. Don’t give in to the impulse to jump in. You are there to watch what will most likely happen in people’s offices and homes.

In the first few tasks, the briefer may need to interrupt to encourage the participant to think out loud. (See Chapter 18, “Caring for Test Participants,” for suggestions on getting quiet participants to verbalize their thoughts.)

If the participant does not stop between tasks, the briefer must quickly interrupt and remind the participant to say when he or she finishes one task and then wait to be told when to turn the page for the next task. Other than those two circumstances, encouraging participants to think out loud and making sure they stop between tasks, you should normally not have any need to interrupt unless you have decided to probe for diagnostic information.

In some tests of early prototypes, we create a set of probes to ask participants when they do not comment on a procedure or when they do not do the task the way we want them to. For example, we might be concerned about what participants would look for in online help during a particular task. If participants do not use online help during the task, we might ask them what they would have looked for. The probe is a reminder to the briefer to ask the question when the participant completes the task without using online help.

At the end or the beginning of tasks, you may want to add words of support or encouragement to your acknowledgment that the task is completed or to your instruction to go on to the next task. That support can be words such as:

“Thank you. The comments you are making are very helpful. We appreciate them. Please continue to think out loud like that.”

“OK. Jim, how are you doing? Are you ready to do the next task?” [Participant says, “Yes, I’m ready.”] “Good, let’s start Task 5, and please continue to think out loud.”
How Do You Avoid Biasing Test Participants?

In the last chapter, we talked about how to encourage participants to think out loud and how to reduce their stress. Here, we want to discuss how to talk with participants in a way that allows you to get the most information you can from them without biasing them or giving them information that may help them with tasks.

During the 1960s and 1970s, researchers discovered that many participants in research studies try to figure out what the experimenter wants them to do and then they do it. In addition, experimenters bias studies by influencing what participants say and do (Rosenthal, 1976). Translated into the usability testing environment, these discoveries suggest two points:

1. Test participants are recruited specifically for the test and are usually reimbursed. Consequently, some participants will be looking for ways to be “good” participants. They want to know what the test administrator wants them to do. Our impression from what we see in our lab is that participants assume that the test team is somehow involved in the design of the product and wants the participant to like the product. This may cause them to make positive comments or refrain from negative comments about the product or to give it higher ratings than they ordinarily would. These expectations can change, however, if the tester reinforces the participants for making negative comments.

2. Testers can influence the way participants act and what participants say by biasing them with leading questions and encouraging either positive or negative statements.

The usability testing environment is a place in which these sources of bias exist. You need to be careful to avoid them. You will need to exercise caution when:

1. you ask questions of the participant
2. you respond to participants’ questions and comments

How to Ask Questions

There are many times when you want to ask questions of the participants in a test. You may be probing them for additional information, or you may be asking about something they did or said.

When you ask these questions, you need to use neutral words. A
A good way to know whether you may be biasing the participant is to examine how you use adjectives and adverbs in your questions? In English, adjectives and adverbs often carry an evaluative component with them. For example, if you ask the participant whether they found a procedure “easy,” or whether they performed a task “easily,” you may be biasing them by focusing on the positive end of the evaluative dimension from easy to difficult. It is better to question them in a more neutral way, such as “How was it performing that procedure?” or “Did you find that procedure easy or difficult?” The first form of the question contains no adjectives or adverbs. Sometimes, you may find that you need to use the second form so that participants describe their experience along a specific dimension, in this case the ease or difficulty of a task. Table 19-1 contains a list of useful neutral questions to ask, biased questions to avoid, and a short explanation of why the neutral questions are preferable.

How to Respond to Participants’ Questions and Comments
Sometimes the participant asks you a question. These questions can make you feel uncomfortable, because you feel pressured into giving the participant information you do not want to give them. There are some strategies you can use in these situations.

**Turn the question around.** Participants might ask, “Do I use Help to find that out?” The participants may be asking you to give them the answer, or they may just be unsure. In either case, you do not want to lead them. You might say, “What do you think you should do?” or “Do whatever you think you should do,” or “I would like you to figure that out,” or “I can’t tell you what you should do, but I’d like you to keep trying.”

Participants might ask, “Did I do that right?” An appropriate response would be, “Do you think you did it right?” Knowing how confident participants are can be useful information. If they are not confident, the software may not provide enough feedback to their actions.

**Don’t answer the question directly.** Participants might ask, “Did everyone else have as much trouble as I did?” You might respond, “Did you have more trouble than you expected?”

Participants might ask, “Do you want me to tell you when I don’t like something?” You might respond, “Tell me what you like and what you don’t like,” or “I’d like to hear any reactions you have about the product, good or bad.”

---

1 We call this rule Coleman’s Rule in honor of Marilyn Coleman who first brought it to our attention.
Table 19-1 Neutral questions to ask, biased questions to avoid, and why the neutral questions are preferable

<table>
<thead>
<tr>
<th>Ask:</th>
<th>Instead of:</th>
<th>why:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Can you tell me what you are thinking right now?</td>
<td>Are you thinking</td>
<td>Even though you may think you know what they are thinking (that may be why you are asking the question), you do not want to put words into their mouths, because you may be wrong. Remember, you do not want participants to know your opinions about what you are questioning them on.</td>
</tr>
<tr>
<td>What are you trying to do?</td>
<td>Are you trying to</td>
<td>By asking participants why they are doing something, they may feel that you are asking them to justify their actions, and, therefore, think that they are going about the task incorrectly.</td>
</tr>
<tr>
<td>What are you thinking?</td>
<td>Why are you</td>
<td>It is, however, acceptable to ask a participant why they went about a task in a certain way after the task has been ended, or at the end of the test if future tasks have components similar to the task you are questioning them about.</td>
</tr>
<tr>
<td>Can you explain to me what you are trying to do?</td>
<td>Are you trying to</td>
<td>Trying to get someone to express an opinion on a specific usability attribute is not always easy. Therefore, you may find that you need to guide participants by specifying the attribute you want them to react to. It is important to use both ends of the spectrum when you do this so that they do not perceive you as encouraging either a positive or negative answer. Also, by doing so, you will encourage a more informative response. Instead of responding “No (it was not easy),” they are more likely to say “I found it very difficult to use,” or “it was pretty easy.” You then can follow up by asking them “why?”</td>
</tr>
<tr>
<td>Can you explain to me your train of thought right now?</td>
<td>Are you trying to</td>
<td>Sometimes, participants need to stop and think—maybe to try to remember how a similar product worked. Though they may appear confused or frustrated, they may just be contemplating. Everyone expresses themselves differently, so we take a risk by trying to guess what they are thinking.</td>
</tr>
<tr>
<td>(After the task is ended) Why did you try to</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Did you find the product easy or difficult to use?</td>
<td>Did you find the product easy to use?</td>
<td></td>
</tr>
<tr>
<td>Were the instructions clear or confusing?</td>
<td>Did you find the product difficult to use?</td>
<td></td>
</tr>
<tr>
<td>Were error messages helpful or hindering?</td>
<td>Were the error messages helpful?</td>
<td></td>
</tr>
<tr>
<td>What are you feeling?</td>
<td>Are you feeling confused?</td>
<td></td>
</tr>
<tr>
<td>How did you feel when you were doing _____?</td>
<td>Are you feeling frustrated?</td>
<td></td>
</tr>
</tbody>
</table>

(continued on next page)
<table>
<thead>
<tr>
<th>Ask:</th>
<th>Instead of:</th>
<th>why:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Would you change anything about this (product, screen, design, etc.)?</td>
<td>Do you think ___ would improve the product?</td>
<td>Unless the design team is considering a particular design change, you should never suggest what changes participants should talk about.</td>
</tr>
<tr>
<td>Are there any changes you would make to ___ to make it easier to use?</td>
<td>If we changed ___ to ___ do you think that it would be easier to use?</td>
<td>Always let participants express their own ideas. And, if there is a design change that the design team feels will improve the product, ask them to react to it only after they have made their suggestions and only after you let them know that their input is considered valuable.</td>
</tr>
<tr>
<td>How do you feel about (that error message, the organization of the screen, the way the procedure is described in the manual, etc.)?</td>
<td>Was the (error message, the organization of the screen, the description of the procedure, etc.) confusing?</td>
<td>Even though you may think you know what they are thinking (that may be why you are asking the question), you do not want to put words into their mouths, because you may be wrong.</td>
</tr>
<tr>
<td>Do you have any reactions to (that error message, the organization of the screen, the way the procedure is described in the manual, etc.)?</td>
<td>Are you confused by the (error message, the organization of the screen, the description of the procedure, etc.)?</td>
<td>Remember, you do not want participants to know your opinions about what you are questioning them on.</td>
</tr>
<tr>
<td>Do you have any comments on the (appearance, size, feel, feedback, etc.) of ___?</td>
<td>Do you like the (appearance, size, feel, feedback, etc.) of ___?</td>
<td></td>
</tr>
</tbody>
</table>
When the Participant Isn't Ready to Say the Task is Done

In many tests, we have seen participants spend time wondering whether they really did the task, checking to see if it was done, or even redoing it several times because they are not sure. All of that is valuable information for you.

If users in their offices or homes are going to spend extra time trying to figure out if the message was really sent, if the file was really saved, or if the changes were really made, that’s part of the time it will take them to do the tasks. Moreover, if users redo the task because they are not sure the software has actually carried out their instructions, the consequences could be more significant than just extra time.

For example, in testing an electronic mail package, we were not comfortable with the way the product handled replying to a memo. While the memo they had received was on the screen, users correctly chose “Reply” from the menu, typed their reply, and then chose “Send” from the menu to send it. The computer did in fact send both their reply and the original memo, if they chose to include it with the reply.

The system message that the reply was being sent flashed by so quickly on the screen, however, that very few users noticed it. They thought they had no word from the computer that the reply had been sent. Furthermore, after they chose “Send,” the product reset the screen as it had been at the beginning of the task, with the original memo ready to be read but without the reply that the user had written.

Users, after looking at this screen and completing the task correctly, concluded that they had somehow messed up and that their reply had been deleted or had somehow been sent off to oblivion. They redid the task. Some redid it three times before concluding that it must have worked even though they were still uncomfortable about it.

Only by letting the users continue until they said they were done would the product developers see what a problem not having a readable message like “Reply sent” was to these users. This was an important problem to see, because the consequences are not just the extra time that users will spend redoing the task, but the embarrassment users might suffer from sending three identical replies to the same writer about the same memo and the confusion and time of the writer who receives the three identical replies.

If the test team, not users, had been the ones to decide when the task was done, they would not have learned about this important problem.

When the Participant Says the Task is Done, But It Is Not

One of the issues you will have to address is deciding when to ask participants to continue with a task that they either think they’ve
completed or want to give up on. In setting policy for this issue, consider these three points:

- You must be consistent for all participants. Otherwise, you won’t have data that you can compare.
- You should be compassionate. If you have seen enough to know the product needs work, don’t push participants beyond endurance.
- But, you want to learn as much as you can.

If the situation isn’t overly frustrating for the participants and if they haven’t been working at the task an inordinate amount of time, you will usually learn more by asking them to continue to work on the task.

For example, in testing a new telephone with a small display screen, we asked participants to do a task that involved changing the setting for a feature. Participants did not connect the feature with the display and usually gave up after trying-and failing-to find a specific button to press or wheel to turn. We could have let them stop and told the developers that users don’t expect to use the display to change that particular feature.

However, we didn’t let the participants stop. We asked them to continue to work at the task. We were able to observe two other very important problems—that the quick reference card didn’t help these participants, and that they couldn’t make the change in the time that the system allowed them before automatically clearing the display.

These were both global problems that the client needed to fix before releasing the product. We may not have seen them if we had let participants give up on the task.

What Makes an Effective Help Desk in a Usability Test?

Deciding when to give help and what help to give is one of the most difficult parts of conducting a usability test. If you are counting error rates as one of your measurements, you need to let participants make errors. Those errors are among your most valuable pieces of data. If you let participants give up easily, you may be losing information that you need to keep other users from making the same errors with the final product. Therefore, you need to let the participant initiate the request for help, and you need to give the minimum amount of help that you can.

One of the reasons for having participants use the telephone to call a “help desk,” rather than just letting the participant ask the briefer for help, is to see people’s tolerances for working out problems before calling for help.