Usability Testing

TR 29.3820
August 24, 2006

James R. Lewis

IBM Software Group

Boca Raton, Florida

Abstract

Usability testing is an essential skill for usability practitioners – professionals whose primary goal is to provide guidance to product developers for the purpose of improving the ease-of-use of their products. It is by no means the only skill with which usability practitioners must have proficiency, but it is an important one. A recent survey of experienced usability practitioners indicated that usability testing is a very frequently used method, second only to the use of iterative design. One goal of this chapter is to provide an introduction to the practice of usability testing. This includes some discussion of the concept of usability and the history of usability testing, various goals of usability testing, and running usability tests. A second goal is to cover more advanced topics, such as sample size estimation for usability tests, computation of confidence intervals, and the use of standardized usability questionnaires.

ITIRC Keywords

Usability evaluation
Usability testing
Formative
Summative
Sample size estimation
Confidence intervals
Standardized usability questionnaires

NOTE: The contents of this technical report have been published as a chapter in the Handbook of Human Factors and Ergonomics (3rd Edition) – Lewis, J. R. (2006). Usability testing. In G. Salvendy (ed.), Handbook of Human Factors and Ergonomics (pp. 1275-1316). Hoboken, NJ: John Wiley. The most recent version of this technical report is available at http://drjim.0catch.com.

Contents

INTRODUCTION

Usability testing is an essential skill for usability practitioners – professionals whose primary goal is to provide guidance to product developers for the purpose of improving the ease-of-use of their products. It is by no means the *only* skill with which usability practitioners must have proficiency, but it is an important one. A recent survey of experienced usability practitioners (Vredenburg et al., 2002) indicated that usability testing is a very frequently used method, second only to the use of iterative design.

One goal of this chapter is to provide an introduction to the practice of usability testing. This includes some discussion of the concept of usability and the history of usability testing, various goals of usability testing, and running usability tests. A second goal is to cover more advanced topics, such as sample size estimation for usability tests, computation of confidence intervals, and the use of standardized usability questionnaires.

THE BASICS

What is Usability?

The term 'usability' came into general use in the early 1980s. Related terms from that time were 'user friendliness' and 'ease-of-use,' which 'usability' has since displaced in professional and technical writing on the topic (Bevan et al., 1991). The earliest publication (of which I am aware) to include the word 'usability' in its title was Bennett (1979).

It is the nature of language that words come into use with fluid definitions. Ten years after the first use of the term 'usability,' Brian Shackel (1990) wrote, "one of the most important issues is that there is, as yet, no generally agreed definition of usability and its measurement." (p. 31) As recently as 1998, Gray and Salzman stated, "Attempts to derive a clear and crisp definition of usability can be aptly compared to attempts to nail a blob of Jell-O to the wall." (p. 242)

There are several reasons why it has been so difficult to define usability. Usability is not a property of a person or thing. There is no thermometer-like instrument that can provide an absolute measurement of the usability of a product (Dumas, 2003). Usability is an emergent property that depends on the interactions among users, products, tasks and environments.

Introducing a theme that will reappear in several parts of this chapter, there are two major conceptions of usability. These dual conceptions have contributed to the difficulty of achieving a single agreed upon definition. One conception is that the primary focus of usability should be on measurements related to the accomplishment of global task goals (summative, or measurement-based, evaluation). The other conception is that practitioners should focus on the detection and elimination of usability problems (formative, or diagnostic, evaluation).

The first conception has led to a variety of similar definitions of usability, some embodied in current standards (which, to date, have emphasized summative evaluation). For example:

- "The current MUSiC definition of usability is: the ease of use and acceptability of a system or product for a particular class of users carrying out specific tasks in a specific environment; where 'ease of use' affects user performance and satisfaction, and 'acceptability' affects whether or not the product is used." (Bevan et al., 1991, p. 652)
- Usability is the "extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use." (ANSI, 2001, p. 3; ISO, 1998, p. 2)
- "To be useful, usability has to be specific. It must refer to particular tasks, particular environments and particular users." (Alty, 1992, p. 105)

One of the earliest formative definitions of usability (ease-of-use) is from Chapanis (1981):

> "Although it is not easy to measure 'ease of use,' it is easy to measure difficulties that people have in using something. Difficulties and errors can be identified, classified, counted, and measured. So my premise is that ease of use is inversely proportional to the number and severity of difficulties people have in using software. There are, of course, other measures that have been used to assess ease of use, but I think the weight of the evidence will support the conclusion that these other dependent measures are correlated with the number and severity of difficulties." (p. 3)

Practitioners in industrial settings generally use both conceptualizations of usability during iterative design. Any iterative method must include a stopping rule to prevent infinite iterations. In the real world, resource constraints and deadlines can dictate the stopping rule (although this rule is valid only if there is a reasonable expectation that undiscovered problems will not lead to drastic consequences). In an ideal setting, the first conception of usability can act as a stopping rule for the second. Setting aside, for now, the question of where quantitative goals come from, the goals associated with the first conception of usability can define when to stop the iterative process of the discovery and resolution of usability problems. This combination is not a new concept. In one of the earliest published descriptions of iterative design, Al-Awar et al. (1981) wrote:

> "Our methodology is strictly empirical. You write a program, test it on the target population, find out what's wrong with it, and revise it. The cycle of test-rewrite is repeated over and over until a satisfactory level of performance is reached. Revisions are based on the performance, that is, the difficulties typical users have in going through the program." (p. 31)

What is Usability Testing?

Imagine the two following scenarios.

Scenario 1: Mr. Smith is sitting next to Mr. Jones, watching him work with a high-fidelity prototype of a web browser for Personal Digital Assistants (PDAs). Mr. Jones is the third person that Mr. Smith has watched performing these tasks with this version of the prototype. Mr. Smith is not constantly reminding Mr. Jones to talk while he works, but is counting on his proximity to Mr. Jones to encourage verbal expressions when Mr. Jones encounters any difficulty in accomplishing his current task. Mr. Smith takes written notes whenever this happens, and also takes notes whenever he observes Mr. Jones faltering in his use of the application (for example, exploring menus in search of a desired function). Later that day he will use his notes to develop problem reports and, in consultation with the development team, will work on recommendations for product changes that should eliminate or reduce the impact of the reported problems. When a new version of the prototype is ready, he will resume testing.

Scenario 2: Dr. White is watching Mr. Adams work with a new version of a word processing application. Mr. Adams is working alone in a test cell that looks almost exactly like an office, except for the large mirror on one wall and the two video cameras overhead. He has access to a telephone and a number to call if he encounters a difficulty that he cannot overcome. If he places such a call, Dr. White will answer and provide help modeled on the types of help provided at the company's call centers. Dr. White can see Mr. Adams through the one-way glass as she coordinates the test. She has one assistant working the video cameras for maximum effectiveness and another who is taking time-stamped notes on a computer (coordinated with the video time stamps) as different members of the team notice and describe different aspects of Mr. Adams' task performance. Software monitors Mr. Adams' computer, recording all keystrokes and mouse movements. Later that day, Dr. White and her associates will put together a summary of the task performance measurements for the tested version of the application, noting where the performance measurements do not meet the test criteria. They will also create a prioritized list of problems and

recommendations, along with video clips that illustrate key problems, for presentation to the development team at their weekly status meeting.

Both of these scenarios provide examples of usability testing. In Scenario 1, the emphasis is completely on usability problem discovery and resolution (formative, or diagnostic evaluation). In Scenario 2, the primary emphasis is on task performance measurement (summative, or measurement-focused evaluation), but there is also an effort to record and present usability problems to the product developers. Dr. White's team knows that they cannot determine if they've met the usability performance goals by examining a list of problems, but they also know that they cannot provide appropriate guidance to product development if they only present a list of global task measurements. The problems observed in the use of an application provide important clues for redesigning the product (Chapanis, 1981; Norman, 1983). Furthermore, as John Karat (1997, p. 693) observed, "The identification of usability problems in a prototype user interface (UI) is not the end goal of any evaluation. The end goal is a redesigned system that meets the usability objectives set for the system such that users are able to achieve their goals and are satisfied with the product."

These scenarios also illustrate the defining properties of a usability test. During a usability test, one or more observers watch one or more participants perform specified tasks with the product in a specified test environment (compare this with the ISO/ANSI definition of usability presented earlier in this chapter). This is what makes usability testing different from other User-Centered Design (UCD) methods. In interviews (including the group interview known as a focus group), participants do not perform work-like tasks. Usability inspection methods (such as expert evaluations and heuristic evaluations), also do not include the observation of users or potential users performing work-like tasks. The same is true of techniques such as surveys and card-sorting. Field studies (including contextual inquiry) can involve the observation of users performing work-related tasks in target environments, but restrict the control that practitioners have over the target tasks and environments. Note that this is not necessarily a bad thing, but it is a defining difference between usability testing and field (ethnographic) studies.

This definition of usability testing permits a wide range of variation in technique (Wildman, 1995). Usability tests can be very informal (as in Scenario 1) or very formal (as in Scenario 2). The observer might sit next to the participant, watch through a one-way glass, or watch the on-screen behavior of a participant who is performing specified tasks at a location halfway around the world. Usability tests can be think-aloud (TA) tests, in which observers train participants to talk about what they're doing at each step of task completion and prompt participants to continue talking if they stop. Observers might watch one participant at a time, or might watch participants work in pairs. Practitioners can apply usability testing to the evaluation of low-fidelity prototypes (see Figure 1), Wizard of Oz (WOZ) prototypes (Kelley, 1985), high-fidelity prototypes, products under development, predecessor products, or competitive products.

Figure 1. Practitioner and participant engaging in an informal usability test with a pencil-and-paper prototype. (Photo courtesy of IBM.)

Where Did Usability Testing Come From?

The roots of usability testing lie firmly in the experimental methods of psychology (in particular, cognitive and applied psychology) and human factors engineering, and are strongly tied to the concept of iterative design. In a traditional experiment, the experimenter draws up a careful plan of study that includes the exact number of participants that the experimenter will expose to the different experimental treatments. The participants are members of the population to which the experimenter wants to generalize the results. The experimenter provides instructions and debriefs the participant, but at no time during a traditional experimental session does the experimenter interact with the participant (unless this interaction is part of the experimental treatment). The more formative (diagnostic, focused on problem discovery) the focus of a usability test, the less it is like a traditional experiment (although the requirements for sampling from a legitimate population of users, tasks, and environments still apply). Conversely, the more summative (focused on measurement) a usability test is, the more it should resemble the mechanics of a traditional experiment. Many of the principles of psychological experimentation that exist to protect experimenters from threats to reliability and validity (for example, the control of demand characteristics) carry over into usability testing (Holleran, 1991; Wenger and Spyridakis, 1989).

As far as I can tell, the earliest accounts of iterative usability testing applied to product design came from Alphonse Chapanis and his students (Al-Awar et al., 1981; Chapanis, 1981; Kelley, 1984), with almost immediate influence on product development practices at IBM (Kennedy, 1982; Lewis, 1982) and other companies, notably Xerox (Smith et al., 1982) and Apple (Williams, 1983). Shortly thereafter, John Gould and his associates at the IBM T. J. Watson Research Center began publishing influential papers on

usability testing and iterative design (Gould, 1988; Gould and Boies, 1983; Gould and Lewis, 1984; Gould et al., 1987).

The driving force that separated iterative usability testing from the standard protocols of experimental psychology was the need to modify early product designs as rapidly as possible (as opposed to the scientific goal of developing and testing competing theoretical hypotheses). As Al-Awar et al. (1981) reported, "Although this procedure [iterative usability test, redesign, and retest] may seem unsystematic and unstructured, our experience has been that there is a surprising amount of consistency in what subjects report. Difficulties are not random or whimsical. They do form patterns." (p. 33)

When, during the early stages of iterative design, difficulties of use become apparent, it is hard to justify continuing to ask test participants to perform the test tasks. There are ethical concerns with intentionally frustrating participants who are using a product with known flaws that the design team can and will correct. There are economic concerns with the time wasted by watching participants who are encountering and recovering from known error-producing situations. Furthermore, any delay in updating the product delays the potential discovery of problems associated with the update or problems whose discovery was blocked by the presence of the known flaws. For these reasons, the earlier you are in the design cycle, the more rapidly you should iterate the cycles of test and design.

Is Usability Testing Effective?

The widespread use of usability testing is evidence that practitioners believe that usability testing is effective. Unfortunately, there are fields in which practitioners' belief in the effectiveness of their methods does not appear to be warranted by those outside of the field (for example, the use of projective techniques such as the Rorschach test in psychotherapy, Lilienfeld et al., 2000). In our own field, a number of recently published papers have questioned the reliability of usability problem discovery (Kessner et al., 2001; Molich et al., 1998, 2004).

The common finding in these studies has been that observers (either individually or in teams across usability laboratories) who evaluated the same product produced markedly different sets of discovered problems. Molich et al. (1998) had four independent usability laboratories carry out inexpensive usability tests of a software application for new users. The four teams reported 141 different problems, with only one problem common among all four teams. Molich et al. (1998) attributed this inconsistency to variability in the approaches taken by the teams (task scenarios, level of problem reporting). Kessner et al. (2001) had six professional usability teams independently test an early prototype of a dialog box. None of the problems were detected by every team, and 18 problems were described by one team only. Molich et al. (2004) assessed the consistency of usability testing across nine independent organizations that evaluated the same website. They documented considerable variability in methodologies, resources applied, and problems reported. The total number of reported problems was 310, with only two problems reported by six or more organizations, and 232 problems uniquely reported. "Our main conclusion is that our simple assumption that we are all doing the same and getting the same results in a usability test is plainly wrong" (Molich et al., 2004, p. 65).

This is important and disturbing research, but there is a clear need for much more research in this area. A particularly important goal of future research should be to reconcile these studies with the documented reality of usability improvement achieved through iterative application of usability testing. For example, a limitation of research that stops with the comparison of problem lists is that it is not possible to assess the magnitude of the usability improvement (if any) that would result from product redesigns based on design recommendations derived from the problem lists (Wixon, 2003). When comparing problem lists from many labs, one aberrant set of results can have an extreme effect on measurements of consistency across labs, and the more labs that are involved, the more likely this is to happen.

The results of these studies (Kessner et al., 2001; Molich et al., 1998, 2004) stand in stark contrast to the published studies in which iterative usability tests (sometimes in combination with other UCD methods) have led to significantly improved products (Al-Awar et al., 1981; Bailey, 1993; Bailey et al., 1992;

Gould et al., 1987; Kelley, 1984; Kennedy, 1982; Lewis, 1982; Lewis, 1996b; Ruthford and Ramey, 2000). For example, in a paper describing their experiences in product development, Marshall et al. (1990) stated, "Human factors work can be reliable – different human factors engineers, using different human factors techniques at different stages of a product's development, identified many of the same potential usability defects" (p. 243). Published cost-benefit analyses (Bias and Mayhew, 1994) have demonstrated the value of usability engineering processes that include usability testing, with cost-benefit ratios ranging from 1:2 for smaller projects to 1:100 for larger projects (C. Karat, 1997).

Most of the papers that describe the success of iterative usability testing are case studies (such as Marshall et al., 1990), but a few have described designed experiments. Bailey et al. (1992) compared two user interfaces derived from the same base interface – one modified via heuristic evaluation and the other modified via iterative usability testing (three iterations, five participants per iteration). They conducted this experiment with two interfaces, one character-based and the other a graphical user interface (GUI), with the same basic outcomes. The number of changes indicated by usability testing was much smaller than the number indicated by heuristic evaluation, but user performance was the same with both final versions of the interface. All designs after the first iteration produced faster performance than and, for the character-based interface, were preferred to, the original design. The time to complete the performance testing was about the same as that required for the completion of multi-reviewer heuristic evaluations.

Bailey (1993) provided additional experimental evidence that iterative design based on usability tests leads to measurable improvements in the usability of an application. In the experiment, he studied the designs of eight designers, four with at least four years of professional experience in interface design and four with at least five years of professional experience in computer programming. All designers used a prototyping tool to create a recipes application (eight applications in all). In the first wave of testing, Bailey videotaped participants performing tasks with the prototypes, three different participants per prototype. Each designer reviewed the videotapes of the people using his or her prototype, and used the observations to redesign his or her application. This process continued until each designer indicated that it was not possible to improve his or her application. All designers stopped after three to five iterations. Comparison of the first and last iterations indicated significant improvement in measurements such as number of tasks completed, task completion times, and repeated serious errors.

In conclusion, the results of the studies of Molich et al. (1998, 2004) and similar studies show that usability practitioners must conduct their usability tests as carefully as possible, document their methods completely, and show proper caution when interpreting their results. On the other hand, as Landauer stated in 1997, "There is ample evidence that expanded task analysis and formative evaluation can, and almost always do, bring substantial improvements in the effectiveness and desirability of systems" (p. 204). This is echoed by Desurvire et al. (1992, p. 98), "It is generally agreed that usability testing in both field and laboratory, is far and above the best method for acquiring data on usability."

Goals of Usability Testing

The fundamental goal of usability testing is to help developers produce more usable products. The two conceptions of usability testing (formative and summative) lead to differences in the specification of goals in much the same way that they contribute to differences in fundamental definitions of usability (diagnostic problem discovery and measurement). Rubin (1994, p. 26) expressed the formative goal as, "The overall goal of usability testing is to identify and rectify usability deficiencies existing in computer-based and electronic equipment and their accompanying support materials prior to release." Dumas and Redish (1999, p. 11) provided a more summative goal with, "A key component of usability engineering is setting specific, quantitative, usability goals for the product early in the process and then designing to meet those goals."

These goals are not in direct conflict, but they do suggest different foci that can lead to differences in practice. For example, a focus on measurement typically leads to more formal testing (less interaction between observers and participants) whereas a focus on problem discovery typically leads to less formal testing (more interaction between observers and participants). In addition to the distinction between diagnostic problem discovery and measurement tests, there are two common types of measurement tests: (1) comparison against objectives and (2) comparison of products.

Problem Discovery Test

The primary activity in diagnostic problem discovery tests is the discovery, prioritization, and resolution of usability problems. The number of participants in each iteration of testing should be fairly small, but the overall test plan should be for multiple iterations, each with some variation in participants and tasks. When the focus is on problem discovery and resolution, the assumption is that more global measures of user performance and satisfaction will take care of themselves (Chapanis, 1981). The measurements associated with problem-discovery tests are focused on prioritizing problems, and include frequency of occurrence in the test, likelihood of occurrence during normal usage (taking into account the anticipated usage of the part of the product in which the problem occurred), and magnitude of impact on the participants who experienced the problem. Because the focus is not on precise measurement of the performance or attitudes of participants, problem discovery studies tend to be informal, with a considerable amount of interaction between observers and participants. Some typical stopping rules for iterations are a preplanned number of iterations or a specific problem discovery goal, such as "Identify 90% of the problems available for discovery for these types of participants, this set of tasks, and these conditions of use." See the section below on sample size estimation and adequacy for more detailed information on setting and using these types of problem-discovery objectives.

Measurement Test Type I: Comparison against Quantitative Objectives

Studies that have a primary focus of comparison against quantitative objectives include two fundamental activities. The first is the development of the usability objectives. The second is iterative testing to determine if the product under test has met the objectives. A third activity (which can take place during iterative testing) is the enumeration and description of usability problems, but this activity is secondary to the collection of precise measurements.

The first step in developing quantitative usability objectives is to determine the appropriate variables to measure. Rengger (1991), as part of the work done for the European MUSiC project (Measuring the Usability of Systems in Context) produced a list of potential usability measurements based on 87 papers out of a survey of 500 papers. He excluded purely diagnostic studies and also excluded papers if they did not provide measurements for the combined performance of a user and a system. He categorized the measurements into four classes:

- Class 1: Goal achievement indicators (such as success rate and accuracy)
- Class 2: Work rate indicators (such as speed and efficiency)
- Class 3: Operability indicators (such as error rate and function usage)
- Class 4: Knowledge acquisition indicators (such as learnability and learning rate)

In a later discussion of the MUSiC measures, Macleod et al. (1997) described measures of effectiveness (the level of correctness and completeness of goal achievement in context) and efficiency (effectiveness related to cost of performance – typically the effectiveness measure divided by task completion time). Optional measures were of productive time and unproductive time, with unproductive time consisting of help actions, search actions, and snag (negation, cancelled, or rejected) actions.

Their (Macleod et al., 1997) description of the measures of effectiveness and efficiency seem to have influenced the objectives expressed in ISO 9241-11 (1998, p. iv): "The objective of designing and evaluating visual display terminals for usability is to enable users to achieve goals and meet needs in a particular context of use. ISO 9241-11 explains the benefits of measuring usability in terms of user performance and satisfaction. These are measured by the extent to which the intended goals of use are

achieved, the resources that have to be expended to achieve the intended goals, and the extent to which the user finds the use of the product acceptable."

In practice (and as recommended in the ANSI Common Industry Format for Usability Test Reports, 2001), the fundamental global measurements for usability tasks are successful task completion rates (for a measure of effectiveness), mean task completion times (for a measure of efficiency), and mean participant satisfaction ratings (either collected on a task-by-task basis or at the end of a test session – see the section below on standardized usability questionnaires for more information on measuring participant satisfaction). There are many other measurements that practitioners could consider (Dumas and Redish, 1999; Nielsen, 1997), including but not limited to:

1. The number of tasks completed within a specified time limit
2. The number of wrong menu choices
3. The number of user errors
4. The number of repeated errors (same user committing the same error more than once)

After determining the appropriate measurements, the next step is to set the goals. Ideally, the goals should have an objective basis and shared acceptance across the various stakeholders, such as Marketing, Development, and Test groups (Lewis, 1982). The best objective basis for measurement goals are data from previous usability studies of predecessor or competitive products. For maximum generalizability, the historical data should come from studies of similar types of participants completing the same tasks under the same conditions (Chapanis, 1988). If this information is not available, then an alternative is for the test designer to recommend objective goals and to negotiate with the other stakeholders to arrive at a set of shared goals.

"Defining usability objectives (and standards) isn't easy, especially when you're beginning a usability program. However, you're not restricted to the first objective you set. The important thing is to establish some specific objectives immediately, so that you can measure improvement. If the objectives turn out to be unrealistic or inappropriate, you can revise them." (Rosenbaum, 1989, p. 211) Such revisions, however, should take place only in the early stages of gaining experience and taking initial measurements with a product. It is important not to change reasonable goals to accommodate an unusable product.

When setting usability goals, it's usually better to set goals that make reference to an average (mean) of a measurement than to a percentile. For example, set an objective such as "The mean time to complete Task 1 will be less than five minutes" rather than "95% of participants will complete Task 1 in less than ten minutes". The statistical reason for this is that sample means drawn from a continuous distribution are less variable than sample medians (the 50th percentile of a sample), and measurements made away from the center of a distribution (for example, measurements made to attempt to characterize the value of the 95th percentile) are even more variable (Blalock, 1972). Cordes (1993) conducted a Monte Carlo study comparing means and medians as measurements of central tendency for time-on-task scores, and determined that the mean should be the preferred metric for usability studies (unless there is missing data due to participants failing to complete tasks, in which case the mean from the study will underestimate the population mean).

A practical reason to avoid percentile goals is that the goal can imply a sample size requirement that is unnecessarily large. For example, you can't measure accurately at the 95th percentile unless there are at least twenty measurements (in fact, there must be many more than twenty measurements for accurate measurement). For more details, see the section below on sample size estimation for measurement.

An exception to this is the specification of successful task completions (or any other measurement that is based on counting events), which necessarily requires a percentile goal, usually set at or near 100% (unless there are historical data that indicate an acceptable lower level for a specific test). If ten out of ten participants complete a task successfully, the observed completion rate is 100%, but a 90% binomial confidence interval for this result ranges from 74% to 100%. In other words, even perfect performance for ten participants with this type of measure leaves open the possibility (with 90% confidence) that the true

completion rate could be as low as 75%. See the section below on binomial confidence intervals for more information on computing and using this information in usability tests.

After the usability goals have been established, the next step is to collect data to determine if the product has met its goals. Representative participants perform the target tasks in the specified environment as test observers record the target measurements and identify, to the extent possible within the constraints of a more formal testing protocol, details about any usability problems that occur. The usability team conducting the test provides information about goal achievement and prioritized problems to the development team, and a decision is made regarding whether or not there is sufficient evidence that the product has met its objectives. The ideal stopping rule for measurement-based iterations is to continue testing until the product has met its goals.

When there are only a few goals, then it is reasonable to expect to achieve all of them. When there are many goals (for example, five objectives per task multiplied by ten tasks for a total of fifty objectives), then it is more difficult to determine when to declare success and to stop testing. Thus, it is sometimes necessary to specify a meta-objective of the percentage of goals to achieve.

Despite the reluctance of some usability practitioners to conduct statistical tests to quantitatively assess the strength of the available evidence regarding whether or not a product has achieved a particular goal, the best practice is to conduct such tests. The best approach is to conduct multiple *t*-tests or nonparametric analogs of *t*-tests (Lewis, 1993) because this gives practitioners the level of detail that they require. There is a well-known prohibition against doing this because it can lead investigators to mistakenly accept as real that some differences that are due to chance (technically, alpha inflation). On the other hand, if this is the required level of information, then it is an appropriate method (Abelson, 1995). Furthermore, the practice of avoiding alpha inflation is a concern more related to scientific hypothesis testing than to usability testing (Wickens, 1998), although usability practitioners should be aware of its existence and take it into account when interpreting their statistical results. For example, if you compare two products by conducting fifty *t*-tests with alpha set to .10, and only five (10%) of the *t*-tests are significant (have p less than .10), then you should question whether or not to use those results as evidence of the superiority of one product over the other. On the other hand, if substantially more than five of the *t*-tests are significant, then you can be more confident that the indicated differences are real.

In addition to (or as an alternative to) conducting multiple *t*-tests, practitioners should compute confidence intervals for their measurements. This applies to the measurements made for the purpose of establishing test criteria (such as measurements made on predecessor versions of the target product or competitive products) and to the measurements made when testing the product under development. See the section below on confidence intervals for more details.

Measurement Test Type II: Comparison of Products

The second type of measurement test is to conduct usability tests for the purpose of directly comparing one product with another. As long as there is only one measurement that decision makers plan to consider, then a standard *t*-test (ideally, in combination with the computation of confidence intervals) will suffice for the purpose of determining which product is superior.

If decision makers care about multiple dependent measures, then standard multivariate statistical procedures (such as MANOVA or discriminant analysis) are not often helpful in guiding a decision about which of two products has superior usability. The statistical reason for this is that multivariate statistical procedures depend on the computation of centroids (a weighted average of multiple dependent measures) using a least-squares linear model that maximizes the difference between the centroids of the two products (Cliff, 1987). If the directions of the measurements are inconsistent (for example, a high task completion rate is desirable, but a high mean task completion time is not), then the resulting centroids are uninterpretable for the purpose of usability comparison. In some cases it is possible to recompute variables so they have consistent directions (for example, recomputing task completion rates as task failure rates). If this is not possible, then another approach is to convert measurements to ranks (Lewis, 1991a) or standardized (Z)

scores (Jeff Sauro, personal communication, March 1, 2004) for the purpose of principled combination of different types of measurements.

To help consumers compare the usability of different products, the American National Standards Institute (ANSI) has published the Common Industry Format (CIF) for usability test reports (ANSI, 2001). Originally developed at the National Institute of Standards and Technology (NIST), this test format requires measurement of effectiveness (accuracy and completeness – completion rates, errors, assists), efficiency (resources expended in relation to accuracy and completeness – task completion time), and satisfaction (freedom from discomfort, positive attitude toward use of the product – using any of a number of standardized satisfaction questionnaires). It also requires a complete description of participants and tasks.

Morse (2000) reviewed the NIST IUSR project conducted to pilot test the CIF. The purpose of the CIF is to make it easier for purchasers to compare the usability of different products. The pilot study ran into problems, such as inability to find a suitable software product for both supplier and consumer, reluctance to share information, and uncertainty about how to design a good usability study. To date, there has been little if any use (at least, no published use) of the CIF for its intended purpose.

Variations on a Theme: Other Types of Usability Tests

Think Aloud

In a standard, formal usability test, test participants perform tasks without necessarily speaking as they work. The defining characteristic of a Think Aloud (TA) study is the instruction to participants to talk about what they are doing as they do it (in other words, to produce verbal reports). If participants stop talking (as commonly happens when they become very engaged in a task), they are prompted to resume talking.

The most common theoretical justification for the use of TA is from the work in cognitive psychology (specifically, human problem solving) of Ericsson and Simon (1980). Responding to a review by Nisbett and Wilson (1977) that described various ways in which verbal reports were unreliable, Ericsson and Simon provided evidence that certain kinds of verbal reports could produce reliable data. They stated that reliable verbalizations are those that participants produce during task performance that do not require additional cognitive processing beyond the processing required for task performance and verbalization.

Some discussions of usability testing hold that the best practice in usability testing is to use the TA method in all usability testing. For example, Dumas (2003) encouraged the use of TA because (1) TA tests are more productive for finding usability problems (Virzi, Sorce, and Herbert, 1993) and (2) thinking aloud does not affect user ratings or performance (Bowers and Snyder, 1990). As the references indicate, there is some evidence in support of these statements, but the evidence is mixed.

Earlier prohibitions against the use of TA in measurement-based tests assumed that thinking aloud would cause slower task performance. Bowers and Snyder (1990), however, found no measurable task performance or preference differences between a test group that thought aloud and one that didn't. Surprisingly, there are some experiments in which the investigators reported better task performance when participants were thinking aloud. Berry and Broadbent (1990) provided evidence that the process of thinking aloud invoked cognitive processes that improved rather than degraded performance, but only if people were given (1) verbal instructions on how to perform the task and (2) the requirement to justify each action aloud. Wright and Converse (1992) compared silent with TA usability testing protocols. The results indicated that the think-aloud group committed fewer errors and completed tasks faster than the silent group, and the difference between the groups increased as a function of task difficulty.

Regarding the theoretical justification for and typical practice of TA, Boren and Ramey (2000) noted that TA practice in usability testing often does not conform to the theoretical basis most often cited for it (Ericsson and Simon, 1980). "If practitioners do not uniformly apply the same techniques in conducing thinking-aloud protocols, it becomes difficult to compare results between studies." (Boren and Ramey, 2000,

p. 261) In a review of publications of TA tests and field observations of practitioners running TA tests, they reported inconsistency in explanations to participants about how to think aloud, practice periods, styles of reminding participants to think aloud, prompting intervals, and styles of intervention. They suggest that rather than basing current practice on Ericsson and Simon, a better basis would be speech communication theory, with clearly defined communicative roles for the participant (in the role of domain expert or valued customer, making the participant the primary speaker) and the usability practitioner (the learner or listener, thus, a secondary speaker).

Based on this alternative perspective for the justification of TA, Boren and Ramey (2000) have provided guidance for many situations that are not relevant in a cognitive psychology experiment, but are in usability tests. For example, they recommend that usability practitioners running a TA test should continually use acknowledgement tokens that do not take speakership away from the participant, such as "mm hm?" and "uh-huh?" (with the interrogative intonation) to encourage the participant to keep talking. In normal communication, silence (as recommended by the Ericsson and Simon protocols) is not a nonresponse – the speaker interprets it in a primarily negative way as indicating aloofness or condescension. They avoided providing precise statements about how frequently to provide acknowledgments or somewhat more explicit reminders (such as "And now…?") because the best cues come from the participants. Practitioners need to be sensitive to these cues as they run the test.

The evidence indicates that, relative to silent participation, TA can affect task performance. If the primary purpose of the test is problem discovery, then TA appears to have advantages over completely silent task completion. If the primary purpose of the test is task performance measurement, then the use of TA is somewhat more complicated. As long as all the tasks in the planned comparisons were completed under the same conditions, then performance comparisons should be legitimate. The use of TA almost certainly prevents generalization of task performance outside of the TA task, but there are many other factors that make it difficult to generalize specific task performance data collected in usability studies.

For example, Cordes (2001) demonstrated that participants assume that the tasks they are asked to perform in usability tests are possible (the "I know it can be done or you wouldn't have asked me to do it" bias). Manipulations that bring this assumption into doubt can have a strong effect on quantitative usability performance measures, such as increasing the percentage of participants who give up on a task. If uncontrolled, this bias makes performance measures from usability studies unlikely to be representative of real-world performance when users are uncertain as to whether the product they are using can support the desired tasks.

Multiple Simultaneous Participants

Another way to encourage participants to talk during task completion is to have them work together (Wildman, 1995). This strategy is similar to TA in its strengths and limitations.

Hackman and Biers (1992) compared three think-aloud methods: thinking aloud alone (Single), thinking aloud in the presence of an observer (Observer), and verbalizations occurring in a two-person team (Team). They found no significant differences in performance or subjective measures. The Team condition produced more statements of value to designers than the other two conditions, but this was probably due to the differing number of participants producing statements in the different conditions. There were three groups, with 10 participants per group for Single and Observer, and 20 participants (10 two-person teams) for the Team condition. "The major result was that the team gave significantly more verbalizations of high value to designers and spent more time making high value comments. Although this can be reduced to the fact that the team spoke more overall and that there are two people talking rather than one, this finding is not trivial." (p. 1208)

Remote Evaluation

Recent advances in the technology of collaborative software have made it easier to conduct remote software tests (tests in which the usability practitioner and the test participant are in different locations). This can be an economical alternative to bringing one or more users into a lab for face-to-face user testing.

A participant in a remote location can view the contents of the practitioner's screen, and in a typical system the practitioner can decide whether the participant can control the desktop. System performance is typically slower than that of a local test session.

Some of the advantages of remote testing are (1) access to participants who would otherwise be unable to participate (international, special needs, etc.), (2) the capability for participants to work in familiar surroundings, and (3) no need for either party to install or download additional software. Some of the disadvantages are (1) potential uncontrolled disruptions in the participant's workplace, (2) lack of visual feedback from the participant, and (3) the possibility of compromised security if the participant takes screen captures of confidential material. Despite these disadvantages, McFadden et al. (2002) reported data that indicated that remote testing was effective at improving product designs and that the test results were comparable to the results obtained with more traditional testing.

Usability Laboratories

A typical usability laboratory test suite is a set of soundproofed rooms with a participant area and observer area separated by a one-way glass, and with video cameras and microphones to capture the user experience (Marshall et al., 1990; Nielsen, 1997), possibly with an executive viewing area behind the primary observers' area. The advantages of this type of usability facility are quick setup, a place where designers can see people interacting with their products, videos to provide a historical record and backup for observers, and a professional appearance that raises awareness of usability and reassures customers about commitment to usability. In a survey of usability laboratories, Nielsen (1994) reported a median floor space of 63 m^2 (678 ft^2) for the observer room and 13 m^2 (144 ft^2) for test rooms. This type of laboratory (see Figure 2) is especially important if practitioners plan to conduct formal, summative usability tests.

Figure 2. View of a usability laboratory. (Photo courtesy of IBM.)

If the practitioner focus is on formative, diagnostic problem discovery, then this type of laboratory is not essential (although still convenient). "It is possible to convert a regular office temporarily into a usability laboratory, and it is possible to perform usability testing with no more equipment than a notepad." (Nielsen, 1997, p. 1561) Making an even stronger statement against the perceived requirement for formal laboratories, Landauer (1997, p. 204) stated, "Many usability practitioners have demanded greater resources and more elaborate procedures than are strictly needed for effective guidance – such as expensive usability labs rather than natural settings for test and observations, time consuming videotaping and analysis where observation and note-taking would serve as well, and large groups of participants to achieve statistical significance when qualitative naturalistic observation of task goals and situations, or of disastrous interface or functionality flaws, would be more to the point."

Test Roles

There are several ways to categorize the roles that testers need to play in the preparation and execution of a usability test (Dumas and Redish, 1999; Rubin, 1994). Most test teams will not have an individual assigned to each role, and most tests (especially informal problem discovery tests) do not require every role. The actual distribution of skills across a team might vary from these roles, but the standard roles help to organize the skills needed for effective usability testing.

Test Administrator

The test administrator is the usability test team leader. He or she designs the usability study, including the specification of the initial conditions for a test session and the codes to use for data logging. The test administrator's duties include conducting reviews with the rest of the test team, leading in the analysis of data, and putting together the final presentation or report. People in this role should have a solid understanding of the basics of usability engineering, ability to tolerate ambiguity, flexibility (knowing when to deviate from the plan), and good communication skills.

Briefer

The briefer is the person who interacts with the participants (briefing them at the start of the test, communicating with them as required during the test, and debriefing them at the end of the test sessions). On many teams, the same person takes the roles of administrator and briefer. In a think-aloud study, the briefer has the responsibility to keep the participant talking. The briefer needs to have sufficient familiarity with the product to be able to decide what to tell participants when they ask questions. People in this role need to be comfortable interacting with people, and need to be able to restrict their interactions to those that are consistent with the purposes of the test without any negative treatment of the participants.

Camera Operator

The camera operator is responsible for running the audio-visual equipment during the test. He or she must be skilled in the setup and operation of the equipment, and must be able to take directions quickly when it is necessary to change the focus of the camera (for example, from the keyboard to the user manual).

Data Recorder

The video record is useful as a data backup when things start happening quickly during the test, and as a source for video examples when documenting usability problems. The primary data source for a usability study, however, is the notes that the data recorder takes during a test session. There just isn't time to take notes from a more leisurely examination of the video record. Also, the camera doesn't necessarily catch the important action at every moment of a usability study.

For informal studies, the equipment used to record data might be nothing more than a notepad and pencil. Alternatively, the data recorder might use data-logging software to take coded notes (often time-stamped, possibly synchronized with the video). Before the test begins, the data recorder needs to prepare the data-logging software with the category codes defined by the test administrator. Taking notes with data-logging software is a very demanding skill, so the test administrator does not usually assign additional tasks to the person taking this role.

Help Desk Operator

The help desk operator takes calls from the participant if the user experiences enough difficulty to place the call. The operator should have some familiarity with the call-center procedures followed by the company that has designed the product under test, and must also have skills similar to those of the briefer.

Product Expert

The product expert maintains the product and offers technical guidance during the test. The product expert must have sufficient knowledge of the product to recover quickly from product failures and to help the other team members understand the system's actions during the test.

Statistician

A statistician has expertise in measurement and the statistical analysis of data. Practitioners with an educational background in experimental psychology typically have sufficient expertise to take the role of statistician for a usability test team. Informal tests rarely require the services of a statistician, but the team needs a statistician to extract the maximum amount of information from the data gathered during a formal test (especially if the purpose of the formal test was to compare two products using a battery of measurements).

Planning the Test

One of the first activities a test administrator must undertake is to develop a test plan. To do this, the administrator must understand the purpose of the product, the parts of the product that are ready for test, the types of people who will use the product, what they are likely to use the product for, and in what settings.

Purpose of Test

At the highest level, is the primary purpose of the test to identify usability problems or to gather usability measurements? The answer to this question provides guidance as to whether the most appropriate test is formal or informal, think-aloud or silent, problem discovery or quantitative measurement. After addressing this question, the next task is to define any more specific test objectives. For example, an objective for an interactive voice response system (IVR) might be to assess whether participants can accomplish key tasks without encountering significant problems. If data are available from a previous study of a similar IVR, an alternative objective might be to determine whether participants can complete key tasks reliably faster with the new IVR than they did with the previous IVR. Most usability tests will include several objectives.

If a key objective of the test is to compare two products, then an important decision is whether the test will be within-subjects or between-subjects. In a within-subjects test, every participant works with both products, with half of the participants using one product first and the other half using the other product first (a technique known as counterbalancing). In a between-subjects study, the test groups are completely independent. In general, a within-subjects test leads to more precise measurement of product differences (requiring a smaller number of participants for equal precision, primarily due to the reduction in variability that occurs when each participant acts as his or her own control) and the opportunity to get direct subjective product comparisons from participants. For a within-subjects test to be feasible, both products must be available and set up for use in the lab at the same time, and the amount of time needed to complete tasks with both products must not be excessive. If a within-subjects test is not possible, a between-subjects test is a perfectly valid alternative. Note that the statistical analyses appropriate for these two types of tests are different.

Participants

To determine who will participate in the test, the administrator needs to obtain or develop a user profile. A user profile is sometimes available from the marketing group, the product's functional specification, or other product planning documentation. It is important to keep in mind that the focus of a usability test is the end user of a product, not the expected product purchaser (unless the product will be purchased by end users). The most important participant characteristic is that the participant is representative of the population of end-users to whom the administrator wants to generalize the results of the test. Practitioners can obtain participants from employment agencies, internal sources if the participants meet the requirements of the user profile (but avoiding internal test groups), market research firms, existing customers, colleges, newspaper ads, and user groups.

To define representativeness, it is important to specify the characteristics that members of the target population share but are not characteristic of nonmembers. The administrator must do this for the target population at large and any defined subgroups. Within group definition constraints, administrators should seek heterogeneity in the final sample to maximize the generalizability of the results (Chapanis, 1988; Landauer, 1997) and to maximize the likelihood of problem discovery. It is true that performance measurements made with a homogeneous sample will almost always have greater precision than measurements made with a heterogeneous sample, but the cost of that increased precision is limited generalizability. This raises the issue of how to define homogeneity and heterogeneity of participants. After all, at the highest level of categorization, we are all humans, with similar general capabilities and limitations (physical and cognitive). At the other end of the spectrum, we are all individuals – no two alike.

One of the most important defining characteristics for a group in a usability test is specific relevant experience, both with the product and in the domain of interest (work experience, general product experience, specific product experience, experience with the product under test, and experience with similar products). One common categorization scheme is to consider people with less than three months experience as novices, with more than a year of experience as expert, and those in between as intermediate (Dumas and Redish, 1999). Other individual differences that practitioners routinely track and attempt to vary are education level, age, and sex.

When acquiring participants, how can practitioners define the similarity between the participants they can acquire and the target population? An initial step is to develop a taxonomy of the variables that affect human performance (where performance should include the behaviors of indicating preference and other choice behaviors). Gawron et al. (1989) produced a human performance taxonomy during the development of a human performance expert system. They reviewed existing taxonomies and filled in some missing pieces. They structured the taxonomy as having three top levels: environment, subject (person), and task. The resulting taxonomy took up 12 pages in their paper, and covered many areas which would normally not concern a usability practitioner working in the field of computer system usability (for example, ambient vapor pressure, gravity, acceleration, etc.). Some of the key human variables in the Gawron et al. (1989) taxonomy that could affect human performance with computer systems are:

- Physical Characteristics
  - Age
  - Agility
  - Handedness
  - Voice
  - Fatigue
  - Gender
  - Body and body part size
- Mental State
  - Attention span
  - Use of drugs (both prescription and illicit)
  - Long-term memory (includes previous experience)
  - Short-term memory
  - Personality traits
  - Work schedule
- Senses
  - Auditory acuity
  - Tone perception
  - Tactual
  - Visual accommodation
  - Visual acuity
  - Color perception

These variables can guide practitioners as they attempt to describe how participants and target populations are similar or different. The Gawron et al. (1989) taxonomy does not provide much detail with regard to some individual differences that other researchers have hypothesized to affect human performance or preference with respect to the use of computer systems: personality traits and computer-specific experience.

Aykin and Aykin (1991) performed a comprehensive review of the published studies to that date that involved individual differences in human-computer interaction (HCI). Table 1 lists the individual differences that they found in published HCI studies, the method used to measure the individual difference, and whether there was any indication from the literature that manipulation of that individual difference led to a crossed interaction.

In statistical terminology, an interaction occurs whenever an experimental treatment has a different magnitude of effect depending on the level of a different, independent experimental treatment. A crossed interaction occurs when the magnitudes have different signs, indicating reversed directions of effects. As an example of an uncrossed interaction, consider the effect of turning off the lights on the typing throughput of blind and sighted typists. The performance of the sighted typists would probably be worse, but the presence or absence of light shouldn't affect the performance of the blind typists. As an extreme example of a crossed interaction, consider the effect of language on task completion for people fluent only in French or English. When reading French text, French speakers would outperform English speakers, and vice versa.

Table 1. Results of Aykin and Aykin (1991) review of individual differences in HCI

Individual Difference	**Measurement Method**	**Crossed Interactions**
Level of experience	Various methods	No
Jungian personality types	Myers-Briggs Type Indicator	No
Field dependence/ independence	Embedded Figures Test	Yes – field dependent participants preferred organized sequential item number search mode, but field independent subjects preferred the less organized keyword search mode (Fowler et al., 1985)
Locus of control	Levenson test	No
Imagery	Individual Differences Questionnaire	No
Spatial ability	VZ-2	No
Type A/Type B personality	Jenkins Activity Survey	No
Ambiguity tolerance	Ambiguity Tolerance Scale	No
Sex	Unspecified	No
Age	Unspecified	No
Other (reading speed and comprehension, intelligence, mathematical ability)	Unspecified	No

For any of these individual differences, the lack of evidence for crossed interactions could be due to a paucity of research involving the individual difference or could reflect the probability that individual differences will not typically cause crossed interactions in HCI. In general, a change made to support a problem experienced by a person with a particular individual difference will either help other users or simply not affect their performance.

For example, John Black (personal communication, 1988) cited the difficulty that field dependent users had working with one-line editors at the time (decades ago) when that was the typical user interface to a mainframe computer. Switching to full-screen editing resulted in a performance improvement for both field dependent and independent users – an uncrossed interaction because both types of users improved, with the performance of field dependent users becoming equal to (thus improving more than) that of field independent users. Landauer (1997) cites another example of this, in which Greene et al. (1986) found that young people with high scores on logical reasoning tests could master database query languages such as SQL with little training, but older or less able people could hardly ever master these languages. They also determined that an alternative way of forming queries, selecting rows from a truth table, allowed almost everyone to make correct specification of queries, independent of their abilities. Because this redesign improved the performance of less able users without diminishing the performance of the more able, it was an uncrossed interaction. In a more recent study, Palmquist and Kim (2000) found that field dependence

affected the search performance of novices using a web browser (with field independent users searching more efficiently), but did not affect the performance of more experienced users.

If there is a reason to suspect that an individual difference will lead to a crossed interaction as a function of interface design, then it could make sense to invest the time (which can be considerable) to categorize users according to these dimensions. Another situation in which it could make sense to invest the time in categorization by individual difference would be if there were reasons to believe that a change in interface would greatly help one or more groups without adversely affecting other groups. (This is a strategy that one can employ when developing hypotheses about ways to improve user interfaces.) It always makes sense to keep track of user characteristics when categorization is easy (for example, age or sex). Another potential use of these types of variables is as covariates (used to reduce estimates of variability) in advanced statistical analyses (Cliff, 1987).

Aykin and Aykin (1991) reported effects of users' levels of experience, but did not report any crossed interactions related to this individual difference. They did report that interface differences tended to affect the performance of novices, but had little effect on the performance of experts. It appears that behavioral differences related to user interfaces (Aykin and Aykin, 1991) and cognitive style (Palmquist and Kim, 2000) tend to fade with practice. Nonetheless, user experience has been one of the few individual differences to receive considerable attention in HCI research (Fisher, 1991; Mayer, 1997; Miller et al., 1997; Smith et al., 1999).

According to Mayer (1997), relative to novices, experts have:

- better knowledge of syntax
- an integrated conceptual model of the system
- more categories for more types of routines
- higher level plans.

Fisher (1991) emphasized the importance of discriminating between computer experience (which he placed on a novice-experienced dimension) and domain expertise (which he placed on a naïve-expert dimension). LaLomia and Sidowski (1990) reviewed the scales and questionnaires developed to assess computer satisfaction, literacy and aptitudes. None of the instruments they surveyed specifically addressed measurement of computer experience. Miller et al. (1997) published the Windows Computer Experience Questionnaire (WCEQ), an instrument specifically designed to measure a person's experience with Windows 3.1. The questionnaire took about five minutes to complete and was reliable (coefficient alpha = .74; test-retest correlation = .97). They found that their questionnaire was sensitive to three experiential factors: general Windows experience, advanced Windows experience, and instruction. Smith et al. (1999) distinguished between subjective and objective computer experience. The paper was relatively theoretical and "challenges researchers to devise a reliable and valid measure" (p. 239) for subjective computer experience, but did not offer one.

One user characteristic not addressed in any of the cited literature is one that becomes very important when designing products for international use – cultural characteristics. For example, it is extremely important that in adapting an interface for use by members of another country that all text is accurately translated. It is also important to be sensitive to the possibility that these types of individual differences might be more likely than others to result in crossed interactions.

For comparison studies, having multiple groups (for example, males and females or experts and novices) allows the assessment of potential interactions that might otherwise go unnoticed. Ultimately, the decision for one or multiple groups must be based on expert judgment and a few guidelines. For example, practitioners should consider sampling from different groups if they have reason to believe:

- There are potential and important differences among groups on key measures (Dickens, 1987)
- There are potential interactions as a function of group (Aykin and Aykin, 1991)

- The variability of key measures differs as a function of group
- The cost of sampling differs significantly from group to group

Gordon and Langmaid (1988) recommended the following approach to defining groups:

1. Write down all the important variables.
2. If necessary, prioritize the list.
3. Design an ideal sample.
4. Apply common sense to collapse cells.

For example, suppose a practitioner starts with 24 cells, based on the factorial combination of six demographic locations, two levels of experience, and the two levels of gender. The practitioner should ask himself or herself whether there is a high likelihood of learning anything new and important after completing the first few cells, or would additional testing be wasteful? Can one learn just as much from having one or a few cells that are homogeneous within cells and heterogeneous between cells with respect to an important variable, but are heterogeneous within cells with regard to other, less important variables? For example, a practitioner might plan to (1) include equal numbers of males and females over and under 40 years of age in each cell, (2) have separate cells for novice and experienced users, and (3) drop intermediate users from the test. The resulting design requires testing only two cells (groups), but a design that did not combine genders and age groups in the cells would have required eight cells.

The final issue is the number of participants to include in the test. According to Dumas and Redish (1999), typical usability tests have 6 to 12 participants divided among two to three subgroups. For any given test, the required sample size depends on the number of subgroups, available resources (time/money), and the purpose of the test (for example, precise measurement versus problem discovery). It also depends on whether a study is single-shot (needing a larger sample size) or iterative (needing a smaller sample size per iteration, building up the total sample size over iterations). For a more detailed treatment of this topic, see the section below on sample size estimation.

Test Task Scenarios

As with participants, the most important consideration for test tasks is that they are representative of the types of tasks real users will perform with the product. For any product, there will be a core set of tasks that anyone using the product will perform. People who use barbecue grills use them to cook. People who use desktop speech dictation products use them to produce text. For usability tests, these are the most important tasks to test.

After defining these core tasks, the next step is to list any more peripheral tasks that the test should cover. If a barbecue grill has an external burner for heating pans, it might make sense to include a task that requires participants to work with that burner. If, in addition to the basic vocabulary in a speech dictation system, the program allows users to enable additional special topic vocabularies such as cooking or sports, then it might make sense to devise a task that requires participants to activate and use one of these topics. Practitioners should avoid frivolous or humorous tasks because what is humorous to one person might be offensive to another.

From the list of test tasks, create scenarios of use (with specific goals) that require participants to perform the identified tasks. Critical tasks can appear in more than one scenario. For repeated tasks, vary the task details to increase the generalizability of the results. When testing relatively complex systems, some scenarios should stay within specific parts of the system (for example, typing and formatting a document) and others should explore usage across different parts of the system (for example, creating a figure using a spreadsheet program, adding it to the document, attaching the document to a note, and sending it to a specified recipient).

The complete specification of a scenario should include several items. It is important to document (but not to share with the participant), the required initial conditions so it will be easy to determine before a

test session starts that the system is ready. The written description of the scenario (presented to the participant) should state what the participant is trying to achieve and why (the motivation), keeping the description of the scenario as short as possible to keep the test session moving quickly. The scenario should end with an instruction for the action the participant should take upon finishing the task (to make it easier to measure task completion times). The descriptions of the scenario's tasks should not typically provide step-by-step instructions on how to complete the task, but should include details (for example, actual names and data) rather than general statements. The order in which participants complete scenarios should reflect the way in which users would typically work and with the importance of the scenario, with important scenarios done first unless there are other less important scenarios that produce outputs that the important scenario requires as an initial condition. Not all participants need to receive the same scenarios, especially if there are different groups under study. The tasks performed by administrators of a web system that manages subscriptions will be different from the tasks performed by users who are requesting subscriptions.

Here are some examples of scenarios:

Example 1: "Frank Smith's business telephone number has changed to (896) 555-1234. Please change the appropriate address book entry so you have this new phone number available when you need it. When you have finished, please say 'I'm done.'"

Example 2: "You've just found out that you need to cancel a car reservation that you made for next Wednesday. Please call the system that you used to make the reservation (1-888-555-1234) and cancel it. When you have finished, please hang up the phone and say 'I'm done.'"

Procedure

The test plan should include a description of the procedures to follow when conducting a test session. Most test sessions include an introduction, task performance, post-task activities, and debriefing.

A common structure for the introduction is for the briefer (see the section above on testing roles) to start with the purpose of the test, emphasizing that its goal is to improve the product, not to test the participant. Participation is voluntary, and the participant can stop at any time without penalty. The briefer should inform the participant that all test results will be confidential. The participant should be aware of any planned audio or video recording. Finally, the briefer should provide any special instructions (for example, think-aloud instructions) and answer any other questions that the participant might have.

The participant should then complete any preliminary questionnaires and forms, such as a background questionnaire, an informed consent form (including consent for any recording, if applicable), and, if necessary, a confidential disclosure form. If the participant will be using a workstation, the briefer should help the participant make any necessary adjustments (unless, of course, the purpose of the test is to evaluate workstation adjustability). Finally, the participant should complete any prerequisite training. This can be especially important if the goal of the study is to investigate usability after some period of use (ease of use) rather than immediate usability (ease of learning).

The procedure section should indicate the order in which participants will complete task scenarios. For each participant, start with the first assigned task scenario and complete additional scenarios until the participant finishes (or runs out of time). The procedure section should specify when and how to interact with participants, according to the type of study. This section should also indicate when it is permissible to provide assistance to participants if they encounter difficulties in task performance.

Normally, practitioners should avoid offering assistance unless the participant is visibly distressed. When participants initially request help at a given step in a task, refer them to documentation or other supporting materials if available. If that doesn't help, then provide the minimal assistance required to keep the participant moving forward in the task, note the assistance, and score the task as failed. When participants ask questions, try to avoid direct answers, instead turning their attention back to the task and

encouraging them to take whatever action seems right at that time. When asking questions of participants, it is important to avoid biasing the participant's response. Try to avoid the use of loaded adjectives and adverbs in questions (Dumas and Redish, 1999). Instead of asking if a task was easy, ask the participant to describe what it was like performing the task. Give a short satisfaction questionnaire (such as the ASQ – see the section on questionnaires for details) at the end of each scenario.

After participants have finished the assigned scenarios, it is common to have them complete a final questionnaire, usually a standard questionnaire and any additional items required to cover other test- or product-specific issues. For standardized questionnaires, ISO lists the SUMI (Software Usability Measurement Inventory – Kirakowski, 1996; Kirakowski and Corbett, 1993) and PSSUQ (Post-Study System Usability Questionnaire – Lewis, 1995, 2002). In addition to the SUMI and PSSUQ, ANSI lists the QUIS (Questionnaire for User Interaction Satisfaction – Chin et al., 1988) and SUS (System Usability Scale – Brooke, 1996) as widely used questionnaires. After completing the final questionnaire, the briefer should debrief the participant. Toward the end of debriefing, the briefer should tell the participant that the test session has turned up several opportunities for product improvement (this is almost always true), and thank the participant for his or her contribution to product improvement. Finally, the briefer should discuss any questions the participant has about the test session, and then take care of any remaining activities, such as completing time cards. If there has been any deception employed in the test (which is rare, but can legitimately happen when conducting certain types of simulations), the briefer must inform the participant.

Pilot Testing

Practitioners should always plan for a pilot test before running a usability test. A usability test is a designed artifact, and like any other designed artifact needs at least some usability testing to find problems in the test procedures and materials. A common strategy is to have an initial walkthrough with a member of the usability test team or some other convenient participant. After making the appropriate adjustments, the next pilot participant should be a more representative participant. If there are no changes made to the design of the usability test after running this participant, then the second pilot participant can become the first real participant (but this is rare). Pilot testing should continue until the test procedures and materials have become stable.

Number of Iterations

It is better to run one usability test than not to run any at all. On the other hand, "usability testing is most powerful and most effective when implemented as part of an iterative product development process" (Rubin, 1994, p. 30). Ideally, usability testing should begin early and occur repeatedly throughout the development cycle. When development cycles are short, it is a common practice to run, at a minimum, exploratory usability tests on prototypes at the beginning of a project, to run a usability test on an early version of the product during the later part of functional testing, and then to run another during system testing. Once the final version of the product is available, some organizations run an additional usability test focused on the measurement of usability performance benchmarks. At this stage of development, it is too late to apply information about any problems discovered during the usability test to the soon-to-be-released version of the product, but the information can be useful as early input to a follow-on product if the organization plans to develop another version of the product.

Ethical Treatment of Test Participants

Usability testing always involves human participants, so usability practitioners must be aware of professional practices in the ethical treatment of test participants. Practitioners with professional education in experimental psychology are usually familiar with the guidelines of the American Psychology Association (APA, see http://www.apa.org/ethics/), and those with training in human factors engineering are usually familiar with the guidelines of the Human Factors and Ergonomics Society (HFES, see http://www.hfes.org/About/Code.html). It is particularly important (Dumas, 2003) to be aware of the concepts of informed consent (participants are aware of what will happen during the test, agree to participate, and can leave the test at any time without penalty) and minimal risk (participating in the test does not place participants at any greater risk of harm or discomfort than situations normally encountered in daily life). Most usability tests are consistent with guidelines for informed consent and minimal risk. Only

the test administrator should be able to match a participant's name and data, and the names of test participants should be confidential. Anyone interacting with a participant in a usability test has a responsibility to treat the participant with respect.

Usability practitioners rarely use deception in usability tests. One technique in which there is potential use of deception is the Wizard of Oz (WOZ) method (originally, the OZ Paradigm, Kelley, 1985, also see http://www.musicman.net/oz.html). In a test using the WOZ method, a human (the Wizard) plays the part of the system, remotely controlling what the participant sees happen in response to the participant's manipulations. This method is particularly effective in early tests of speech recognition interactive voice response (IVR) systems because all the Wizard needs is a script and a phone (Sadowski, 2001). Often, there is no compelling reason to deceive participants, so they know that the system they are working with is remotely controlled by another person for the purpose of early evaluation. If there is a compelling need for deception (for example, to manage the participant's expectations and encourage natural behaviors), then this deception must be revealed to the participant during debriefing.

Reporting Results

There are two broad classes of usability test results, problem reports and quantitative measurements. It is possible for a test report to contain one type exclusively (for example, the ANSI Common Industry Format has no provision for reporting problems), but most usability test reports will contain both types of results.

Describing Usability Problems

"We broadly define a usability defect as: Anything in the product that prevents a target user from achieving a target task with reasonable effort and within a reasonable time. … Finding usability problems is relatively easy. However, it is much harder to agree on their importance, their causes and the changes that should be made to eliminate them (the fixes)." (Marshall et al., 1990, p. 245)

The best way to describe usability problems depends on the purpose of the descriptions. For usability practitioners, the goal should be to describe problems in such a way that the description leads logically to one or more potential interventions (recommendations). Ideally, the problem description should also include some indication of the importance of fixing the problem (most often referred to as problem severity). For more scientific investigations, there can be value in higher levels of problem description (Keenan et al., 1999), but developers rarely care about these levels of description. They just want to know what they need to do to make things better while also managing the cost (both monetary and time) of interventions (Gray and Salzman, 1998).

The problem description scheme of Lewis and Norman (1986) has both scientific and practical merit because their problem description categories indicate, at least roughly, an appropriate intervention. They stated (p. 413) that "although we do not believe it possible to design systems in which people do not make errors, we do believe that much can be done to minimize the incidence of error, to maximize the discovery of the error, and to make it easier to recover from the error." They separated errors into mistakes (errors due to incorrect intention) and slips (errors due to appropriate intention but incorrect action), further breaking slips down into mode errors (which indicate a need for better feedback or elimination of the mode), capture errors (which indicate a need for better feedback), and description errors (which indicate a need for better design consistency). In one study using this type of problem categorization, Prümper et al. (1992) found that expertise did not affect the raw number of errors made by participants in their study, but experts handled errors much more quickly than novices. The types of errors that experts made were different from those made by novices, with experts' errors primarily occurring at the level of slips rather than mistakes (knowledge errors).

Rasmussen (1986), using an approach similar to that of Lewis and Norman (1986), described three levels of errors: skill-based, rule-based, and knowledge-based. Two relatively new classification schemes are Structured Usability Problem EXtraction, or SUPEX (Cockton and Lavery, 1999) and the User Action

Framework, or UAF (Andre et al., 2000). The UAF requires a series of decisions, starting with an Interaction Cycle (Planning, Physical Actions, Assessment) based on the work of Norman (1986). Most classifications require four or five decisions, with inter-rater reliability (as measured with kappa) highest at the first step (κ=.978), but remaining high through the fourth and fifth steps (κ>.7).

Whether any of these classification schemes will see widespread use by usability practitioners is still unknown. There is considerable pressure on practitioners to produce results and recommendations as quickly as possible. Even if these classification schemes see little use by practitioners, effective problem classification is a very important problem to solve as usability researchers strive to compare and improve usability testing methods.

Crafting Design Recommendations from Problem Descriptions

As indicated by the title of this section, the development of recommendations from problem descriptions is a craft rather than a rote procedure. A well-written problem description will often strongly imply an intervention, but it is also often the case that there might be several ways to attack a problem. It can be helpful for practitioners to discuss problems and potential interventions with the other members of their team, and to get input from other stakeholders as necessary (especially, the developers of the product). This is especially important if the practitioner has observed problems but is uncertain as to the appropriate level of description of the problem.

For example, suppose you have written a problem description about a missing Help button in a software application. This could be a problem with the overall design of the software, or might be a problem isolated to one screen. You might be able to determine this by inspecting other screens in the software, but it could be faster to check with one of the developers.

The first recommendations to consider should be for interventions that will have the widest impact on the product. "Global changes affect everything and need to be considered first." (Rubin, 1994, p. 285) After addressing global problems, continue working through the problem list until there is at least one recommendation for each problem. For each problem, start with interventions that would eliminate the problem, then follow, if necessary, with other less drastic (less expensive, more likely to be implemented) interventions that would reduce the severity of the remaining usability problem. When different interventions involve different tradeoffs, it is important to communicate this clearly in the recommendations. This approach can lead to two tiers of recommendations – those that will happen for the version of the product currently under development (short-term) and those that will happen for a future version of the product (long-term).

Prioritizing Problems

Because usability tests can reveal more problems than there are resources to address, it is important to have some means for prioritizing problems. There are two approaches to prioritization that have appeared in the usability testing literature: (1) judgment driven (Virzi, 1992) and (2) data driven (Dumas and Redish, 1999; Lewis et al., 1990; Rubin, 1994). The bases for judgment-driven prioritizations are the ratings of stakeholders in the project (such as usability practitioners and developers). The bases for data-driven prioritizations are the data associated with the problems, such as frequency, impact, ease of correction, and likelihood of usage of the portion of the product that was in use when the problem occurred. Of these, the most common measurements are frequency and impact (sometimes referred to as severity, although, strictly speaking, severity should include the effect of all of the types of data considered for prioritization). Hassenzahl (2000), in a study of the two approaches to prioritization, found a lack of correspondence between data-driven and judgment-driven severity estimates. This suggests that the preferred approach should be data-driven.

The usual method for measuring the frequency of occurrence of a problem is to divide the number of occurrences within participants by the number of participants. A common method (Dumas and Redish, 1999; Rubin, 1994) for assessing the impact of a problem is to assign impact scores according to whether the problem (1) prevents task completion, (2) causes a significant delay or frustration, (3) has a relatively minor

effect on task performance, or (4) is a suggestion. This is similar to the scheme of Lewis et al. (1990), in which the impact levels were (1) scenario failure or irretrievable data loss (for example, the participant required assistance to get past the problem or caused the participant to believe the scenario to be properly completed when it wasn't), (2) considerable recovery effort (recovery took more than one minute or the participant repeatedly experienced the problem within a scenario), (3) minor recovery effort (the problem occurred only once within a scenario with recovery time at or under one minute), or (4) inefficiency (a problem not meeting any of the other criteria).

When considering multiple types of data in a prioritization process, it is necessary to combine the data in some way. A graphical approach is to create a problem grid with frequency on one axis and impact on the other (see Figure 3). High-frequency high-impact problems would receive treatment before low-frequency low-impact problems. The relative treatment of high-frequency low-impact problems and low-frequency high-impact problems depends on practitioner judgment.

An alternative approach is to combine the data arithmetically. Rubin (1994) described a procedure for combining four levels of impact (using the criteria described above with 4 assigned to the most serious level) with four levels of frequency (4: frequency = 90%; 3: 51-89%; 2: 11-50%; 1: = 10%) by adding the scores. For example, if a problem had an observed frequency of occurrence of 80% and had a minor effect on performance, then its priority would be 5 (a frequency rating of 3 plus an impact rating of 2). With this approach, priority scores can range from a low of 2 to a high of 8. If information is available about the likelihood that a user would work with the part of the product that enables the problem, then this information would be used to adjust the frequency rating. Continuing the example, if the expectation is that only 10% of users would encounter the problem, then the priority would be 3 (a frequency rating of 1 for the 10% x 80%, or 8% likelihood of occurrence plus an impact rating of 2).

A similar strategy is to multiply the observed percentage frequency of occurrence by the impact score. The range of priorities depends on the values assigned to each impact level. Assigning 10 to the most serious impact level leads to a maximum priority (severity) score of 1000 (which can optionally be divided by 10 to create a scale that ranges from 1 to 100). Appropriate values for the remaining three impact categories depend on practitioner judgment, but a reasonable set is 5, 3, and 1. Using those values, the problem with an observed frequency of occurrence of 80% and a minor effect on performance would have a priority of 24 (80 x 3/10). It is possible to extend this method to account for likelihood of use using the same procedure as that described by Rubin (1994), which in the example resulted in modifying the frequency measurement from 80% to 8%. Another way to extend the method is to categorize the likelihood of use with a set of categories such as very high likelihood (assigned a score of 10), high likelihood (assigned a score of 5), moderate likelihood (assigned a score of 3), and low likelihood (assigned a score of 1), and multiplying all three scores to get the final priority (severity) score (then optionally divide by 100 to create a scale that ranges from 1 to 100). Continuing the previous example with the assumption that the task in which the problem occurred has a high likelihood of occurrence, the problem's priority would be 12 (5 x 240/100). In most cases, applying the different data-driven prioritization schemes to the same set of problems should result in a very similar prioritization.

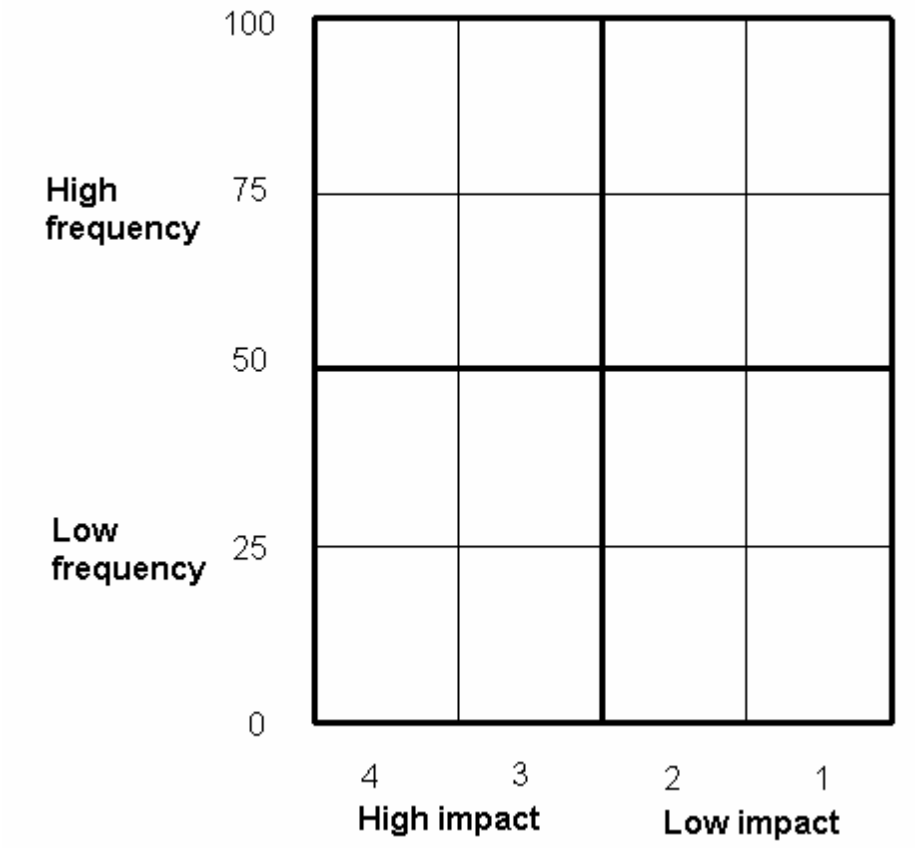

Figure 3. Sample problem grid.

Working with Quantitative Measurements

The most common use of quantitative measurements is to characterize performance and preference variables by computing means, standard deviations, and, ideally, confidence intervals. Practitioners use these results to compare observed to target measurements when targets are available. When targets are not available, the results can still be informative, for example, for use as future target measurements or as relatively gross diagnostic indicators.

The failure to meet targets is an obvious diagnostic cue. A less obvious cue is an unusually large standard deviation. Landauer (1997) describes a case in which the times to record an order were highly variable. The cause for the excessive variability was that a required phone number was sometimes, but not always, available, which turned out to be an easy problem to fix. Because the means and standard deviations of time scores tend to correlate, one way to detect an unusually large variance is to compute the coefficient of variation by dividing the standard deviation by the mean (Jeff Sauro, personal communication, April 26, 2004) or the normalized performance ratio by dividing the mean by the standard deviation (Moffat, 1990). Large coefficients of variation (or, correspondingly, small normalized performance ratios) are potentially indicative of the presence of usability problems.