❑ P. H. Winston and S. Narasimhan, *On to Java*, Addison-Wesley, 1996.
Java is another good way to get into user interface programming as examples can easily be embedded into web pages as applets. This is a good first book on Java using the construction of an applet in AWT as its motivating example. For clear reference books on aspects of the language look at the O'Reilly series.

❑ C. Gram and G. Cockton, editors, *Design Principles for Interactive Software*, Chapman and Hall, 1996.
Produced by IFIP Working Group 2.7 (User Interface Engineering). This critically discusses several user interface architectures and looks at the way architecture can help or hinder the pursuit of principles similar to those in Chapter 4.

Chapter 11

Evaluation techniques

Overview

❑ Evaluation tests the usability and functionality of an interactive system.

❑ Evaluation may take place
  – in the laboratory
  – in the field

❑ Some approaches evaluate designs
  – analytic methods
  – review methods
  – model-based methods

❑ Some approaches evaluate implementations
  – experimental methods
  – observational methods
  – query methods

❑ An evaluation method must be chosen carefully and must be suitable for the job.

11.1 What is evaluation?

In previous chapters we have discussed methodologies and models which support the design of usable interactive systems. However, even if such techniques are employed, we still need to assess our designs and test our systems to ensure that they actually behave as we expect and meet the requirements of the user. This is the role of evaluation.

Evaluation should not be thought of as a single phase in the design process (still less as an activity tacked on the end of the process if time permits). Ideally, evaluation should occur throughout the design life cycle, with the results of the evaluation feeding back into modifications to the design. Clearly it is not usually possible to perform extensive experimental testing continuously throughout the design, but analytic and informal techniques can and should be used. In this respect there is a close link between evaluation and the modelling and prototyping techniques we have already discussed – such techniques help to ensure that the design is assessed continually. This has the advantage that problems can be ironed out before considerable effort and resources have been expended on the implementation itself: it is much easier to change a design in the early stages of development than in the later stages. We can make a broad distinction between evaluation of the design of an interactive system and evaluation of an implementation, whether full or prototype. The former tends to focus on evaluation by the designer without direct involvement by users; the latter studies actual use of the system. We will discuss evaluation techniques under these two headings. However, it should be noted that these distinctions are not fixed and some techniques can be applied at either stage. Indeed, some techniques have wider application still, being used, as we have seen in Chapter 7, to develop task analyses.

11.2 Goals of evaluation

Evaluation has three main goals: to assess the extent of the system's functionality, to assess the effect of the interface on the user, and to identify any specific problems with the system. The system's functionality is important in that it must accord with the user's task requirements. In other words, the design of the system should enable the user to perform the required tasks more easily. This includes not only making the appropriate functionality available within the system, but making it clearly reachable by the user in terms of the actions that the user needs to take to perform the task. It also involves matching the use of the system to the user's expectations of the task. For example, if a filing clerk is used to retrieving a customer's file by the postal address, the same capability (at least) should be provided in the computerized file system. Evaluation at this level may also include measuring the user's performance with the system, to assess the effectiveness of the system in supporting the task.

In addition to evaluating the system design in terms of its functional capabilities, it is important to be able to measure the impact of the design on the user. This includes considering aspects such as how easy the system is to learn, its usability and the user's attitude to it. In addition, it is important to identify areas of the design which overload the user in some way, perhaps by requiring an excessive amount of information to be remembered, for example. A fuller classification of such usability measures is provided in Chapter 4. Much evaluation is aimed at measuring features such as these.

The final goal of evaluation is to identify specific problems with the design. These may be aspects of the design which, when used in their intended context, cause unexpected results, or confusion amongst users. This is of course related to both the functionality and usability of the design (depending on the cause of the problem). However, it is specifically concerned with the negative aspects of the design.

11.3 Styles of evaluation

Before we consider some of the techniques that are available for evaluation we will distinguish between a number of distinct evaluation styles. There are two main styles of evaluation: those performed under laboratory conditions and those conducted in the work environment or 'in the field'.

11.3.1 Laboratory studies

The first style of evaluation studies the use of the system within the laboratory. In some cases (particularly in evaluating the design) this involves the designer performing some assessment of the design without the involvement of users. However, users may also be brought into the laboratory to take part in evaluation studies. This approach has a number of benefits and disadvantages.

A well-equipped usability laboratory may contain sophisticated audio/visual recording facilities, two-way mirrors, instrumented computers and the like, which cannot be replicated in the field. In addition, the subject operates in an interruption-free environment. However, the lack of context – filing cabinets, wall calendars, books and so on – and the unnatural situation may mean that one accurately records a situation which never arises in the real world. It is especially difficult to observe several people cooperating on a task in a laboratory situation as interpersonal communication is so heavily dependent on context.

There are some situations where laboratory observation is the only option: if the system is to be located in a dangerous or remote location, for example a space station. Also some very constrained single-user tasks may be adequately performed in a laboratory. Finally, we may deliberately want to manipulate the context in order to uncover problems or observe less used procedures, or we may want to compare alternative designs within a controlled context. For these types of applications, laboratory studies are appropriate.

11.3.2 Field studies

The second style of evaluation takes the designer or evaluator out into the user's work environment in order to observe the system in action. Again this approach has its pros and cons.

High levels of ambient noise, greater levels of movement and constant interruptions, such as phone calls, all make field observation difficult. However, the very 'open' nature of the situation means that you will observe interactions between systems and between individuals which would have been missed in a laboratory study. The context is retained and you are seeing the user in his 'natural environment'. In addition, some activities, such as those taking days or months, are impossible to study in the laboratory (and difficult even in the field).

On balance, field observation is to be preferred to laboratory studies as it allows us to study the interaction as it occurs in actual use. Even interruptions are important as these will expose behaviours such as saving and restoring state during a task. However, we should remember that even in field observations, the subjects are likely to be influenced by the presence of the analyst and/or recording equipment, so we always operate slightly removed from the natural situation, a sort of Heisenberg uncertainty principle.

This is of course a generalization: there are circumstances, as we have noted, in which laboratory testing is necessary. In particular, controlled experiments can be useful for evaluation of specific interface features, and must normally be conducted under laboratory conditions. From an economic angle, we need to weigh the costs of establishing recording equipment in the field, and possibly disrupting the actual work situation, with the costs of taking one or more subjects away from their jobs into the laboratory. This balance is not at all obvious.

11.4 Evaluating the design

As we have noted, evaluation should occur throughout the design process. In particular, the first evaluation of a system should ideally be performed before any implementation work has started. If the design itself can be evaluated, expensive mistakes can be avoided, since the design can be altered prior to any major resource commitments. Typically, the later in the design process that an error is discovered, the more costly it is to put right. Consequently, a number of methods have been proposed to evaluate the design prior to implementation. Most of these do not involve the user directly (although there are exceptions, for example the paper and pencil walkthrough described in Section 6.5). Instead they depend upon the designer, or a human factors expert, taking the design and assessing the impact that it will have upon a typical user. The basic intent is to identify any areas which are likely to cause difficulties because they violate known cognitive principles, or ignore accepted empirical results. The methods are therefore largely analytic. Although these methods do not rely on the availability of an implementation, they can be used later in the development process on prototyped or full versions of the system, making them flexible evaluation approaches.

We will consider four possible approaches to evaluating design: the *cognitive walkthrough*, *heuristic evaluation*, review-based evaluation and the use of models. Again, these are not mutually exclusive methods.

11.4.1 Cognitive walkthrough

Cognitive walkthrough was originally proposed by Polson and colleagues [200] as an attempt to introduce psychological theory into the informal and subjective walkthrough technique. It has more recently been developed and revised making it more accessible to system designers [258]. The revised version of the walkthrough is discussed here.

The origin of the cognitive walkthrough approach to evaluation is the code walkthrough familiar in software engineering. Walkthroughs require a detailed review of a sequence of actions. In the code walkthrough, the sequence represents a segment of the program code that is stepped through by the reviewers to check certain characteristics (for example, that coding style is adhered to, conventions for spelling variables versus procedure calls, and to check that system-wide invariants are not violated). In the cognitive walkthrough, the sequence of actions refers to the steps that an interface will require a user to perform in order to accomplish some task. The evaluators then step through that action sequence to check it for potential usability problems. Usually, the main focus of the cognitive walkthrough is to establish how easy a system is to learn. More specifically, the focus is on learning through exploration. Experience shows that many users prefer to learn how to use a system by exploring its functionality hands on, and not after sufficient training or examination of a user's manual. So the checks that are made during the walkthrough ask questions that address this exploratory learning. To do this, the evaluators go through each step in the task and provide a story about why that step is or is not good for a new user. To do a walkthrough (the term walkthrough from now on refers to the cognitive walkthrough, and not any other kinds of walkthroughs), you need four things:

1. A description of the prototype of the system. It doesn't have to be complete, but it should be fairly detailed. Details such as the location and wording for a menu can make a big difference.
2. A description of the task the user is to perform on the system. This should be a representative task that most users will want to do.
3. A complete, written list of the actions needed to complete the task with the given prototype.
4. An indication of who the users are and what kind of experience and knowledge the evaluators can assume about them.

Given this information, the evaluators step through the action sequence (item 3 above) to critique the system and tell a believable story about its usability. To do this, for each action, the evaluators try to answer the following four questions.

1. **Will the users be trying to produce whatever effect the action has?** Are the assumptions about what task the action is supporting correct given the users' experience and knowledge up to this point in the interaction?
2. **Will users be able to notice that the correct action is available?** Will users see the button or menu item, for example, by which the next action is

actually achieved by the system? This is not asking whether they will know that the button is the one they want. This is merely asking whether it is visible to them at the time when they will need to invoke it. An example of when this question gets a negative supporting story might be if a VCR remote control has a hidden panel of buttons that are not obvious to a new user.

3. **Once users find the correct action at the interface, will they know that it is the right one for the effect they are trying to produce?** This complements the previous question. It is one thing for a button or menu item to be visible, but will the users know that it is the one they are looking for to complete their task?
4. **After the action is taken, will users understand the feedback they get?** Assuming the users did the correct action, will they know that? This is the completion of the execution/evaluation interaction cycle. In order to determine if they have accomplished their goal, the users need appropriate feedback.

It is vital to document the cognitive walkthrough to keep a record of what is good and what needs improvement in the design. It is therefore good to produce some standard evaluation forms for the walkthrough. The cover form would list the information in items 1–4 above, as well as identify the date and time of the walkthrough and the names of the evaluators. Then for each action (from item 3 on the cover form), a separate standard form is filled out that answers each of the questions above. Any negative answer for any of the questions for any particular action should be documented on a separate usability problem report sheet. This problem report sheet should indicate the system being built (the version, if necessary), the date, the evaluators and a detailed description of the usability problem. It would also be useful to determine the severity of the problem, that is whether the evaluators think this problem will occur often, and an impression of how serious the problem will be for the users. This information will help the designers to decide priorities for correcting the design.

Example: programming a video by remote control

We can illustrate how the walkthrough method works using a simple example. Imagine we are designing a remote control for a video recorder which will allow users to program the video to perform timed recording. Our initial design is shown in Figure 11.1. The first picture illustrates the hand set in normal use, the second after the timed record button has been pressed. The video allows the user to program up to three timed recordings in different 'streams'. The next available stream number is automatically assigned. We want to know whether our design supports the user's task. We begin by identifying a representative task.

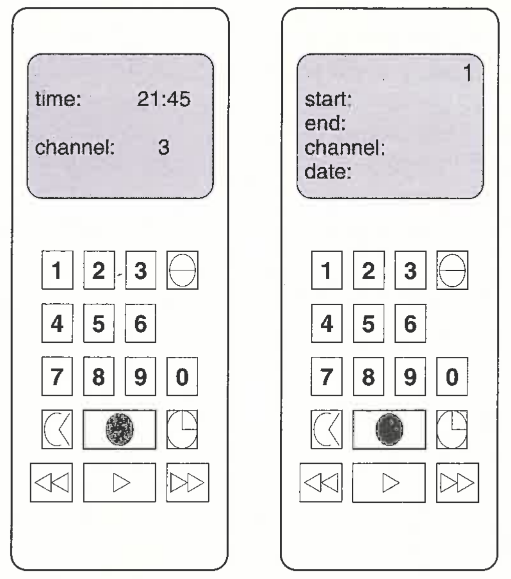

Figure 11.1 An initial remote control design

Program the video to time-record a program starting at 18.00 and finishing at 19.15 on channel 4 on 1 January 1999.

We will assume that the user is familiar with video recorders but not with this particular design.

The next step in the walkthrough is to identify the action sequence for this task. We specify this in terms of the user's action (Action) and the system's response (Response):

Action A: Press the 'timed record' button
Response A: Display moves to timer mode. Flashing cursor appears after 'Start:'
Action B: Press digits 1800
Response B: Each digit is displayed as typed and flashing cursor moves to next position
Action C: Press the 'timed record' button
Response C: Flashing cursor moves to 'End:'
Action D: Press digits 1915
Response D: Each digit is displayed as typed and flashing cursor moves to next position
Action E: Press the 'timed record' button
Response E: Flashing cursor moves to 'Channel:'
Action F: Press digit 4
Response F: Digit is displayed as typed and flashing cursor moves to next position
Action G: Press the 'timed record' button
Response G: Flashing cursor moves to 'Date:'

Action H: Press digits 010199
Response H: Each digit is displayed as typed and flashing cursor moves to next position
Action I: Press the 'timed record' button
Response I: Stream number in top right hand corner of display flashes
Action J: Press the transmit button
Response J: Details are transmitted to video player and display returns to normal mode

Having determined our action list we are in a position to proceed with the walkthrough. For each action (A, B, C, etc.) we must answer the four questions and tell a story about the usability of the system. Beginning with Action A:

Action A: Press the 'timed record' button
Question 1: Will the users be trying to produce whatever effect the action has?
The interface provides no indication that the user needs to press the 'timed record'. However, it is reasonable to assume that a user familiar with video recorders would know that this was required.
Question 2: Will users be able to notice that the correct action is available?
The 'timed record' button is visible on the remote control.
Question 3: Once users find the correct action at the interface, will they know that it is the right one for the effect they are trying to produce?
It is not clear which button is the 'timed record' button. The icon of a clock (fourth button down on right) is a possible candidate but this could be interpreted as a button to change the time. Other possible candidates might be the fourth button down on the left or the filled circle (associated with record). In fact the icon of the clock is the correct choice but it is quite possible that the user would fail at this point.
Question 4: After the action is taken, will users understand the feedback they get?
Once the action is taken the display changes to the timed record mode and shows familiar headings (start, end, channel, date). It is reasonable to assume that the user would recognize these as indicating successful completion of the first action.

So we find we have a potential usability problem relating to the icon used on the 'timed record' button. We would have to establish whether our user group would correctly distinguish this icon from others on the remote. The analysis proceeds in this fashion, with a walkthrough form completed for each action. We will leave the rest of the walkthrough for you to complete as an exercise. What other usability problems can you identify?

11.4.2 Heuristic evaluation

A heuristic is a guideline or general principle or rule of thumb that can guide a design decision or be used to critique a decision that has already been made. *Heuristic evaluation*, developed by Jakob Nielsen and Rolf Molich, is a method for structuring the critique of a system using a set of relatively simple and general heuristics.

The general idea behind heuristic evaluation is that several evaluators independently critique a system to come up with potential usability problems. It is important that there be several of these evaluators and that the evaluations be done independently. Nielsen's experience indicates that around five evaluators usually results in about 75% of the overall usability problems being discovered.

What is evaluated? Heuristic evaluation is best used for evaluating early designs, because it is easier to fix a lot of the usability problems that arise. But all that is required to do the evaluation is some kind of artefact that describes the system, which can range from a set of storyboards giving an overview of the system to a fully functioning system that is in use in the field.

To aid the evaluators in discovering usability problems, there is a list of nine heuristics which can be used to generate ideas while critiquing the system. The heuristics are related to *principles* and *guidelines* (see Chapter 4). The original list of heuristics is as follows:

1. **Simple and natural dialog**
   - ❑ simple means no irrelevant or rarely used information
   - ❑ natural means an order that matches the task.
2. **Speak the user's language**
   - ❑ use concepts from the user's world
   - ❑ don't use system-specific engineering terms.
3. **Minimize user memory load**
   - ❑ don't make the user remember things from one action to the next
   - ❑ leave information on the screen until it is no longer needed.
4. **Be consistent**
   - ❑ action sequences learned in one part of the system should apply in other parts.
5. **Provide feedback**
   - ❑ let users know what effect their actions have on the system.
6. **Provide clearly marked exits**
   - ❑ if users get into part of the system that doesn't interest them, they should be able to get out quickly without damaging anything.
7. **Provide short cuts**
   - ❑ help experienced users avoid lengthy dialogs and informational messages they don't need.
8. **Good error messages**
   - ❑ let the user know what the problem is and how to correct it.
9. **Prevent errors**
   - ❑ whenever you discover an error message, ask if that error could have been prevented.

These heuristics have recently been improved to make them more comprehensive. Now there are 10:

1. **Visibility of system status** The system should always keep users informed about what is going on, through appropriate feedback within reasonable time.
2. **Match between system and the real world** The system should speak the user's language, with words, phrases and concepts familiar to the user, rather than system-oriented terms. Follow real-world conventions, making information appear in natural and logical order.
3. **User control and freedom** Users often choose system functions by mistake and will need a clearly marked 'emergency exit' to leave the unwanted state without having to go through an extended dialog. Support undo and redo.
4. **Consistency and standards** Users should not have to wonder whether different words, situations or actions mean the same thing. Follow platform conventions.
5. **Error prevention** Even better than good error messages is a careful design which prevents a problem from occurring in the first place.
6. **Recognition rather than recall** Make objects, actions and options visible. The user should not have to remember information from one part of the dialog to another. Instructions for use of the system should be visible or easily retrievable whenever appropriate.
7. **Flexibility and efficiency of use** Accelerators – unseen by the novice user – may often speed up the interaction for the expert user to such an extent that the system can cater to both inexperienced and experienced users. Allow users to tailor frequent actions.
8. **Aesthetic and minimalist design** Dialogs should not contain information which is irrelevant or rarely needed. Every extra unit of information in a dialog competes with the relevant units of information and diminishes their relative visibility.
9. **Help users recognize, diagnose and recover from errors** Error messages should be expressed in plain language (no codes), precisely indicate the problem, and constructively suggest a solution.
10. **Help and documentation** Even though it is better if the system can be used without documentation, it may be necessary to provide help and documentation. Any such information should be easy to search, focused on the user's task, list concrete steps to be carried out, and not be too large.

Remember – the purpose of the evaluation is to uncover usability problems. Any problem you as an evaluator think is a potential problem IS a usability problem. Don't worry too much about which heuristic justifies the problem. The heuristics are there to guide you in finding the problems. Once all of the problems are collected, the design team can determine which ones are the most important and will receive attention.

11.4.3 Review-based evaluation

Experimental psychology and human–computer interaction between them possess a wealth of experimental results and empirical evidence. Some of this is specific to a particular domain, but much deals with more generic issues and applies in a variety of situations. Examples of such issues are the usability of different menu types, the recall of command names, and the choice of icons.

One approach to evaluating a design is to exploit this inheritance and scour the literature for evidence to support (or refute) aspects of the design. After all, it is wasteful to repeat experiments continually, for the sake of it, and, although a literature review is time consuming to perform, it can probably be completed in the time that it would take to repeat just one experiment.

However, it should be noted that experimental results cannot be expected to hold arbitrarily across contexts. The reviewer must therefore select evidence carefully, noting the experimental design chosen, the population of subjects used, the analyses performed and the assumptions made. For example, an experiment testing the usability of a particular style of help system using novice subjects may not provide accurate evaluation of a help system designed for expert users. The review should therefore take account of both the similarities and the differences between the experimental context and the design under consideration.

11.4.4 Model-based evaluation

The final approach to evaluating the design that we will note is the use of models. Certain cognitive and design models provide a means of combining design specification and evaluation into the same framework. For example, the GOMS model (see Chapter 6) predicts user performance with a particular interface and can be used to filter particular design options. Similarly, lower-level modelling techniques such as the keystroke-level model (Chapter 6) provide predictions of the time users will take to perform low-level physical tasks.

Design methodologies, such as design rationale (see Chapter 5), also have a role to play in evaluation at the design stage. Design rationale provides a framework in which design options can be evaluated. By examining the criteria that are associated with each option in the design, and the evidence that is provided to support these criteria, informed judgements can be made in the design.

11.5 Evaluating the implementation

The techniques we have considered so far concentrate on evaluating the design before it exists as a runnable system, although as we have noted they can also be used with implementations. They typically involve analysis by the designer (or an expert evaluator) rather than testing with actual users. However, vital as these techniques are for filtering and refining the design, they are not a replacement for actual usability testing with the people for whom the system is intended: the users. In this section we will look at a number of different approaches to user-centred evaluation. These include empirical or experimental methods, observational methods and techniques which we will call *query techniques*, which ask the user for feedback directly.

The major difference between this type of evaluation and evaluation of design, aside from the involvement of the user, is the existence of an actual implementation of the system in some form. This may range from a simulation of the system's interactive capabilities, without its underlying functionality (for example, the *Wizard of Oz* technique, which is discussed in Chapter 5), through a basic functional prototype to a fully implemented system. The evaluation techniques described in this section can be used to evaluate any of these.

11.5.1 Empirical methods: experimental evaluation

One of the most powerful methods of evaluating a design or an aspect of a design is to use a controlled experiment. This provides empirical evidence to support a particular claim or hypothesis. It can be used to study a wide range of different issues at different levels of detail.

Any experiment has the same basic form. The evaluator chooses a hypothesis to test, which can be determined by measuring some attribute of subject behaviour. A number of experimental conditions are considered which differ only in the values of certain controlled variables. Any changes in the behavioural measures are attributed to the different conditions. Within this basic form there are a number of factors that are important to the overall reliability of the experiment, which must be considered carefully in experimental design. These include the subjects chosen, the variables tested and manipulated, and the hypothesis tested.

Subjects

The choice of subjects is vital to the success of any experiment. In evaluation experiments subjects should be chosen to match the expected user population as closely as possible. Ideally this will involve experimental testing with the actual users but this is not always possible. If subjects are not actual users they should be chosen to be of a similar age and level of education as the intended user group. Their experience with computers in general, and with systems related to that being tested, should be similar, as should their experience or knowledge of the task domain. It is no good testing an interface designed to be used by the general public on a subject set made up of computer science undergraduates: they are simply not representative of the intended user population.

A second issue relating to the subject set is the sample size chosen. Often this is something which is determined by pragmatic considerations: the availability of subjects is limited or resources are scarce. However, the sample size must be large enough to be considered to be representative of the population taking into account the design of the experiment and the statistical methods chosen. As a rough guide a sample size of at least 10 subjects is recommended for controlled experiments.

Variables

Experiments manipulate and measure variables under controlled conditions, in order to test the hypothesis. There are two main types of variable: those that are manipulated and those that are measured. The former are known as *independent variables*, the latter as *dependent variables*.

Independent variables are those characteristics of the experiment which are manipulated to produce different conditions for comparison. Examples of independent variables in evaluation experiments are interface style, level of help, number of menu items and icon design. Each of these variables can be given a number of different values; each value that is used in an experiment is known as a *level* of the variable. So, for example, an experiment that wants to test whether search speed improves as the number of menu items decreases may consider menus with five, seven, and 10 items. Here the independent variable, number of menu items, has three levels.

More complex experiments may have more than one independent variable. For example, in the above experiment, we may suspect that the speed of the user's response depends not only on the number of menu items but also on the choice of commands used on the menu. In this case there are two independent variables. If there were two sets of command names (that is, two levels), we would require six experimental conditions to investigate all the possibilities.

Dependent variables on the other hand are the variables which can be measured in the experiment. In the example given above, this would be the speed of menu selection. The dependent variable must be measurable in some way, it must be affected by the independent variable, and, as far as possible, unaffected by other factors. Common choices of dependent variable in evaluation experiments are the time taken to complete a task, the number of errors made, user preference and the quality of the user's performance. Obviously some of these are easier to measure objectively than others. However, the more subjective measures can be applied against predetermined scales, and can be very important factors to consider.

Hypotheses

A hypothesis is a prediction of the outcome of an experiment. It is framed in terms of the independent and dependent variables, stating that a variation in the independent variable will cause a difference in the dependent variable. The aim of the experiment is to show that this prediction is correct. This is done by disproving the null hypothesis, which states that there is no difference in the dependent variable between the levels of the independent variable. The statistical measures described below produce values which can be compared with various levels of significance. If a result is significant it shows, at the given level of certainty, that the differences measured would not have occurred by chance (that is, that the null hypothesis is incorrect).

Experimental design

In order to produce reliable and generalizable results, an experiment must be carefully designed. We have already looked at a number of the factors which the experimenter must consider in the design, namely the subjects, the independent and dependent variables, and the hypothesis. The first phase in experimental design then is to choose the hypothesis: to decide exactly what it is you are trying to demonstrate. In doing this you are likely to clarify the independent and dependent variables, in that you will have identified what you are going to manipulate and what change you expect. If your hypothesis does not clearly identify these variables then you need to rethink it. At this stage you should also consider your subjects: how many are available and are they representative of the user group?

The next step is to decide on the *experimental method* which you will use. There are two main methods: *between-groups* and *within-groups*. In a between-groups (or *randomized*) design, each subject is assigned to a different condition. There are at least two conditions: the experimental condition (in which the variable has been manipulated) and the control, which is identical to the experimental condition except for this manipulation. This control serves to ensure that it is the manipulation that is responsible for any differences which are measured. There may of course be more than two groups, depending on the number of independent variables and the number of levels which each variable can take.

The advantage of a between-groups design is that any learning effect resulting from the user performing in one condition and then the other is controlled: each user performs under only one condition. The disadvantages are that a greater number of subjects are required, and that significant variation between the groups can negate any results. Also, individual differences between users can bias the results. These problems can be handled by a careful selection of subjects, ensuring that all are representative of the population.

The second experimental design is within-groups. Here each user performs under each different condition. This design can suffer from transfer of learning effects, but this can be lessened if the order in which the conditions are tackled is varied between users. Within-groups is less costly than between-groups, since fewer users are required, and it can be particularly effective where learning is involved. There is also less chance of effects from variation between subjects.

The choice of experimental method will depend on the resources available, how far learning transfer is likely or can be controlled, and how representative the subject group is considered to be. A popular compromise, in cases where there is more than one independent variable, is to devise a mixed design where one variable is placed between groups and one within groups. So, returning to our example of the menu design, the subjects would be split into two groups, one for each command set, but each group would perform in three conditions, corresponding to the three possible levels of the number of menu items.

Once we have determined the hypothesis we are trying to test, the variables we are studying, the subjects at our disposal, and the design that is most appropriate, it remains for us to decide how we are going to analyze the results we record. There are a number of statistical tests available, and the choice of test is vital to the success of the experiment. Different tests make different assumptions about the data and if an inappropriate test is chosen, the results can be invalid. The next subsection discusses the factors to consider in choosing a statistical test and surveys the most common statistical measures available.

Statistical measures

The first two rules of statistical analysis are to *look* at the data and to *save* the data. It is easy to carry out statistical tests blindly when a glance at a graph, histogram or table of results would have been more instructive. In particular, it can expose *outliers*, single data items which are very different from the rest. Outliers are often the result of a transcription error, or a freak event not connected to the experiment. For example, we notice that one subject took three times as long as everyone else to do a task. We investigate and discover that the subject had been suffering from flu on the day of the experiment. Clearly, if the subject's data were included it would bias the results.

Saving the data is important as we may later want to try a different analysis method. It is all too common for an experimenter to take some averages or otherwise tabulate results, and then throw away the original data. At worst, the remaining statistics can be useless for statistical purposes, and, at best, we have lost the ability to trace back odd results to the original data, as, for example, we want to do for outliers.

Our choice of statistical analysis depends on the type of data and the questions we want to answer. It is worth having important results checked by an experienced statistician, but in many situations standard tests can be used.

Variables can be classified as either *discrete variables* or *continuous variables*. A discrete variable can only take a finite number of values or *levels*, for example a screen colour which can be red, green or blue. A continuous variable can take any value (although it may have an upper or lower limit), for example a person's height or the time taken to complete a task. A special case of continuous data is when they are *positive*, for example a response time cannot be negative. A continuous variable can be rendered discrete by clumping it into classes, for example we could divide heights into short (<5 ft (1.5 m)), medium (5 ft–6 ft (1.5 m–1.8 m)) and tall (>6 ft (1.8 m)). In many interface experiments we will be testing one design against another. In these cases the independent variable is usually discrete.

The dependent variable is the measured one and subject to random experimental variation. In the case when this variable is continuous, the random variation may take a special form. If the form of the data follows a known *distribution* then special and more powerful statistical tests can be used. Such tests are called *parametric tests* and the most common of these are used when the variation follows the *normal distribution*. This means that if we plot a histogram of the random errors, they will form the well-known bell-shaped graph (Figure 11.2). Happily, many of these tests are fairly *robust*, that is they give reasonable results even when the data are not precisely normal. This means that you need not worry too much about checking normality during early analysis.

There are ways of checking whether data are really normal, but for these the reader should consult a statistics book, or a professional statistician. However, as a general rule, if data can be seen as the sum or average of many small *independent* effects they are likely to be normal. For example, the time taken to complete a *complex* task is the sum of the times of all the minor tasks of which it is composed. On the other hand, a subjective rating of the usability of an interface will not be normal. Occasionally data can be *transformed* to become approximately

Worked exercise
Design an experiment to test whether adding colour coding to an interface will improve accuracy.

Answer
The following is only an example of the type of experiment that might be devised.

Subjects Taken from user population.

Hypothesis Colour coding will make selection more accurate.

IV (Independent Variable) Colour coding.

DV (Dependent Variable) Accuracy measured as number of errors.

Design Between groups to ensure no transfer of learning (or within groups with appropriate safeguards if subjects are scarce).

Task The interfaces are identical in each of the conditions, except that, in the second, colour is added to indicate related menu items. Subjects are presented with a screen of menu choices (ordered randomly) and verbally told what they have to select. Selection must be done within a strict time limit when the screen clears. Failure to select the correct item is deemed an error. Each presentation places items in new positions. Subjects perform in one of the two conditions.

Analysis t test.

Example of non-parametric statistics
We will not see an example of the use of non-parametric statistics later, so we will go through a small example here. Imagine we had the following data for response times under two conditions:

condition A:	33, 42, 25, 79, 52
condition B:	87, 65, 92, 93, 91, 55

We gather the data together and sort them into order: 25, 33, 42, ... , 92, 93. We then substitute for each value its rank in the list: 25 becomes 1, 33 becomes 2, etc. The transformed data are then

condition A:	2, 3, 1, 7, 4
condition B:	8, 6, 10, 11, 9, 5

Tests are then carried out on the data. For example, to test whether there is any difference between the two conditions we can use the *Wilcoxon test*. To do this we take each condition and calculate the sum of ranks, and subtract the least value it could have (that is, 1 + 2 + 3 + 4 + 5 = 15 for condition A, 1 + 2 + 3 + 4 + 5 + 6 = 21 for condition B), giving the statistic U:

	rank sum		least		U
condition A:	(2 + 3 + 1 + 7 + 4)	–	15	=	2
condition B:	(8 + 6 + 10 + 11 + 9 + 5)	–	21	=	28

In fact, the sum of these two U statistics, 2 + 28 = 30, is the product of the number of data values in each condition 5×6. This will always happen and so one can always get away with calculating only one of the U. Finally, we then take the smaller of two U values and compare it with a set of *critical values* in a book of statistical tables, to see if it is unusually small. The table is laid out dependent on the number of data values in each condition (five and six). The critical value at the 5% level turns out to be 3. As the smallest statistic is smaller than this, we can *reject the null hypothesis* and conclude that there is likely to be a difference between the conditions. To be precise it says that there is only a 1 in 20 (5%) chance that the data happened by chance. In fact the test is right – the authors constructed random data in the range 1–100 and then subtracted 10 from each of the values in condition A.

An example: evaluating icon designs
Imagine you are designing a new word-processing package which is to use icons for presentation. You are considering two styles of icon design and you wish to know which design will be easiest for users to remember. One set of icons uses naturalistic images (based on a paper document metaphor), the other uses abstract images (see Figure 11.3). How might you design an experiment to help you decide which style to use?

The first thing you need to do is form a hypothesis: what do you consider to be the likely outcome? In this case you might expect the natural icons to be easier to recall since they are more familiar to users. We can therefore form the following hypothesis:

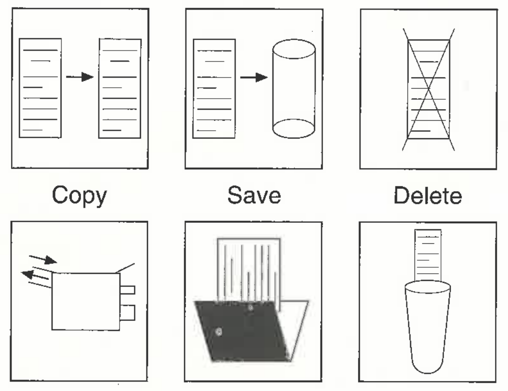

Figure 11.3 Abstract and concrete icons for file operations

between the means is 52 seconds, but the *standard error of the difference* (s.e.d.) is 117. This is calculated as follows:

$$\text{s.e.d} = \sqrt{\frac{\sigma_N^2}{n_N} + \frac{\sigma_A^2}{n_A}} = \sqrt{\frac{265^2}{10} + \frac{259^2}{10}} = 117.2$$

where σ_N and σ_A are the standard deviations of the two conditions and n_N and n_A are the number of data items in each condition (10 in each). The s.e.d. is a measure of the expected variability of the difference between the means, and as we see the actual difference is well within this random variation. Testing the ratio 52/117 against tables of Student's *t* distribution indeed shows that this is not significant.

However, if we glance down the table, we see that in almost every case the time taken with the abstract icons is greater than the time taken for the natural icons. That is, the data seem to support our claim that natural icons are better than abstract ones, but the wide variation between individuals has hidden the effect.

A more sophisticated analysis, a special case of ANOVA, can expose the difference. Looking back at the table, column (3) shows, for each subject, the average of the time they took under the two conditions. This subject mean is then subtracted from the data for each condition, yielding columns (4) and (5). These columns show the effect of the icon design *once the differences between subjects have been removed*. The two columns are redundant as they always add up to zero. They show that in all but one case (subject 9) the natural icons are faster than the abstract ones.

Even a non-parametric test would show this as a significant difference at the 5% level, but the use of a *t* test is more precise. We can take either column and see that the column average 26 is much greater than the standard error ($14.4 / \sqrt{10}$). The ratio (mean/s.e.) is compared with the Student's *t* table (in statistical tables) using nine degrees of freedom (10 values minus 1 for the mean), and is indeed far greater than the 1% level (3.250); that is, the chance of getting our results by chance is less than 1 in 100. So, we reject the *null hypothesis* that there is no difference and conclude that natural icons are more easily remembered than abstract ones.

In fact, the last statement is not quite correct. What we have shown is that in this experiment natural icons are more *rapidly* remembered. Possibly if we go on to analyze the errors, these may present a different story. If these error figures were quite large (say 15 errors or more per condition), then we may be able to assume these are normal and use ANOVA. If not, we can either use non-parametric tests, or make use of special tests based on the *binomial distribution*. We will not perform these analyses here. Possibly, looking at the errors we may find that the natural icons have *more* errors – it could well be that they are more rapidly, but less accurately, remembered. It is always worth keeping in mind the difference between the intended purpose of the experiment (to see which is better remembered) and the actual measurements (speed and accuracy).

Finally, one ought to look carefully at the experimental results to see whether there is any other effect which might confuse the results. The graphical presentation of results will help with this, possibly highlighting odd clumps in the data or other irregularities. In this experiment we may want to check to see if there has been

any significant *transfer effect* between the first and second condition for each subject. The second set may be faster as the subjects are more practised, or possibly the second set may be slower as learning a second set of icons may be confusing. This will not matter if the effect is uniform – say they always are 15 seconds slower on the second test. But there may be systematic effects. For example, seeing the natural icons first might make it more difficult to learn the abstract ones, but not vice versa. If this were the case, our observed effect may be about the interference between the icon sets, rather than that one is better than the other.

11.5.2 Observational techniques

Think aloud and cooperative evaluation

A popular way to gather information about actual use of a system is to observe users interacting with it. Usually they are asked to complete a set of predetermined tasks, although, if observation is being carried out in the users' place of work, they may be observed going about their normal duties. The evaluator watches and records the users' actions (using a variety of techniques – see below). Simple observation is seldom sufficient to determine how well the system meets the users' requirements since it does not always give insight into the users' decision processes or attitude. Consequently users are asked to elaborate their actions by 'thinking aloud': describing what they believe is happening, why they take an action, what they are trying to do.

Think aloud has the advantage of simplicity; it requires little expertise to perform and can provide useful insight into problems with an interface. Also it may be used to observe how the system is actually used. It can be used for evaluation throughout the design process using paper or simulated mock-ups for the earlier stages. However, the information provided is often subjective and may be selective, depending on the tasks provided. The process of observation can alter the way that people perform tasks and so provide a biased view. The very act of describing what you are doing often changes the way you do it – like the joke about the centipede who was asked how he walked …

A variation on think aloud is known as *cooperative evaluation* [162] in which the user is encouraged to see himself as a collaborator in the evaluation and not simply as an experimental subject. As well as asking the user to think aloud at the beginning of the session the evaluator can ask the user questions (typically of the 'why?' or 'what-if?' type) if the user's behaviour is unclear, and the user can ask the evaluator for clarification if a problem arises. This more relaxed view of the think aloud process has a number of advantages:

- the process is less constrained and therefore easier to learn to use by the evaluator
- the user is encouraged to criticize the system
- the evaluator can clarify points of confusion at the time they occur and so maximize the effectiveness of the approach for identifying problem areas.

The usefulness of think aloud and general observation is largely dependent on the effectiveness of the recording method and subsequent analysis. The record of an evaluation session of this type is known as a *protocol*, and there are a number of methods from which to choose.

Protocol analysis

There are a number of methods for recording user actions. These include the following:

Paper and pencil This is primitive, but cheap, and allows the analyst to note interpretations and extraneous events as they occur. However, it is hard to get detailed information as it is limited to the analyst's writing speed. Coding schemes for frequent activities, developed during preliminary studies, can improve the rate of recording substantially. A variation of paper and pencil is the use of a notebook computer for direct entry, but then one is limited to the analyst's typing speed, and one loses the flexibility of paper for writing styles, quick diagrams and spatial layout.

Audio recording This is useful if the user is actively 'thinking aloud'. However, it may be difficult to record sufficient information to identify exact actions in later analysis, and difficult to match an audio recording to some other form of protocol (such as a handwritten script).

Video recording This has the advantage that we can see *what* the subject is doing, *as long* as the subject stays within the range of the camera. Choosing suitable camera positions and viewing angles so that you get sufficient detail (remember the resolution is not that good) and yet keep the subject in view is difficult. Alternatively, one has to ask the subject not to move – not conducive to normal activity. For single-user computer-based tasks, one typically uses two video cameras, one looking at the computer screen and one with a wider focus including the user's face and hands. The former camera may not be necessary if the computer system is being logged.

Computer logging It is relatively easy to get a system automatically to record user actions at a keystroke level, particularly if this facility has been considered early in the design. It can be more difficult with proprietary software where source code is not available (although some software now provides built-in logging and play-back facilities). Obviously, computer logging only tells us what the user is doing on the system, but this may be sufficient for some purposes. Keystroke data are also 'semantics free' in that they only tell us about the lowest-level actions, not why they were performed or how they are structured (although slight pauses and gaps can give clues). Direct logging has the advantages that it is cheap (except in terms of disk storage), unobtrusive and can be used for *longitudinal studies*, where we look at one or more users over periods of weeks or months. Technical problems with it are that the sheer volume of data can become unmanageable without automatic analysis, and that one often has to be careful to restore the state of the system (file contents etc.) before replaying the logs.

User notebooks The subjects themselves can be asked to keep logs of activity/problems. This obviously will be at a very coarse level – at most records every few minutes and more likely hourly or less. It also gives us 'interpreted' records which have advantages and problems. The technique is especially useful in longitudinal studies, and also where we want a log of unusual or infrequent tasks and problems.

In practice, one uses a mixture of recording methods as they complement one another. For instance, we may keep a paper note of special events and circumstances, even when we have more sophisticated audio/visual recording. Similarly, one may use separate audio recording, even where a video recorder is used, as the quality of specialist audio recording is better than most built-in video microphones. In addition, we may use stereo audio recording which helps us to locate out-of-screen noises. If one is using a collection of different sources, say audio, video (×2) and keystroke logging, there is considerable difficulty in synchronizing them during play-back. Most video recorders can superimpose an on-screen clock, which can help, but ideally one uses specialized equipment which can automatically synchronize the different sources, possibly merging several video displays onto a single screen. Unfortunately, this sort of equipment is often only available in specialized laboratories.

With both audio and video recording, a major problem is *transcription*. Typing a transcript from a tape is not the same as taped dictation. The conversation will typically consist of part or broken sentences, mumbled words and inarticulated noises. In addition, the transcript will need annotating with the different voices (which may only be clear from context) and with non-verbal items such as pauses, emphases, equipment noises, phones ringing, etc. A good audio-typist will be accustomed to completing mumbled words and correcting ungrammatical sentences – typing *exactly* what is recorded may prove difficult. Some practitioners say that the use of typists is not good practice anyway as the analyst will miss many nuances which are lost in the written transcript. However, if you wish to produce your own typed transcripts from tape, a course in touch-typing is highly recommended.

For video transcription, professional typists are not an option; there is no standard way of annotating video recordings, and the analyst must invent notations to suit the particular circumstances. The scale of this task is not to be underestimated. It is common to talk to practitioners who have tens or hundreds of hours of video recording, but have only analyzed tiny fragments in detail. Of course, the fragments will have been chosen after more extensive perusal of the material, but it certainly removes any idea of comprehensive coverage.

Automatic protocol analysis tools

Analyzing protocols, whether video, audio or system logs, is time consuming and tedious by hand. It is made harder if there is more than one stream of data to synchronize. A possible solution to this problem is to provide automatic analysis tools to support the task. As yet most systems of this type are experimental but they offer a means of editing and annotating video, audio and system logs and synchronizing these for detailed analysis.

EVA (*Experimental Video Annotator*) is a system which runs on a multimedia workstation with a direct link to a video recorder [145]. The evaluator can devise a set of buttons indicating different events. These may include timestamps and snapshots, as well as notes of expected events and errors. The buttons are used within a recording session by the evaluator to annotate the video with notes. During the session the user works at a workstation and is recorded, using video and perhaps audio and system logging as well. The evaluator uses the multimedia workstation running EVA. On the screen is the live video record and a view of the user's screen (see Figure 11.4). The evaluator can use the buttons to tag interesting events as they occur and can record additional notes using a text editor. After the session the evaluator can ask to review the tagged segments and can then use these and standard video controls to search the information. Links can be made with other types of record such as audio and system logs. A system such as EVA alleviates the burden of video analysis but it is not without its problems. The act of tagging and annotating events can prevent the evaluator actually concentrating on the events themselves. This may mean that events are missed or tagged late. It also requires specialist hardware to run, which may only be available in the most well-equipped laboratories.

The *Workplace project* at Xerox PARC [241] also includes a system to aid protocol analysis. The main emphasis here is to support the analysis of synchronized information from different data streams, such as video, audio, notes and diagrams. Each data stream is viewed in an aligned display so that it is possible to compare the records of each for a given point in the interaction. The alignment may be based on timestamps or on an event or action and is implemented using hypertext links.

A third example is DRUM [148] which also provides video annotation and tagging facilities. DRUM is part of the MUSiC (Measuring the Usability of Systems in Context/Metrics for Usability Standards in Computing) toolkit which supports

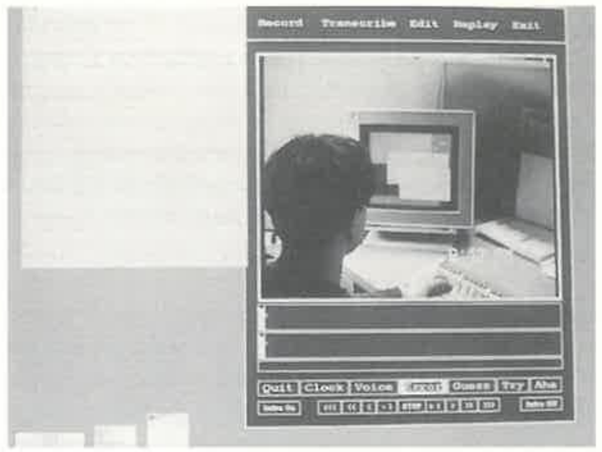

Figure 11.4 EVA: an automatic protocol analysis tool. Courtesy Wendy Mackay

a complete methodology for evaluation, based upon the application of usability metrics on analytic metrics, cognitive workload, performance and user satisfaction. DRUM is concerned particularly with measuring performance. The methodology provides a range of tools as well as DRUM, including manuals, questionnaires, analysis software and databases.

Systems such as these are extremely important as evaluation tools since they offer a means of handling the data that are collected in observational studies and allowing a more systematic approach to the analysis. The evaluator's task is facilitated and it is likely that more valuable observations will emerge as a result.

Post-task walkthroughs

Often data obtained via direct observation lack interpretation. We have the basic actions which were performed, but little knowledge as to why. Even where the subject has been encouraged to think aloud through the task, the information may be at the wrong level. For example, the subject may say 'and now I'm selecting the undo menu', but not tell us what was wrong to make undo necessary. In addition, a think aloud does not include information such as alternative, but not pursued, actions.

A walkthrough attempts to alleviate these problems, by reflecting back to the subjects their actions, after the event. The transcript, whether written or recorded, is replayed to the subject who is invited to comment, or is directly questioned by the analyst. This may be done straight away, when the subject may actually remember why certain actions were performed, or after an interval, when the answers are more likely to be the subject's *post hoc* interpretation. (In fact, interpretation is likely even in the former case.) The advantage of a delayed walkthrough is that the analyst has had time to frame suitable questions and focus on specific incidents. The disadvantage is a loss of freshness.

There are some circumstances when the subject cannot be expected to talk during the actual observation, for instance during a critical task, or when the task is too intensive. In these circumstances, the post-task walkthrough is the only way to obtain a subjective viewpoint of the user's behaviour. There is also an argument that it is preferable to minimize non-task-related talk during direct observation in order to get as natural a performance as possible. Again this makes the walkthrough essential.

11.5.3 Query techniques

Query techniques are less formal than controlled experimentation, but can be useful in eliciting detail of the user's view of a system. They embody the philosophy which states that the best way to find out how a system meets user requirements is to 'ask the user'. They can be used in evaluation and more widely to collect information about user requirements and tasks. The advantage of such methods is that they get the user's viewpoint directly and may reveal issues which have not been considered by the designer. In addition they are relatively simple and cheap to administer. However, the information gained is necessarily subjective, and may be a 'rationalized' account of events rather than a wholly accurate one. Also it may be difficult to get accurate feedback about alternative designs if the user has not experienced them, which limits the scope of the information that can be gleaned. However, the methods provide useful supplementary material to other methods. There are two main types of query technique: interviews and questionnaires.

Interviews

Interviewing users about their experience with an interactive system provides a direct and structured way of gathering information. Interviews have the advantages that the level of questioning can be varied to suit the context and that the evaluator can probe the user more deeply on interesting issues as they arise. An interview will usually follow a top-down approach, starting with a general question about a task and progressing to more leading questions (often of the form 'why ...?' or 'what if ...?') to elaborate aspects of the user's response.

Interviews can be effective for high-level evaluation, particularly in eliciting information about user preferences, impressions and attitudes. They may also reveal problems which have not been anticipated by the designer or which have not occurred under observation. When used in conjunction with observation they are a useful means of clarifying an event (compare the post-task walkthrough).

In order to be as effective as possible, the interview should be planned in advance, with a set of central questions prepared. This helps to focus the purpose of the interview which may, for instance, be to probe a particular aspect of the interaction. It also helps to ensure a base of consistency between the interviews of different users. That said, the evaluator may of course choose to adapt the interview form to each user in order to get the most benefit: the interview is not intended to be a controlled experimental technique.

Questionnaires

An alternative method of querying the user is to administer a questionnaire. This is clearly less flexible than the interview technique, since questions are fixed in advance, and it is likely that the questions will be less probing. However, it can be used to reach a wider subject group, it takes less time to administer, and it can be analyzed more rigorously. It can also be administered at various points in the design process, including during requirements capture, task analysis and evaluation, in order to get information on the user's needs, preferences and experience.

Given that the evaluator is not likely to be directly involved in the completion of the questionnaire, it is vital that it is well designed. The first thing that the evaluator must establish is the purpose of the questionnaire: what information is sought? It is also useful to decide at this stage how the questionnaire responses are to be analyzed. For example, do you want specific, measurable feedback on particular interface features, or do you want the user's impression of using the interface?

There are a number of styles of question which can be included in the questionnaire. These include the following:

General These are questions which help to establish the background of the user and his place within the subject population. They include questions about age, sex, occupation, place of residence, and so on. They may also include questions on previous experience with computers which may be phrased as open-ended, multi-choice or scalar questions (see below).

Open ended These ask the user to provide his own unprompted opinion on a question, for example 'Can you suggest any improvements to the interface?' They are useful for gathering general subjective information but are difficult to analyze in any rigorous way, or to compare, and can only be viewed as supplementary. However, they may identify errors or make suggestions that have been missed by the designer. A special case of this type is where the user is asked for factual information, for example how many commands were used.

Scalar These ask the user to judge a specific statement on a numeric scale, usually corresponding to a measure of agreement or disagreement with the statement. For example,

It is easy to recover from mistakes.

Disagree	1	2	3	4	5	Agree

The granularity of the scale varies: a coarse scale (say, from 1 to 3) gives a clear indication of the meaning of the numbers (disagree, neutral and agree). However, it gives no room for varying levels of agreement, and users may therefore be tempted to give neutral responses to statements that they do not feel strongly about but with which they mildly disagree or agree. A very fine scale (say 1 to 10) suffers from the opposite problem: the numbers become difficult to interpret in a consistent way. One user will undoubtedly interpret the scale differently from another. A middle ground is therefore advisable. Scales of 1 to 5 or 1 to 7 have been used effectively. They are fine enough to allow users to differentiate adequately but still retain clarity in meaning. This can help by providing an indication of the meaning of intermediate scalar values.

Multi-choice Here the respondent is offered a choice of explicit responses, and may be asked to select only one of these, or as many as apply. For example,

How do you most often get help with the system (tick one)?

On-line manual	☐
Contextual help system	☐
Command prompt	☐
Ask a colleague	☐

Which types of software have you used (tick all that apply)?

Word processor	☐
Database	☐
Spreadsheet	☐
Expert system	☐
On-line help system	☐
Compiler	☐

These are particularly useful for gathering information on a user's previous experience. A special case of this type is where the offered choices are yes or no.

Ranked These place an ordering on items in a list and are useful to indicate a user's preferences. For example,

Please rank the usefulness of these methods of issuing a command
(1 most useful, 2 next, 0 if not used).

Menu selection	☐
Command line	☐
Control key accelerator	☐

These question types are all useful for different purposes as we have noted. However, in order to reduce the burden of effort on the respondent, and so encourage a high response rate amongst users, it is best to use closed questions, such as scalar, ranked or multi-choice, as much as possible. These provide the user with alternative responses and so reduce the effort required. They also have the advantage of being easier to analyze. Responses can be analyzed in a number of ways, from determining simple percentages for each response, to looking at correlations and factor analysis. For more detail on available methods the reader is referred to the recommended reading list at the end of the chapter.

Worked exercise
You have been asked to compare user performance and preferences with two different learning systems, one using hypermedia (see Chapter 15), the other sequential lessons. Design a questionnaire to find out what the users think of the system. How would you go about comparing user performance with these two systems?

Answer
Assume that all users have used both systems.

Questionnaire

Consider the following questions in designing the questionnaire:

- what information is required?
- how is the questionnaire to be analyzed?

You are particularly interested in user preferences so questions should focus on different aspects of the systems and try to measure levels of satisfaction. The use of scales will make responses for each system easier to compare.

Table 11.3 shows an example questionnaire.

Table 11.3 Questionnaire to compare two systems

PART I: Repeat for each system

Indicate your agreement or disagreement with the following statements (1 indicates complete disagreement and 5 complete agreement).

The system tells me what to do at every point.
Disagree 1 2 3 4 5 Agree

It is easy to recover from mistakes.
Disagree 1 2 3 4 5 Agree

It is easy to get help when needed.
Disagree 1 2 3 4 5 Agree

I always know what the system is doing.
Disagree 1 2 3 4 5 Agree

I always know where I am in the training material.
Disagree 1 2 3 4 5 Agree

I have learned the material well using the system.
Disagree 1 2 3 4 5 Agree

I could have learned the material more effectively using a book.
Disagree 1 2 3 4 5 Agree

I always know how well I am doing.
Disagree 1 2 3 4 5 Agree

PART II: Comparing both systems:

Which system (choose 1) was most

helpful	A	B
effective	A	B
usable	A	B

Please add any comments you have about either system:

To test performance you would design an experiment where two groups of subjects learn the same material using the two systems, and test how well they have learned (using a standard measurable test).

Subjects user group

IV (independent variable) style of learning system

DV (dependent variable) performance (measured as test score)

Design Between-groups design.

11.6 Choosing an evaluation method

As we have seen in this chapter, a range of techniques is available for evaluating an interactive system, at all stages in its development. So how do we decide which methods are most appropriate for our needs? There are no hard and fast rules in this – each method has its particular strengths and weaknesses and each is useful if applied appropriately. However, there are a number of factors which should be taken into account when selecting evaluation techniques. These also provide a way of categorizing the different methods so that we can compare and choose between them. In this final section we will consider these factors.

11.6.1 Factors distinguishing evaluation techniques

We can identify at least eight factors which distinguish different evaluation techniques and therefore help us to make an appropriate choice. These are

- the stage in the cycle at which the evaluation is carried out
- the style of evaluation
- the level of subjectivity or objectivity of the technique
- the type of measures provided
- the information provided
- the immediacy of the response
- the level of interference implied
- the resources required

Design vs. implementation

The first factor to affect our choice of evaluation method is the stage in the design process at which evaluation is required. As we saw earlier in this chapter, it is desirable to include evaluation of some sort throughout the design process. The main distinction between evaluation of a design and evaluation of an implementation is that in the latter case a physical artefact exists. This may be anything from a paper mock-up to a full implementation, but it is something concrete which can be tested. Evaluation of a design on the other hand precedes this stage and seeks instead to provide information to feed the development of the physical artefact.

Roughly speaking, evaluation at the design stage tends to involve design experts only and be analytic, whereas evaluation of the implementation brings in users as subjects and is experimental. There are of course exceptions to this: participatory design (see Chapter 6) involves users throughout the design process, and techniques such as cognitive walkthrough are expert based and analytic but can be used to evaluate implementations as well as designs.

Early evaluation, whether of a design or an early prototype or mock-up, will bring the greatest pay-off since problems can be easily resolved at this stage. As more commitment is made to a particular design in the implementation, it becomes increasingly difficult for changes to be made, no matter what the evaluation suggests. Ironically, the most resources are often ploughed into late evaluations. This

is less profitable and should be avoided, although obviously some evaluation with users is required with a complete, or almost complete, system.

Laboratory vs. field studies

We have already discussed the pros and cons of these two styles of evaluation. Laboratory studies allow controlled experimentation and observation while losing something of the naturalness of the user's environment. Field studies retain the latter but do not allow control over user activity. Ideally the design process should include both styles of evaluation, probably with laboratory studies dominating the early stages and field studies conducted with the new implementation.

Subjective vs. objective

Evaluation techniques also vary according to their objectivity – some techniques rely heavily on the interpretation of the evaluator, others would provide the same information more or less regardless of who is performing the evaluation. The more subjective techniques, such as cognitive walkthrough or think aloud, rely to a large extent on the knowledge and expertise of the evaluator, who must recognize problems and understand what the user is doing. They can be powerful if used correctly and will provide information that may not be available from more objective methods. However, the problem of evaluator bias should be recognized and avoided. One way to decrease the possibility of bias is to use more than one evaluator. Objective techniques, on the other hand, should produce repeatable results which are not dependent on the persuasion of the particular evaluator. Controlled experiments are an example of an objective measure. These avoid bias and provide comparable results but may not reveal the unexpected problem or give detailed feedback on user experience. Ideally, both objective and subjective measures should be used.

Qualitative vs. quantitative measures

The type of measurement provided by the evaluation technique is also an important consideration. There are two main types: *quantitative measurement* and *qualitative measurement*. The former is usually numeric and can be easily analyzed using statistical techniques. The latter is non-numeric and is therefore more difficult to analyze, but can provide important detail which cannot be determined from numbers. The type of measure is related to the subjectivity or objectivity of the technique, with subjective techniques tending to provide qualitative measures and objective techniques, quantitative measures. This is not a hard and fast rule, however. It is sometimes possible to quantify what is in fact qualitative information by mapping it onto a scale or similar measure. A common example of this is in questionnaires where qualitative information is being sought (for example, user preferences) but a quantitative scale is used. This is also common in experimental design where factors such as the quality of the user's performance are used as dependent variables, and measured on a quantitative scale.

Information provided

The level of information required from an evaluation may also vary. The information required by an evaluator at any stage of the design process may range from low-level information to enable a design decision to be made (for example, which font is most readable) to higher-level information, such as 'Is the system usable?' Some evaluation techniques, such as controlled experiments, are excellent at providing low-level information – an experiment can be designed to measure a particular aspect of the interface. Higher-level information can be gathered using questionnaire and interview techniques which provide a more general impression of the user's view of the system.

Immediacy of response

Another factor distinguishing evaluation techniques is the immediacy of the response they provide. Some methods, such as think aloud, record the user's behaviour at the time of the interaction itself. Others, such as post-task walkthrough, rely on the user's recollection of events. Such recollection is liable to suffer from bias in recall and reconstruction, with users interpreting events according to their preconceptions. Recall may also be incomplete. However, immediate techniques can also be problematic, since the process of measurement can actually alter the way the user works.

Intrusiveness

Related to the immediacy of the response is the intrusiveness of the technique itself. Certain techniques, particularly those which produce immediate measurements, are obvious to the user during the interaction and therefore run the risk of influencing the way the user behaves. Sensitive activity on the part of the evaluator can help to reduce this but cannot remove it altogether. Most immediate evaluation techniques are intrusive, with the exception of automatic system logging. Unfortunately this is limited in the information that it can provide and is difficult to interpret.

Resources

The final consideration when selecting an evaluation technique is the availability of resources. Resources to consider include equipment, time, money, subjects, expertise of evaluator and context. Some decisions are forced by resource limitations: it is not possible to produce a video protocol without access to a video camera (and probably editing facilities to boot). However, other decisions are not so clear cut. For example, time and money may be limited forcing a choice between two possible evaluations. In these circumstances, the evaluator must decide which evaluation tactic will produce the most effective and useful information for the system under consideration. It may be possible to use results from other people's experiments to avoid having to conduct new experiments.

Some techniques are more reliant on evaluator expertise than others, for example the more formal analytic techniques. If evaluator expertise is limited it may be more practical to use heuristic methods than analytic methods which require understanding of user goal structures and so on.

Finally the context in which evaluation can occur will influence what can be done. For practical reasons it may not be possible to gain access to the intended users of a system (if it is a general system, for example) or it may not be feasible to test the system in its intended environment (for example, a system for a space station or a defence system). In these circumstances simulations must be used.

11.6.2 A classification of evaluation techniques

Using the factors discussed in the previous section we can classify the evaluation techniques we have considered in this chapter. This allows us to identify the techniques which most closely fit our requirements. Table 11.4 shows the classification for analytic techniques, Table 11.5 for experimental and query techniques and Table 11.6 for observational techniques.

Table 11.4 Classification of analytic evaluation techniques

	Cognitive walkthrough	Heuristic evaluation	Review based	Model based
Stage	Throughout	Throughout	Design	Design
Style	Laboratory	Laboratory	Laboratory	Laboratory
Objective?	No	No	As source	No
Measure	Qualitative	Qualitative	As source	Qualitative
Information	Low level	High level	As source	Low level
Immediacy	N/A	N/A	As source	N/A
Intrusive?	No	No	No	No
Time	Medium	Low	Low–medium	Medium
Equipment	Low	Low	Low	Low
Expertise	High	Medium	Low	High

Table 11.5 Classification of experimental and query evaluation techniques

	Experiment	Interviews	Questionnaire
Stage	Throughout	Throughout	Throughout
Style	Laboratory	Lab/field	Lab/field
Objective?	Yes	No	No
Measure	Quantitative	Qualitative/ quantitative	Qualitative/ quantitative
Information	Low/high level	High level	High level
Immediacy	Yes	No	No
Intrusive?	Yes	No	No
Time	High	Low	Low
Equipment	Medium	Low	Low
Expertise	Medium	Low	Low

Table 11.6 Classification of observational evaluation techniques

	Think aloud[a]	Protocol analysis[b]	Post-task WT[c]
Stage	Implementation	Implementation	Implementation
Style	Lab/field	Lab/field	Lab/field
Objective?	No	No	No
Measure	Qualitative	Qualitative	Qualitative
Information	High/low level	High/low level	High/low level
Immediacy	Yes	Yes	No
Intrusive?	Yes	Yes[d]	No
Time	High	High	Medium
Equipment	Low	High	Low
Expertise	Medium	High	Medium

[a]Assuming a simple paper and pencil record.
[b]Including video, audio and system recording.
[c]WT = walkthrough.
[d]Except system logs.

The classification is intended as a rough guide only – some of the techniques do not fit easily into such a classification since their use can vary considerably.

11.7 Summary

Evaluation is an integral part of the design process and should take place throughout the design life cycle. Its aim is to test the functionality and usability of the design and to identify and rectify any problems. It can take place in the laboratory or in the user's workplace, and may involve active participation on the part of the user.

A design can be evaluated before any implementation work has started, to minimize the cost of early design errors. Most techniques for evaluation at this stage are analytic and involve using an expert to assess the design against cognitive and usability principles. Previous experimental results and modelling approaches can also provide insight at this stage. Once an artefact has been developed (whether a prototype or full system), experimental and observational techniques can be used to get both quantitative and qualitative results. Query techniques provide subjective information from the user.

The choice of evaluation method is largely dependent on what is required of the evaluation. Evaluation methods vary in the stage at which they are commonly used and where they can be used. Some are more subjective than others and provide qualitative rather than quantitative measures. Some provide immediate information while others get feedback after the event. However, the more immediate methods also tend to intrude most seriously on the interaction. Finally some require more resources in terms of time, equipment and expertise than others.

Exercises

11.1 In groups or pairs, use the cognitive walkthrough example, and what you know about user psychology (see Chapter 1), to discuss the design of a computer application of your choice (for example, a word processor or a drawing package). (**Hint:** Focus your discussion on one or two specific tasks within the application.)

11.2 What are the benefits and problems of using video in experimentation? If you have access to a video recorder, attempt to transcribe a piece of action and conversation (it does not have to be an experiment – a soap opera will do!). What problems did you encounter?

11.3 In Section 11.5.1, we saw that the observed results could be the result of interference. Can you think of alternative designs that may make this less likely? Remember that individual variation was very high, so you *must* retain a within-groups design, but you may perform more tests on each subject.

11.4 Choose an appropriate evaluation method for each of the following situations. In each case identify

1. The subjects.
2. The technique used.
3. Representative tasks to be examined.
4. Measurements that would be appropriate.
5. An outline plan for carrying out the evaluation.

- ❑ You are at an early stage in the design of a spreadsheet package and you wish to test what type of icons will be easiest to learn.
- ❑ You have a prototype for a theatre booking system to be used by potential theatre-goers to reduce queues at the box office.
- ❑ You have designed and implemented a new game system and want to evaluate it before release.
- ❑ You have developed a group decision support system for a solicitor's office.
- ❑ You have been asked to develop a system to store and manage student exam results and would like to test two different designs prior to implementation or prototyping.

11.5 Complete the cognitive walkthrough example for the video remote control design.

Recommended reading

❑ C. Robson, *Experiment, Design and Statistics in Psychology*, 2nd edition, Penguin, 1985.
An accessible introduction to statistics and experimental design and analysis for the uninitiated, using worked examples throughout.