Human–Computer Interaction

Teacher's Notes

Alan J. Dix, Janet E. Finlay
University of York, UK
Gregory D. Abowd
Carnegie Mellon University, USA
Russell Beale
University of Birmingham, UK

Introduction

This set of teacher's notes accompanies the book "Human–Computer Interaction" and is intended to support teachers using the book as a core course text. It includes solutions to and suggestions for the use of the exercsies in the book, together with additional project material.

In addition, a set of master slides are available which can be used to produce overhead projector foils. These cover the key points for each chapter, making them ideal for a lecture-based course. Please contact the publisher for further details.

We have tried to design the book and additional material to be flexible and to support effective teaching of HCI. Obviously in any such venture there is room for improvement. We therefore welcome comments on both the book and this set of notes which will allow us to improve future editions. We are particularly keen to hear suggestions for material that you would find useful that we have not included and, conversely, to know if we have included anything which you do not find useful. Please send your comments to us care of the publishers.

Course structure

Human–Computer Interaction is a subject which is by definition practical and whch lends itself well to novel teaching methods. It is our intention that the book and these notes be able to support both traditional lecture-style courses and those based mainly on project work.

In either case we strongly recommend that students be given the opportunity to do some practical work, both in experimenting with and evaluating existing available systems and in designing their own interfaces. HCI cannot be taught exclusively through lectures and books and requires some "hands-on" experience. These notes offer examples of the types of project that can be used to provide such experience. As far as possible we have not assumed the availability of any particular resource so that teachers can adapt them to what is available. However, if possible, we suggest that all students be given the chance to experiment with both graphical and command based applications and to use a prototyping system such as Hypercard or Visual Basic. These allow the student to develop their own mock up interfaces.

If such tools are unavailable, drawing packages can be used to design individual screens but have the disadvantage of being static.

For a traditional lecture-based course we suggest the use of the overhead projector slides (selectively if necessary) backed up by related reading and exercises from the book and at least one practical design project (see Additional Projects). If possible other practicals can be included as well. The design project should encourage the student to use the modelling approaches and to evaluate their design.

A project-based course can be designed primarily around the exercises, with suggested recommended related reading. Such a course demands a certain amount of commitment from students and may therefore be best suited for option courses and other advanced courses where the students are motivated towards HCI. Obviously, the choice of exercises and projects will depend largely on the resources and time available but the following is a suggested programme which assumes 10 2-hour practicals and an individual or group project. In selecting exercises we have tried to include some that involve research, some that involve observation and some that require practical application. Clearly some of these can be done in the student's own time. If this is the case the extra practical time can be spent introducing different interactive applications and tools.

Week	Chapter	Exercises
1	1, 2, 3	1.2, 2.3, 3.1 or Project 1
2	4, 5	4.1, 5.1
3	6	6.1, 6.3
4	7	7.5, 7.7
5	8	8.1, 8.2 or Project 2
6	9, 10	9.4, 10.2
7	11	11.1, 11.3, 11.4 or Project 3
8	12	12.2, 12.4
9	13, 14	13.2, 14.2
10	15	15.2, 15.3

At least one design project should be attempted. Depending on time, this can be done instead of the exercises at the points suggested, or as an end of term project in addition. Projects 2 and 3 provide most scope for incorporating different aspects of the course.

An additional resource which we recommend to aid course design is the SIGCHI Curriculum Development Group's report [3]. This provides suggestions for curricula for different groups of students and for different purposes. The book covers most of material outlined in the SIGCHI report and can be adapted for use with most of their suggested curricula.

Exercises and projects

The rest of this booklet contains solutions to exercises from the book and 3 additional design projects which can be used to bring together the techniques from the different stages of design.

The exercises provided in the book are of three main types: those that require factual answers (some of which may require additional reading or research), those that provide practice in using the techniques described, and those which encourage the student to observe and evaluate existing designs. Consequently the exercises solutions also differ. For factual questions pointers are given to enable the teacher to guide the student in the right direction.

However, students may in the process of their research uncover additional information on a subject and this should be encouraged. In the case of practice questions example solutions are given. Again variations on these are sometimes possible (individual solutions indicate this) but the solution given here can be provided to the student as an example solution.

Observational exercises are usually small practical projects which require the student to interact with an application or watch someone else do so. Here the solutions suggest factors that the student should consider in the observation and hints to the teacher as to how to best encourage effective use of such exercises.

Some exercises refer to sections or diagrams in the book itself, others to diagrams internal to these notes. The former are referred to using the same notation as the book; the latter are distinguished by the use of the letter x i.e. Figure 12.1 refers to Figure 12.1 in the book while Figure 12.x1 refers to a Figure in the notes.

The three additional projects are more extensive than any of the exercises and can be used as end of term projects or assessments. They are not included in the book itself for this reason.

They require knowledge of more than one aspect of design and are intended to demonstrate how the different stages of design fit together. Each project is described in full, together with any support material, and hints to the teacher on how to use the projects are given. These projects can be reproduced for the purposes of teaching on courses where the core text is “Human–Computer Interaction”.

Chapter 1

The Human

Exercise 1.1

Suggest ideas for an interface which uses the properties of sound effectively.

Answer

This is an opportunity for the student both to use his/her imagination and/or to do a literature survey (starting with the references in Chapter 15). One possibility for tackling the exercise is to encourage the student to think how sound could be added to an application with which s/he is familiar.

Speech sounds can obviously be used to convey information. This is useful not only for the visually impaired but also for any application where the user's attention has to be divided (for example, power plant control, flight control etc.). Uses of non-speech sounds include

- Attention — to attract the user's attention to a critical situation or to the end of a process for example.
- Status information — continuous background sounds can be used to convey status information. For example, monitoring the progress of a process (without the need for visual attention).
- Confirmation — a sound associated with an action to confirm that the action has been carried out. For example, associating a sound with deleting a file.
- Navigation — using changing sound to indicate where the user is in a system. For example, what about sound to support navigation in hypertext?

Exercise 1.2

> Devise experiments to test the properties of (i) short-term memory (ii) long-term memory, using the experiments described in this chapter to help you. Try out your experiments on your friends. Are your results consistent with the properties described in this chapter?

Answer

The aim of this exercise is to get the student to think about experimental design. The experiments devised can effectively be repeats of the originals. Chapter 11 can be used for reference on experimental design.

(i) Short term memory (STM)

The student should first choose an aspect to investigate, for example, digit span, recency effect, decay.

Example solution: STM decay

Subjects ideally selected to represent population, more probably undergraduate students (try to get a range of academic subjects). Sample size: 10+

Experiment split subjects into two groups. Each subject studies list of 15-20 words (could try with both nonsense words and actual words to see any difference). Subject has to recall list either (a) immediately or (b) after 20 second delay. Measure the number (or percentage) of the words remembered correctly. A within groups design can be used to avoid individual bias or group variation (as long as different lists are used for each attempt).

independent variable — delay in recall
dependent variable — number correctly recalled.

Group (b) should be given a task to do during the delay period in order to avoid rehearsal. If possible this task should occupy a different channel to minimise interference, e.g., a visual recognition task.

Hypothesis Those in (b) will perform worse than those in (a) since STM will decay.

Analysis graphs to see decay. T test *(ii) Long term memory (LTM)*

The student should first choose an aspect to investigate, for example, the total time hypothesis or the distribution of practice effect.

Example solution: distribution of practice effect

Subjects as above. Should have no prior experience of the skill to be learned.

Experiment split subjects into 3 groups. Each subject must learn a skill (for example short hand or typing — must be measurable). Group A learns for 1 hour a week for 6 weeks. Group B learns for 2 hours a week for 3 weeks. Group C learns for 3 hours a week for 2 weeks. After each group's training is complete the subjects are tested and the number of mistakes made noted.

independent variable — style of learning
dependent variable — accuracy

Between groups design.

Hypothesis Group A will be best (due to the distribution of practice effect)

Analysis ANOVA

(N.B. This one is not easy to run but could be done with cooperation from friends)

Exercise 1.3

Identify the goals and operators involved in the problem 'delete the second paragraph of the document' on a word-processor. Now use a word-processor to delete a paragraph and note your actions, goals and sub-goals. How well did they match your earlier description?

Answer

Assume you have a document open and you are at some arbitrary position within it. You also need to decide which operators are available and what their preconditions and results are. Based on an imaginary word processor we assume the following operators (you may wish to use your own WP package):

Operator	Precondition	Result
delete_paragraph	cursor at start of paragraph	paragraph deleted
move_to_paragraph	cursor anywhere in document	cursor moves to start of next paragraph (except where there is no next paragraph when no effect)
move_to_start	cursor anywhere in document	cursor at start of document

Goal: *delete second paragraph in document*

Looking at the operators an obvious one to resolve this goal is delete_paragraph which has the pre-condition 'cursor at start of paragraph'. We therefore have a new sub-goal: *move_to_paragraph*. The pre-condition is 'cursor anywhere in document' (which we can meet) but we want the second paragraph so must initially be in the first.

We set up a new subgoal, *move_to_start*, with pre-condition 'cursor anywhere in document' and result 'cursor at start of document'. We can then apply *move_to_paragraph* and finally *delete_paragraph*.

We assume some knowledge here (that the second paragraph is the paragraph after the first one).

Exercise 1.4

Observe skilled and novice operators in a familiar domain, for example, touch and 'hunt-and-peck' typists, expert and novice game players, or expert and novice users of a computer application. What differences can you discern between their behaviours?

Answer

This is an exercise in observation. The student should think about skill acquisition, proceduralization, chunking etc. Is there any evidence of this in practice? How do the groups differ (speed, error, style, strategy)? Do the differences suggest different skill levels.

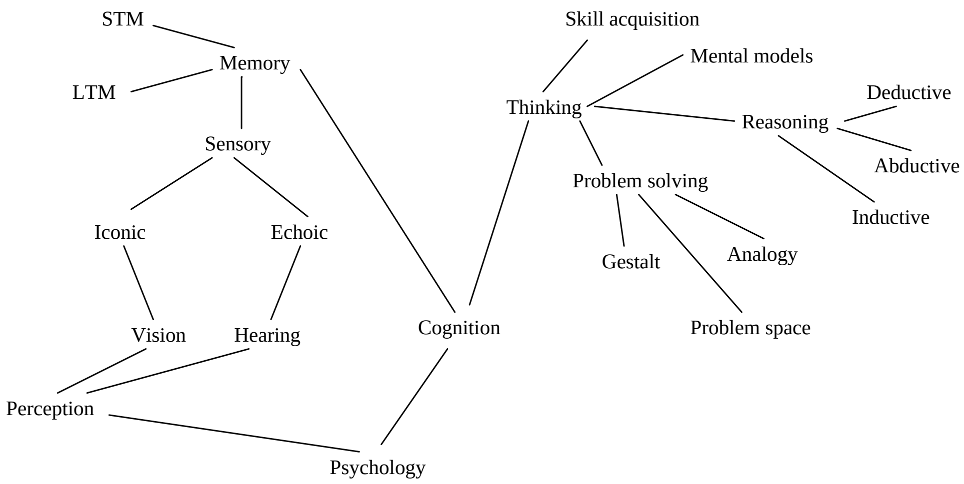

Figure 1.x1 *The top-down view*

Exercise 1.5

Produce a semantic network of the main information in this chapter.

Answer

This network is potentially huge so it is probably unnecessary to devise the whole thing! Encourage the student to be selective. One helpful way to tackle the exercise is to approach it in both a top down and a bottom up manner. Top down will give you a general overview of topics and how they relate; bottom up can fill in the details of a particular field. These can then be 'glued' together to build up the whole picture. Perhaps a group of students could tackle the problem together, each taking one part of it. We will not provide the full network here but will give examples of the level of detail anticipated for the overview and the detailed versions. In the overview we have not included labels on the arcs for clarity.

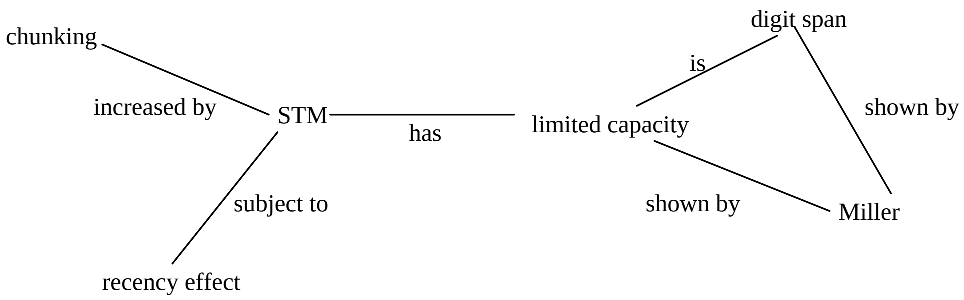

Figure 1.x2 *The bottom-up view*

Chapter 2

The Computer

The aim of this chapter is to give the students a general view of the capabilities of typical computer systems insofar as they impact upon the users. This is obviously particularly relevant for students from a human sciences background as they may be unaware of the basic components of computer equipment. However, many computer scientists are equally ignorant when it comes to the gross capabilities of standard systems. They may be able to tell you that a bubble sort is an order n^2 algorithm whereas quicksort is order $n \log n$, but may have no idea how long a typical database would take to sort 1000 records!

Exercise 2.1

What is the basic architecture of a computer system?

Answer

Some students might have done a 'computer architectures' course, but they should *not* give the stock answer for such a question in that context. Instead, they should be assessing the architecture from the point of view of the user. The material for this question is scattered throughout the chapter. However, students should also be directed to personal computer magazines where adverts and articles will give them some idea of typical capabilities …and costs. It might also prompt some questions: just what is the difference to the user between a 16ms and a 20ms disk drive?

The example answer below gives the general style, although more detail would be expected of a full answer. In particular, the students should be encouraged to have a *feel* for capacities either as ball park figures or in terms of typical capabilities (seconds of video, pages of text).

Example

The basic architecture of a computer systems consists of the computer itself (with associated memory), input and output devices for user interaction and various forms of hard copy devices. (Note, the computer science answer regards output to the user and output to a printer as essentially equivalent. This is not an acceptable user centred view.)

A typical configuration of user input/output devices would be a screen with a keyboard for typing text and a mouse for pointing and positioning. Depending on circumstance, different pointing devices may be used such as lightpen (for more direct interaction) or a trackball (especially on portable computers).

The computer itself can be considered as composed of some processing element and memory. The memory is itself divided into short term memory which is lost when the machine is turned off and permanent memory which persists.

Exercise 2.2

> How do you think new, fast, high-density memory devices and quick processors have influenced recent developments in HCI? Do they make systems any easier to us? Do they expand the range of applications of computer systems?

Answer

Arguably it is not so much the increase in computer power as the decrease in the cost of that power which has had the most profound effect. Because, 'ordinary' users have powerful machines on their desktops it has become possible to view that power as available for the interface rather than hoarded for number crunching applications.

Modern graphical interaction consumes vast amounts of processing power and would have been completely impossible only a few years ago. There is an extent to which systems have to run faster to stay still, in that as screen size, resolution and colour range increase, so does the necessary processing power to maintain the 'same' interaction. However, this extra processing is not really producing the same effect, screen quality is still a major block on effective interaction.

The increase in RAM means that larger programs can be written, effectively allowing the programmer 'elbow room'. This is used in two ways: to allow extra functionality and to support easier interaction. Whether the former really improves usability is debatable — unused functionality is a good marketing point, but is of no benefit to the user. The ease of use of a system is often determined by a host of small features, such as the appropriate choice of default options. These features make the interface seem 'simple', but make the program very complex …and large.

Certainly the availability of 'elbow room', both in terms of memory and processing power, has made such features possible.

The increase in both short term (RAM) and long term (disks and optical storage) has also removed many of the arbitrary limits in systems: it is possible to edit documents of virtually unlimited size and to treat the computer (suitably backed up) as one's primary information repository.

Some whole new application areas have become possible because of advances in memory and processing. For example, most applications of multi-media, for example voice recognition and online storage and capture of video and audio, require enormous amounts of processing and/or memory. In particular, large optical storage devices have been the key to electronic document storage whereby all paper documents are scanned and stored within a computer system. In some contexts such systems have completely replaced paper based filing cabinets.

Exercise 2.3

What input and output devices would you use for the following systems? For each, compare and contrast alternatives, and if appropriate indicate why the conventional keyboard, mouse and c.r.t screen may be less suitable.

a) portable word processor
b) tourist information system
c) tractor-mounted crop-spraying controller
d) air traffic control system
e) worldwide personal communications system
f) digital cartographic system

Answer

Whereas question 2.1 focuses on 'typical' systems, the emphasis here is on the diversity of different devices needed for specialised purposes. During classes and practicals the students can be exposed to a wide variety of input and output devices. They can also be encouraged to 'collect' devices, that is, to watch out for shop tills, bank tellers, taxi meters, lift buttons, domestic appliances etc.

a) portable word processor
The determining factors are size, weight and battery power. However, remember the purpose, this is a word processor not an address book or even data entry device.

- LCD screen — low power requirement

- Trackball or stylus for pointing
- Real keyboard – you can't word process without a reasonable keyboard and stylus handwriting recognition is not good enough.
- Small, low power bubble-jet printer — although not always necessary, this makes the package stand alone. It is probably not so necessary that the printer have large battery capacity as printing can probably wait until a power point is found.

b) tourist information system
This is likely to be in a public place. Most users will only visit the system once, so the information and mode of interaction must be immediately obvious.

- Touch screen only — easy and direct interaction for first time users (see also Chapter 3).
- NO mice or styluses — in a public place they wouldn't stay long!

c) tractor-mounted crop-spraying controller.
A hostile environment with plenty of mud and chemicals. Requires numerical input for flow rates etc., but probably no text.

- Touch sensitive keypad — , ordinary keypads would get gunged up.
- Small dedicated LED display (LCD often can't be read in sunlight and large screens are fragile.
- Again no mice or styluses — they would get lost.

d) air traffic control system
The emphasis is on immediately available information and rapid interaction. The controller cannot afford to spend time searching for information, all frequently used information must be readily available.

- Several specialised displays — including overlays of electronic information on radar.
- Lightpen or stylus — high precision direct interaction.
- Keyboard — for occasional text input, but consider making it fold out of the way.

e) worldwide personal communications system
Basically a super mobile phone! If is to be kept on hand all the time it must be very light and pocket sized. However, to be a 'communications' system one would imagine that it should also act as a personnel address/telephone book etc.

- Standard telephone keypad — the most frequent use
- Small dedicated LCD display — low power, specialised functions.

- Possibly stylus for interaction — it allows relatively relatively rich interaction with the address book software, but little space.
- A 'docking' facility — the system itself will be too small for a full sized keyboard (!), but you won't want to enter in *all* your addresses and telephone numbers by stylus!

f) digital cartographic system
This calls for very high precision input and output facilities. It is similar to CAD in terms of the screen facilities and printing, but in addition will require specialised data capture.

- Large high resolution colour VDU (20 inch or bigger) — these tend to be enormously big (from back to front). LCD screens, although promising far thinner displays in the long term, cannot at present be made large enough.
- Digitising tablet — for tracing data on existing paper maps. It could also double up as a pointing device for some interaction.
- Possibly thumbwheels — for detailed pointing and positioning tasks.
- Large format printer — indeed VERY large an A2 or A1 plotter at minimum.

Chapter 3

The Interaction

Exercise 3.1

Choose two of the interaction styles (described in Section 3.3) that you have experience of using. Use the interaction framework to analyze the interaction involved in using these interface styles for a database selection task. Which of the distances is greatest in each case?

Answer

There is no single answer for this exercise, so we will provide an example of the style of answer that is suitable and the level of analysis which is appropriate. The students should be aware that although the term distance is used, we have not associated any real measures to any of the translations in the interaction framework. As a result, this analysis can only be informal and at this point is mainly informed by the student's intuition and experience with various interaction styles.

As was stated in Section 3.2.3, assessment of any interaction with the interaction framework can only be relative to some task. For this example we will choose a common database selection task — that of selecting records from an online library database. The two interaction styles we will analyze are a natural language interface and a command line interface.

The task is to select a set of references from the library database that satisfy some search criteria. Once the task has been formulated in the user's task language (for instance, the user wants to see all of the books written by Alan Dix since 1990), that task must be articulated in the input language. A natural language interaction style would allow the user simply to type in the selection query exactly as they think of it. The articulation distance is small because it is both easy to articulate (possibly even easier if a spoken interface is provided rather than typing) and the coverage is total (the user is allowed to articulate anything as a query). On the other hand, for a command line interface, the limited vocabulary of the input language makes it more difficult for the user to articulate a task even though the limited language provides complete coverage in terms of possible queries allowed. The real difficulty for a natural language interface is how the system translates the input expression into the actual query that accesses the library records. This performance translation would be much easier for the command line interface since it may not even require any translation of an input expression, that language having already been constructed with the database engine in mind. Therefore, we can see that for a natural language interface, the performance distance is greatest, whereas for a command line interface it is the articulation distance which is greatest.

But the above analysis only really deals with the execution translations. On the evaluation side, a natural language interface must try to present the results of the database query in the form in which the user phrased the question. This could in general be a difficult translation for the system as it attempts to answer questions in the style in which an arbitrary user has posed that question. Having accomplished that, the observation by the user should be easy to perform. For a command line interface, there is no guarantee that the result of the query will be automatically displayed and the user may have to explicitly request a display (and they may have to express how the display be formatted). Neglecting that point, presentation by the system is made easier as the output language can be very constrained. Observation is made more difficult as the user must translate the output into the terms of their original task formulation. For example, having asked for books by Alan Dix published after 1990, the user may have a difficult time locating author name and year of publication to determine if the resulting records match their expectations. For evaluation, a natural language interface has a greater presentation distance and a command line interface a greater observation distance. In general, therefore, we would expect that a natural language interface would be easier from the user's perspective but more difficult from the system builder's perspective. The opposite should hold for a command language interface.

There are some issues that we haven't addressed in this example, such as displaying a large set of records that satisfy the query and being able to reuse the output of one query as the input to another to easily compound searches. Since the performance translation is so difficult for a natural language interaction style it is important for a natural language interface to present the results of the query in such a way that the user is able to determine if the system understood the original query in the way the user intended. This would involve the presentation translation both reiterating the user's query and the selected records simultaneously. In our example, since the user was interested in the author and date of publication (Dix, after 1990), it would help if that information was prominently presented in the result set.

We also have not considered what effect experience with the system provides. As users become more comfortable with the syntax and semantics of a command language, its perceived difficulty will decrease. Also, a verbose natural language output could limit the number of records from a result set that could be displayed. The moral of the story is that despite their intuitive allure, such informal analyses as suggested by this exercise cannot be the last word on analysis of an interactive system. Ultimately, our judgements must be made more precise and concrete.

Exercise 3.2

Find out all that you can about natural language interfaces. Are there any successful systems? For what applications are these most appropriate?

Answer

This exercise is intended to encourage the student to do some personal research in the form of a brief literature survey. Pointers can be given to appropriate places to begin such a search. For example, general text books on Artificial Intelligence will include basic information on natural language processing and famous systems such as SHRDLU; proceedings from conferences such as the AAAI and specialist journals will have more up to date research papers in the area; proceedings of HCI conferences such as CHI, Interact and HCI will include natural language systems particularly geared towards the interface. Other likely sources are popular journals such as BYTE and personal computer magazines which are likely to review commercial systems. The student's response to the second part of the question will depend upon what is unearthed, but it is likely that the systems that they find out about operate in very constrained domains and that the natural language used is restricted. There are as yet no general purpose natural language interfaces.

Exercise 3.3

What influence does the social environment in which you work have on your interaction with the computer? What effect does the organization (commercial or academic) to which you belong have on the interaction?

Answer

The aim of this exercise is to get the student to explore the social and environmental influences which effect interaction, often without the user being aware of them. The particular influences will vary from environment to environment but the student should be encouraged to consider some or all of the following.

- work context — is the work place shared? are the machines shared?
- peer pressure — is there pressure to compete or impress?
- management pressure — is there pressure to achieve? Is the interaction carried out in the presence of management?
- motivation - what motivates the interaction? Does this encourage or discourage experimentation?

- organizational goals — what is the objective of the organization? (profit? education? etc.) How does this effect the interaction?
- organizational decision making - who determines the systems that you use? Do you have any choice or influence? Does this influence the way you interact with the system?

In each case the student should discuss what influence this may have on the interaction. It may be helpful to consider other possible environments in order to identify how the interaction would differ under these different circumstances. For example, if the student currently shares a machine with colleagues, would his/her interaction practice change if s/he was given a private machine? Chapter 14 also discusses the influence of groups of workers within an organization on the an interaction, and is suggested as further reading material on this topic.

Chapter 4

Usability Paradigms and Principles

Exercise 4.1

Look up and report back guidelines for the use of colour. Be able to state the empirical psychological evidence which supports the guidelines. Do the guidelines conflict with any other known guidelines? Which principles of interaction do they support?

Answer

(Sample) There are many examples of guidelines for the use of colour in the literature. Here are three good sources:

- Brown, C. Marlin, *Human-Computer Interface Design Guidelines*, Ablex, 1988.
- Mayhew, Deborah J., *Principles and Guidelines in Software User Interface Design*, Prentice-Hall, 1992.
- Sun Microsystems, Inc., OPEN LOOK *Graphical User Interface Application Style Guidelines*, Addison-Wesley, 1990.

Taking an example from Mayhew, we have the following design guideline for the use of colour as an informational cue for the user (e.g., to inform the user that a string of text is a warning or error message):

Do not use colo(u)r without some other redundant cue

Mayhew provides 3 reasons which empirically support this guideline:

1. Colour may not be available on all machines on which the system is to be implemented. Therefore, if use of colour is the only means to convey some important information to the user, then that information will be lost in a monochrome (no colour) system. Redundant colour coding will allow for portability across different computing platforms.

2. Empirical evidence shows that 8% of the (general) male population and 0.4% of the female population has some colour deficiency, so they cannot accurately recognize or distinguish between various colours. Again, if colour is the only means for conveying some information, this significant portion of the user population will be slighted.

3. It has been shown that redundant colour coding enhances user performance

This guideline supports several of the principles discussed in this chapter:

Substitutivity The system is able to substitute colour-coded information and other means (e.g., text, sound) to represent some important information. We could turn the argument around and suggest that then the user be able to provide colour input (by selecting from a palette menu) or other forms of input to provide relevant information to the system.

Observability This principle is all about the system being able to provide the user with enough information about its internal state to assist the user's task. Relying strictly on colour-coded information, as pointed out above, could reduce the observability of a system for some users.

Synthesis If a change in colour is used to indicate the changing status of some system entity (perhaps a change in temperature above a threshold value is signaled by an icon becoming red), those who cannot detect the change in colour would be deprived of this information. Synthesis is about supporting the user's ability to detect such significant changes, especially when they are a result of previous user actions.

There is no evidence of existing guidelines which this particular guideline for colour violates.

Another example of a colour guideline (found in all three of the above references) is the demand to consider cultural information in the selection of particular colours. For example, Mayhew states that Western cultures tend to interpret green to mean go or safe, red to mean stop, on, hot or emergency and blue to mean cold or off. Using colour to suggest these kinds of meanings is in support of the familiarity principle within learnability. However, in other cultures different meanings may be associated with these colours and consistent use of colour (another guideline) might lead to confusion. Hence, strict adherence to this guideline would suggest a violation of the consistency of colour application guideline. However, if consistency is applied relative to the meaning of the colour (as opposed to its actual colour), this guideline would not have to conflict.

Exercise 4.2

What was the problem with the synthesis example comparing a command language interface with a visual interface? Can you suggest a fix to make a visual interface really immediately honest?

Answer

In this example to demonstrate the principle of synthesizability within learnability, it was stated that a visual interface to a file management system provided immediate information about the changed location of some file after a move operation performed by the user. In contrast, a command language interface requires the user to remember the directory to which a file was moved and explicitly issue commands to browse the directory to verify that the file has been moved. To really be sure that a move occurred, the user would have to also browse the original directory to determine that the file is no longer there. The fallacy in this argument is that visual file management systems do always provide information about the new whereabouts of a moved file. To take a counterexample using the Macintosh example from the text, if a file is moved from one open folder (in which the contents of the folder is revealed to the user) to a closed folder (contents not revealed) then the location of the moved file is not indicated to the user unless she remembers to open up the destination folder to reveal its contents. This is an example of eventual honesty and not immediate, as the example suggests.

We could "fix" this problem of eventual honesty for the visual system by demanding that the destination folder be an open folder (probably too restrictive, given the limited screen size) or by having the destination folder temporarily open up to reveal that the file is now located within it. This last suggestion is also a bit tricky, for we would still want to determine that the file no longer resides in the original folder, so we would have to be sure that the new folder does not obstruct the view of the old folder. In practice, this might be too difficult to guarantee in general.

Exercise 4.3

It has been suggested in this chapter that consistency could be considered a major category of interactive principles, on the same level as learnability, flexibility and robustness. If this had been the case, which principles discussed in this chapter would appear in support of consistency?

Answer

As mentioned in the discussion of consistency, it can take many forms because consistency is usually referred to relative to some other feature of the interaction between user and system. Mentioned already in the text we have consistency related to the following principles:

Familiarity consistency with respect to prior real-world experience

Generalizability consistency with respect to experience with the same system or set of applications on the same platform

In addition, we could interpret some other principles as contributors to consistency:

Affordance consistency with understood intrinsic properties of an object, so a soft button on the screen should allow us to always "push" on it to select some action

Predictability consistency of system response with user's expectation, given the user has some information about past interaction history

Substitutivity consistent permission from system to allow use of equivalent values for input and output

Commensurate effort consistency of effort with respect to doing and undoing tasks

Response time stability consistency of system response for similar actions

Some other principles for consistency from the text and elsewhere:

- Consistency can be relative to the form of input/output expressions relative to user's conceptual model of the system. An example in the text involve using keys whose relative positions are similar to commands for the systems (any set of four typewriter keys which form a diagonal to indicate up, down, left and right information for an input command).
- As discussed in the exercise on colour, consistency can be with respect to social or cultural conventions (e.g., using red to indicate stop or hot, green for go, blue for cool).

Exercise 4.4

Discuss the ways in which a full-page word-processor is or is not a direct manipulation interface for editing a document using Shneiderman's criteria. What features of a modern word processor break the metaphor of composition with pen (or typewriter) and paper?

Answer

We will answer the first question by evaluating the word-processors relative to the criteria for direct manipulation given by Shneiderman.

- *Visibility of the objects of interest* The most important objects of interest in a word-processor are the words themselves. Indeed, the visibility of the text on a continual basis was one of the major usability advances in moving from line-oriented to display-oriented editors. Depending on the user's application, there may be other objects of interest in word-processing that may or may not be visible. For example, are the margins for the text on screen similar to the ones which would eventually printed? Is the spacing within a line and the line-breaks similar? Are the different fonts and formatting characteristics of the text visible (without altering the spacing)? Expressed in this way, we can see the visibility criterion for direct manipulation as very similar to the criteria for a WYSIWYG (What You See Is What You Get) interface.

- *incremental action at the interface with rapid feedback on all actions* We expect from a modern word-processor that characters appear in the text as we type them it at the keyboard, with little delay. If we are inserting text within a paragraph, we might also expect that the format of the paragraph adjust immediately to accommodate the new changes. Various word processors do this reformatting automatically, whereas others do it occasionally or only at the explicit request of the user. One of the other important actions which requires incremental and rapid feedback is movement of the insertion point, usually by means of arrow keys. If there is a significant delay between the input command to move the insertion point down one line and the actual movement of the cursor on screen, it is quite possible that the user will "overshoot" the target when repeatedly pressing the down-arrow key to move down a few lines on the screen.

- *Reversibility of all actions, so that users are encouraged to explore without severe penalties* Single step undo commands in most word-processors allow the user to recover from the last action performed. One problem with this is that the user must recognize the error before doing any other action. More sophisticated undo facilities allow the user to retrace back more than one command at a time. The kind of exploration this reversibility provides in a word-processor is best evidenced with the ease of experimentation that is now available for formatting changes in a document (fonts types and sizes and margin changes). One problem with the ease of exploration is that emphasis may move to the look of a document rather than what the text actually says (style over content).

- *Syntactic correctness of all actions, so that every operation is a legal operation* WYSYWYG word-processors usually provide menus and buttons which the user uses to articulate many commands. These interaction mechanisms serve to constrain the input language to only allow legal input from the user. Document production systems, such as troff, TeX, and Scribe, force the user to input textual commands (which may be erroneously entered by the user) to achieve desired formatting effects.

- *Replacement of complex command languages with actions to manipulate directly the visible objects* The case for word processors is similar to that described above for syntactic correctness. In addition, operations on portions of text are achieved many times by allowing the user to directly highlight the text with a mouse (or arrow keys). Subsequent action on that text, such as moving it or copying it to somewhere else, can then be achieved more directly by allowing the user to "drag" the selected via the mouse to its new location.

To answer the second question concerning the drawback of the pen (or typewriter) metaphor for word-processing, we refer to the discussion on metaphors in Section 4.2.6. The example there compares the functionality of the space key in typewriting versus word-processing. For a typewriter, the space key is passive, it merely moves the insertion point one space to the right. In a word processor, the space key is active, as it inserts a character (the space character) into the document. The functionality of the typewriter space key is produced by the movement keys for the word-processor (typically an arrow key pointing right to move forward within one line). In fact, much of the functionality that we have come to expect of a word-processor is radically different from that expected of a typewriter, so much so that the typewriter as a metaphor for word-processing is not all that instructive. In practice, modern typewriters have begun to borrow from word-processors when defining their functionality!

Chapter 5

The Design Process

Exercise 5.1

Starting with some of the principles outlined in Chapter 4, provide a usability specification for an electronic meetings diary or calendar. First identify some of the tasks that would be performed by a user trying to keep track of future meetings, and then complete the usability specification assuming that the electronic system will be replacing a paper-based system. What assumptions do you have to make about the user and the electronic diary in order to create a reasonable usability specification?

Answer

This exercise could be easily extended to a small project which would involve the design of such an electronic diary or calendar. The purpose of this smaller usability engineering exercise is to show how usability goals can be formulated early on to drive the design activity. We will select two of the usability principles from Chapter 4 which will serve as attributes for separate usability specifications.

In the first example, we will consider the interaction principle of guessability, which concerns how easy it is for new users to perform tasks initially. The measuring concept will be how long it takes a new user, without any instruction on the new system, to enter their first appointment in the diary. A sample usability specification is given below.

Attribute:	Guessability
Measuring Concept:	Ease of first use of system without training
Measuring Method:	Time to create first entry in diary
Now Level:	30 seconds on paper-based system
Worst Case:	1 minute
Planned Level:	45 seconds
Best Case:	30 seconds (equivalent to now)

The values in this usability specification might seem a little surprising at first, since we are saying that the best case is only equivalent to the currently achievable level now. The point in this example is that the new system is replacing a very familiar paper and pencil system which requires very little training. The objective of this system is not so much to improve guessability but to preserve it. In the chapter, we discussed that the worst case level should not usually be worse than the now level, but we are hoping for this product to improve overall functionality of the system. The user will be able to do more things with the electronic diary than they could with the conventional system. As a result, we worry less about improving its guessability. Perhaps we could have been more ambitious in setting the best case value by considering the potential for voice input or other exotic input techniques which would make entry faster than writing.

As another example, this time within the flexibility category, we want to support the task migratability of the system. A frequent sort of task for a diary is to schedule weekly meetings. The conventional system would require the user to make an explicit entry for the meeting each week—the task of the scheduling is the responsibility of the user. In the new system, we want to allow the user to push the responsibility of scheduling over to the system, so that the user need only indicate the desire to have a meeting scheduled for a certain time each week and the system will take care of entering the meeting at all of the appropriate times. The task of scheduling has thus migrated over to the system. The usability specification for this example follows.

Attribute:	Task migratability
Measuring Concept:	Scheduling a weekly meeting
Measuring Method:	Time it takes to enter a weekly meeting appointment
Now Level:	(time to schedule one appointment) $\times$ (number of weeks)
Worst Case:	time to schedule two appointments
Planned Level:	$1.5 \times$ (time to schedule one appointment)
Best Case:	time to schedule one appointment

In this specification, we have indicated that the now level is equivalent to the time it takes to schedule each appointment separately. The worst, planned and best case levels are all targeted at some proportion of the time it takes to schedule just a single appointment—a dramatic improvement. The difference between the worst, planned and best case levels is the amount of overhead it will take to indicated that a single appointment is to be considered an example to repeated at the weekly level.

What are the assumptions we have to make in order to arrive at such a usability specification? One of the problems with usability specifications, as we have stated in the chapter, is that they sometimes require quite specific information about the design in order to be expressed. For example, had we set one of our measuring methods to count keystrokes or mouse clicks, we would have had to start making assumptions about the method of interaction that the system would allow. Had we tried to set a usability specification concerning the browsing of the diary, we would have had to start making assumptions about the layout of the calendar (monthly, weekly, daily) in order to make our estimates specific enough to measure. In the examples we have provided above, we have tried to stay as abstract as possible, so that the usability specifications could be of use as early in the design life cycle as possible. A consequence of this abstractness, particularly evident in the second example, is that we run the risk in the usability specification of setting goals that may be completely unrealistic, though well-intentioned. If the usability specification were to be used as a contract with the customer, such speculation could spell real trouble for the designer.

Exercise 5.2

> Can you think of any instances in which the 'noun–verb' guideline for operations, as suggested in the Apple human interface guidelines for the Desktop Interface, would be violated? Suggest other abstract guidelines or principles besides consistency which support your example. (**Hint:** Think about moving files around on the Desktop.)

Answer

The noun-verb guideline suggests that we can view all operations that the user will perform as being composed of an action (the verb) acting with one argument (the noun). In the case of moving a file (or copying for that matter), the action (move or copy) requires more than one argument. The way the move operation is performed requires the user to first select the icon for the file to be moved and then indicate the move operation implicitly by dragging the selected icon to the destination folder. The nouns in this dialogue are the file to be moved and the destination folder. The verb is the move operation. The natural way to express this is in the order noun-verb-noun. Strictly speaking, in order to stick with the noun-verb guideline, we would have to indicate both the target file and the destination folder before indicating the move operation. That would be consistent with relative to input expression with most other commands on the desktop. However, some principles of direct manipulation and the familiarity are more important. Moving files by dragging them on the desktop is very similar to the way we can pick up any object in the physical world and move it to its new location. And the dragging operation is incremental and easily recoverable; moving to one place can be undone within the same operation since the dragging can continue until the file is released.

The file moving example is a slightly contrived one, because some could argue that there is no violation of the noun-verb guideline (hence, moving is still consistent with respect to input expression) because the verb is "move to destination folder." Perhaps a better example is a command to search a file system for files matching some specification. Here, the action is to do the qualified search and the argument or noun is the set of folders or volumes of the system that you want searched. Typically, this kind of operation is defined by some dialogue box that allows the user to indicate in any order the specifics of the operation (the search parameters) and the folders or volumes to search. Once this unordered dialogue is complete, the user then indicates that it is OK for the system to perform the operation. This kind of form-filling dialogue prescribes to neither the noun-verb or verb-noun guideline; the order is more flexible for the user than consistent.

Exercise 5.3

Can you think of any instances in which the user control guideline suggested by Apple is not followed? (**Hint:** Think about the use of dialogue boxes.)

Answer

The user control guideline states that, "The user, not the computer, initiates and controls all actions." In the case of dialogue boxes, this guideline is clearly contradicted. A dialogue box can be used to indicate when an error occurs in the system. Once this error has been detected and presented to the user in the dialogue box, the only action that the system allows the user is to acknowledge the error and dismiss the dialogue box. The system preempts the user dialogue, with good reason. The preemptive nature of the dialogue box is to ensure that the user actually notices that there was an error. Presumably, the only error that will be produced in such an intrusive manner are ones which the user must know about before proceeding, so the preemption is warranted. But sometimes dialogue boxes are not used to indicate errors and they still prevent the user from performing some actions that they might otherwise wish to perform. The dialogue box might be asking the user to fill in some information to specify parameters for a command. If the user does not know what to provide, then they are stuck. A lot of the time, the user can find out the information by browsing through some other part of the system, but in order to do that they must exit the dialogue box (and forfeit any of the settings that they might have already entered), find out the missing information and begin again. This kind of preemption is not desirable. It is probably the kind of preemption this guideline is intending to prevent, but it doesn't always get applied.

Exercise 5.4

Find a book on guidelines. List the guidelines that are provided and classify them in terms of the activity in the software life cycle to which they would most likely apply.

Answer

We use as a source of guidelines Mayhew's book *Principles and Guidelines in Software User Interface Design*. In general, all guidelines offer constraints on the design activity and so should be known during requirements phase. In the following list, we will concentrate on what other stages (architectural design, detailed design, coding and unit testing, integration and testing) will be most affected by the guideline. The numbers in parentheses indicate the page reference for the given guideline.

- Architectural design
  - Present functionality through a familiar metaphor. (97)
  - Provide similar execution style of analogous operations in different applications. (97)
  - Organize the functionality of a system to support common user tasks. (442)
  - Make invisible parts and processes visible to the user. (95)
- Detailed design
  - Consistent dialogue style for different functions. (97)
  - Match menu structure to task structure. (144)
  - Create logical, distinctive and mutually exclusive semantic categories with clear meanings. (150)
  - Design and organize a fill-in form to support the task. (184)
  - Consider voice synthesis as an output device when the user's eyes are busy, when mobility is required, or when the user has no access to a workstation or screen. (427)
- Coding and unit testing
  - On full-screen text menus, present menu choice lists vertically. (148)
  - In a fill-in form, use white space to create a balance and symmetry and lead the eye in the appropriate direction. (186)
  - Avoid frequent use of shift or control keys. (256)
  - Place high-use function keys within easy reach of the home row on the keyboard. (281)
- Integration and testing
  - Allow full command names and emphasize them in training, even if abbreviations are allowed. (261)

Exercise 5.5

What is the distinction between a process-oriented and a structure-oriented design rationale technique? Would you classify psychological design rationale as process- or structure-oriented? Why?

Answer

The distinction between process- and structure-oriented design rationale resides in what information the design rationale attempts to capture. Process-oriented design rationale is interested in recording an historically accurate description of a design team making some decision on a particular issue for the design. In this sense, process-oriented design rationale becomes an activity concurrent with the rest of the design process. Structure-oriented design rationale is less interested in preserving the historical evolution of the design. Rather, it is more interested in providing the conclusions of the design activity, so it can be done in a *post hoc* and reflective manner after the fact.

The purpose of psychological design rationale is to support the task-artifact cycle. Here, the tasks that the user community performs are changed by the systems on which they perform the tasks. A psychological design rational proceeds by having the designers of the system record what they believe the tasks are that the system should support and then building the system to support the tasks. The designers suggest scenarios for the tasks which will be used to observe new users of the system. Observations of the users provide the information needed for the actual design rationale of that version of the system. The consequences of the design's assumptions about the important tasks are then gauged against the actual use in an attempt to justify the design or suggest improvements.

Psychological design rationale is mainly a process-oriented approach. The activity of a claims analysis is precisely about capturing what the designers assumed about the system at one point in time and how those assumptions compared to actual use. Therefore, the history of the psychological design rationale is important. The discipline involved in performing a psychological design rationale requires designers to perform the claims analysis during the actual design activity, and not as *post hoc* reconstruction.

Chapter 6

Models of the User in Design

Exercise 6.1

Create a GOMS description of the task of photocopying a paper from a journal. Discuss the issue of closure in terms of your GOMS description.

Answer

One possible GOMS description of the goal hierarchy for this task is given below. Answers will vary depending on assumptions about the photocopier used as the model for the exercise. In this example, we will assume that the article is to be copied one page at a time and that a cover over the imaging surface of the copier has to be in place before the actual copy can be made.

```
GOAL: PHOTOCOPY-PAPER
.     GOAL: LOCATE-ARTICLE
.     GOAL: PHOTOCOPY-PAGE repeat until no more pages
.     .     GOAL: ORIENT-PAGE
.     .     .       OPEN-COVER
.     .     .       SELECT-PAGE
.     .     .       POSITION-PAGE
.     .     .       CLOSE-COVER
.     .     GOAL: VERIFY-COPY
.     .     .       LOCATE-OUT-TRAY
.     .     .       EXAMINE-COPY
.     GOAL: COLLECT-COPY
.     .     LOCATE-OUT-TRAY
.     .     REMOVE-COPY  (outer goal satisfied!)
.     GOAL: RETRIEVE-JOURNAL
.     .     OPEN-COVER
.     .     REMOVE-JOURNAL
.     .     CLOSE-COVER
```

The closure problem which appears in this example occurs when the copy of the article is removed from the photocopier out tray, satisfying the overall goal for the task. In the above description, however, the original journal article is still on the imaging surface of the photocopier, and the cover is closed. The user could easily forget to remove the journal. How could the photocopying procedure be revised to eliminate this problem? One answer is to force the goal RETRIEVE-JOURNAL to be satisfied before COLLECT-COPY.

Exercise 6.2

Recall the CCT description of the rule INSERT-SPACE-2 discussed in Section 6.3.2:

```
(INSERT-SPACE-2
IF (AND (TEST-GOAL insert space)
        (TEST-CURSOR %LINE %COL) )
THEN (  (DO-KEYSTROKE 'I')
        (DO-KEYSTROKE SPACE)
        (DO-KEYSTROKE ESC)
        (DELETE-GOAL insert space) ))
```

As we discussed, this is already proceduralized, that is, the rule is an expert rule. Write new 'novice' rules where the three keystrokes are not proceduralized. That is, you should have separate rules for each keystroke and suitable goals (such as GET-INTO-INSERT-MODE) to fire them.

Answer

```
(INSERT-SPACE-BEGIN-SET-MODE
IF (AND (TEST-GOAL insert space)
        (TEST-CURSOR %LINE %COL)
        (TEST-NOTE in command mode ))
THEN (  (ADD-GOAL get into insert mode)))

(INSERT-SPACE-END-SET-MODE
IF (AND (TEST-GOAL insert space)
        (TEST-GOAL get into insert mode))
THEN (  (DO-KEYSTROKE 'I')
        (ADD-NOTE in insert mode)
        (DELETE-GOAL get into insert mode)))

(INSERT-SPACE-DOIT
IF (AND (TEST-GOAL insert space)
```

```
        (TEST-NOTE in insert mode)
        (TEST-CURSOR %LINE %COL))
THEN (  (DO-KEYSTROKE SPACE)
        (ADD-GOAL get into command mode)))

(INSERT-SPACE-CLEAN-UP
IF (AND (TEST-GOAL insert space)
        (TEST-NOTE in insert mode)
        (TEST-GOAL get into command mode))
THEN (  (DO-KEYSTROKE ESC)
        (DELETE-GOAL get into command mode)
        (DELETE-GOAL insert space)
        (DELETE-NOTE in insert mode)
        (ADD-NOTE in command mode)))
```

Exercise 6.3

Do a keystroke level analysis for opening up an application in a visual desktop interface using a mouse as the pointing device, comparing at least two different methods for performing the task. Repeat the exercise using a trackball. Discuss how the analysis would differ for various positions of the trackball relative to the keyboard and for other pointing devices.

Answer

We provide a keystroke level analysis for three different methods for launching an application on a visual desktop. These methods are analyzed for a conventional one-button mouse, a trackball mounted away from the keyboard and one mounted close to the keyboard. The main distinction between the two trackballs is that the second one does not require an explicit repositioning of the hands, that is, there is no time required for homing the hands between the pointing device and the keyboard.

Method 1: Double clicking on application icon.

Steps		Operator	Mouse	Trackball_1	Trackball_2
1.	move hand to mouse	H[mouse]	0.400	0.400	0.000
2.	mouse to icon	P[to icon]	0.664	1.113	1.113
3.	double click	2B[click]	0.400	0.400	0.400
4.	return to keyboard	H[kbd]	0.400	0.400	0.000
Total times			1.864	2.313	1.513

Method 2: Using an accelerator key

	Steps	Operator	Mouse	Trackball_1	Trackball_2
1.	move hand to mouse	H[mouse]	0.400	0.400	0.000
2.	mouse to icon	P[to icon]	0.664	1.113	1.113
3.	click to select	B[click]	0.200	0.200	0.200
4.	pause	M	1.350	1.350	1.350
5.	return to keyboard	H[kbd]	0.400	0.400	0.000
6.	press accelerator	K	0.200	0.200	0.200
Total times			3.214	3.663	2.763

Method 3: Using a menu

	Steps	Operator	Mouse	Trackball_1	Trackball_2
1.	move hand to mouse	H[mouse]	0.400	0.400	0.000
2.	mouse to icon	P[to icon]	0.664	1.113	1.113
3.	click to select	B[click]	0.200	0.200	0.200
4.	pause	M	1.350	1.350	1.350
5.	mouse to File menu	P	0.664	1.113	1.113
6.	pop-up menu	B[down]	0.100	0.100	0.100
7.	drag to open	P_{drag}	0.713	1.248	1.248
8.	release mouse	B[up]	0.100	0.100	0.100
9.	return to keyboard	H[kbd]	0.400	0.400	0.000
Total times			4.591	6.024	5.224

Exercise 6.4

One of the assumptions underlying the programmable user model approach is that it is possible to provide an algorithm to describe the user's behaviour in interacting with a system. Taking this position to the extreme, choose some common task with a familiar interactive system (e.g. creating a column of numbers in a spreadsheet and calculating their sum, or any other task you can think of) and describe the algorithm needed by the user to accomplish this task. Write the description in pseudocode. Does this exercise suggest any improvements in the system?

Answer

This is a pretty open-ended exercise, so no model answer is provided.

Chapter 7

Task analysis

The exercises for this chapter, like the chapter itself, concentrate on real world rather than computer examples. This is largely because task analysis of current computer systems depends on the particular systems available. However, the form of a general student exercise could be:

> Observe use of the …computer system. Perform a task analysis (of some kind). To what extent do you think the way people perform these tasks is determined by the system and to what extent by the fundamental aspects of the task? Does the system's menu layout etc. support the tasks it is used for, in particular, are frequent task sequences easy to perform? Suggest potential improvements, both incremental changes and radical redesigns of the system.

The details of the question can be of course varied depending on the system and the students can be directed to particular sub-systems where you have observed problems.

Exercise 7.1

The following is a list of objects found in one of the authors' kitchens.

> teapot, mug, soup bowl, plate, spoon, table knife, cook's knife, fork, saucepan, frying pan, kettle, casserole, fish slice, tin opener, baking tray, scales, miking bowl, glasses, jugs, corkscrew, rolling pin, ladle, egg cup, chopping board

Produce a taxonomy using the TDH notation of these objects. Does it obey the TAKD uniqueness rule? Compare your answer with someone else's. (Note, the authors had great difficulty with items like the corkscrew, which did not fit easily into any generic category — perhaps you did better.)

```
kitchen object XOR
|__ preparation XOR
|   |__ pre-preparation XOR
|   |   |__ opening
|   |   |       tin opener, cork screw
|   |   |__ measuring
|   |           scales, (measuring)jug
|   |__ 'proper' preparation XOR
|       |__ active
|       |       rolling pin, cook's knife, (cook's) spoon
|       |__ passive
|               mixing bowl, chopping board
|__ cooking XOR
|   |__ passive
|   |       teapot
|   |__ active XOR
|       |__ external power
|       |       saucepan, frying pan, casserole, baking tray
|       |__ internal power
|               (electric) kettle
|__ serving XOR
    |__ serving
    |       fish slice, (serving) jug, ladle
    |__ eating XOR
        |__ active
        |       spoon, fork, knife
        |__ passive XOR
            |__ food
            |       egg cup, soup bowl, plate
            |__ drink
                    mug, glass
```

Figure 7.x1 *TDH taxonomy produced by first subject*

Answer

As the authors had already produced a partial taxonomy, we interviewed two domain experts (cooks). They were asked to describe how they would group and classify the kitchen items. they were explicitly told (and reminded) that they could have multiple classifications and put the same item into several categories. The authors then cast their answers into TDH notation.

One of the subjects was a doctor and used to medical taxonomies of disease. Despite stressing the looseness of the classifications, he insisted on a complete taxonomic tree (Figure 7.x1). As you see all his branches are **XOR** branches. On discovering that 'jug' had to fit in two places in his taxonomy, he split it into

kitchen object **OR**
{___ things for making tea
{ *teapot, mug, kettle, spoon*
{___ things for eating meals
{ *soup bowl, plate, glasses, egg cup*
{___ cutlery for meals
{ *spoon, table knife, fork*
{___ cutlery for cooking
{ *spoon, fork, fish slice, tin opener, table knife,*
{ *corkscrew, rolling pin, ladle*
{___ things for making meals
{ *saucepan, frying pan, casserole, baking tray, scales,*
{ *mixing bowl, jugs, chopping board*
{___ things for serving meals
jugs, casserole, fish slice, corkscrew, ladle, spoon

Figure 7.x2 *Initial version of TDH taxonomy produced by second subject*

'serving jug' and 'measuring jug'. This emphasises the need for the task analyst rather than the domain expert to actually draw up the taxonomy!

As it is a true tree it clearly does not satisfy the uniqueness rule, but the only way it could is to invent spurious new categories. One could under 'opening' add categories for 'bottles' (containing corkscrew) and tins (continuing tin opener), but this would not improve clarity.

If the first subject was a stickler for precision the second subject, preferred broad categories. Figure 7.x2 shows her initial classification. We wanted to put some additional structure on this and so, after some discussion, the subject agreed that her basic distinctions were those of function ('making meals' etc.) and between cutlery and non-cutlery. Using these to form an **AND** branch, we obtained Figure 7.x3.

This taxonomy does not obey the uniqueness rule either, for example, fish slice and ladle always appear together. In terms of KRG they are both:

kitchen object/material(cutlery)/
function{cooking meals,serving meals}/

The TAKD purist might demand extra categories to fulfill the uniqueness rule. However, the authors would recommend that students simply be taught to recognise the rule and use it as a heuristic.

It is interesting that both domain experts focussed on the functional view of the items, just as the authors did in the book. This suggests that it is indeed a generic way of classifying kitchen objects and would thus be a good candidate for classification in a catalogue or menu system. The second subject also noted that her original breakdown was inspired, not so much by the function *per se*, but by

```
kitchen object AND
/__ material XOR
/     |__ cutlery
/     |         spoon, table knife, fork, fish slice, tin opener
/     |         corkscrew, rolling pin, ladle
/     |__ non-cutlery
/               teapot, mug, kettle,
/               soup bowl, plate, glasses, egg cup,
/               saucepan, frying pan, casserole, baking tray, scales,
/               mixing bowl, jugs, chopping board
/__ function OR
      {__ making tea
      {         teapot, mug, kettle, spoon
      {__ cooking meals
      {         spoon, fork, fish slice, tin opener, table knife,
      {         corkscrew, rolling pin, ladle,
      {         saucepan, frying pan, casserole, baking tray, scales,
      {         mixing bowl, jugs, chopping board
      {__ serving meals
      {         jugs, casserole, fish slice, corkscrew, ladle, spoon
      {__ eating meals
                soup bowl, plate, glasses, egg cup,
                spoon, table knife, fork
```

Figure 7.x3 *Refined version of TDH taxonomy produced by second subject*

where they were stored in her kitchen – itself determined largely by function. This is perhaps the physical equivalent of a menu system!

Exercise 7.2

Complete the tea-making manual in Figure 7.7. Do you think it would be useful? Think of situations where such a manual would be helpful and where a more conceptual manual would be better.

Answer

Although a manual for the tea making might be regarded as a little extreme, such manuals are useful in several situations. You could pose this exercise, together with the initial task analysis, for different domains where more of the following situations are pertinent.

The first situation where a procedural manual is useful is for the absolute novice who has no idea of the conceptual background. This might be a first time user or may be for an infrequently performed activity. A good example of the latter is the installation of computer equipment, which most users perform only once every couple of years. Similarly, recipe books are laid out in a highly procedural fashion, although unfortunately not always clearly:

> Beat the egg whites until they froth, then put them into a ramekin. While beating the egg whites, slowly add the white wine.

The second situation is where there is some sort of safety critical aspect and errors, however well thought out, can be disastrous. Often in such a situation, the additional stress can cloud judgement and make it far safer to stick to a predetermined drill. One example, of this are emergency procedures in large chemical or nuclear installations — when an emergency arises the operators are expected to stick closely to the set procedures. The accident at Chernobyl came about in part because the operators felt that they knew enough to override the rule book. Reading a manual in such circumstances may be too time consuming, but an HTA can be used to train the operator to respond automatically. The use of HTA for military training is largely in this vein.

Thirdly, the situation may not be safety critical, but may be time critical. Much analysis may have gone into discovering the most efficient manner to perform a task and that way is then taught, by rote, to the operators. Although this form of time and motion approach is less likely to be useful in an information intensive job than in a factory (if there!), there are jobs, such as telephony, where it is still important.

Finally, the user may not have sufficient knowledge to understand why a process works, but can follow a set of instructions. This may either relate to the complexity of the task or the skill of the operator. If one were teaching kitchen craft to the mentally handicapped, then just such a procedural description of tea making would be required.

The problem with such procedural manuals is that they give the operator no real feeling as to why the tasks are performed in the way they are. Whether such a manual is preferred by a novice user depends very much on the user's personality. Some people prefer to have a set of instructions to get them started, whereas others find it very difficult to use something without some sort of conceptual understanding.

The procedural manual really becomes unstuck when the set of tasks considered are not complete. When faced with a radically new task the user must understand enough of the domain to perform it *ad hoc* or to modify an existing procedure. One frequent cause of entirely new situations is unforeseen breakdowns of equipment. For example, if the kettle was broken, one could then abstract that the real reason for boiling the kettle was to heat water and that this could be performed by heating a bowl of water in the microwave oven. Such a modification of the procedure is not even suggested by the procedural manual.

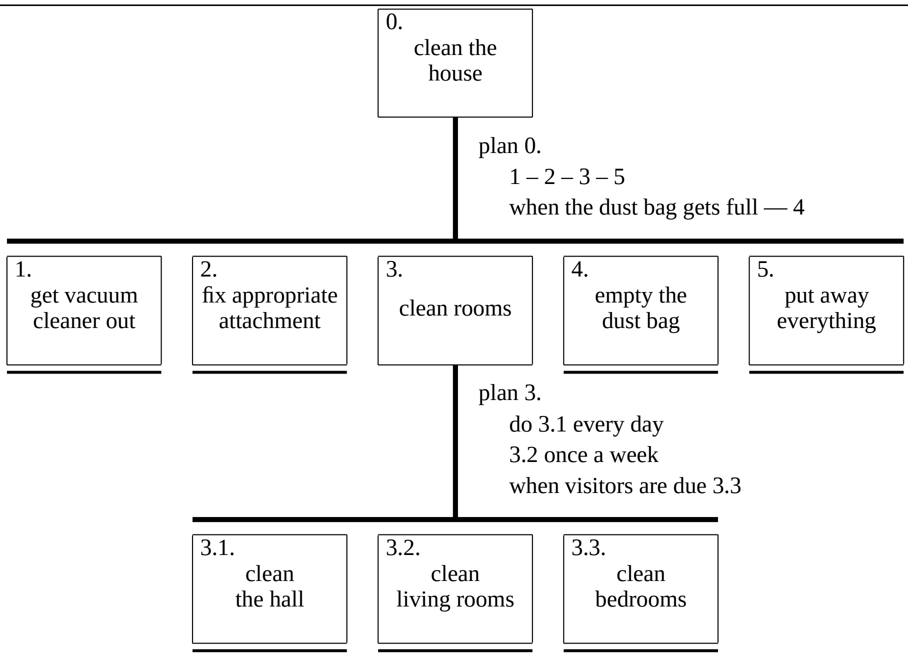

Figure 7.x4 *HTA diagram for vacuum cleaning a house*

Exercise 7.3

Figure 7.1 shows a textual representation of an HTA description of vacuum cleaning. Present the same information in a diagrammatic form.

Answer

This is given in Figure 7.x4. In fact, this uses the revised version of 'plan 3', but the choice of plan makes no difference to the structure of the diagram.

Exercise 7.4

(Converse to above) Figure 7.6 uses an HTA diagram to show the actions in which a tractor is involved; show the same information textually.

Answer

0. life cycle of tractor
    1. maintenance
    2. cultivation
        2.1. connect implement
            2.1.1 fix harrow
            2.1.2 fix plough
        2.2. drive to field
        2.3. cultivate field
        2.4. drive to shed
        2.5. put away

Plan 0: as required — 2
when tractor breaks down — 1

Plan 2: 2.1 – 2.2 – 2.3 –2.4 –2.5

Plan 2.1: one of 2.1.1 or 2.1.2 depending on job

Exercise 7.5

> Observe an office, note the actions performed and the objects used depending on the available equipment, use different recording techniques as described in Chapter 11. Then use the different task analysis techniques to structure your findings. (Note, this could be a group project.)

Answer

The easiest starting point for the students is simply to go around the office noting down what is there:

> typewriter, corrector fluid, desk diary, pen, pencil, scissors, envelopes, paper clips, typing paper, post-it notes, telephone, telephone directory (internal and external), filing cabinet containing folders, clock, wall calendar.

This list can then be used to begin to build both a knowledge based or an entity-relation description. However, the later will also require at least a list of actors. In a university office this might include the following:

> secretary, lecturer, student (undergraduate), research student, research staff, administrator

However, the rôles that they take may not be simple. For example, we may find that a lecturer comes into the office to use the typewriter. That is the lecturer acts in the rôle of *typist*.

Neither of these descriptions can be complete, nor can an HTA begin, without a list of activities. This can be obtained in two main ways. First, the students can simply make an unstructured list of all the activities they see and then add structure to it. Alternatively, they can follow specific tasks noting what is done in what order. In the later case, they should be encouraged to write the list of activities in a purely sequential manner – they are observing. Only later will they build upon this an hierarchical interpretation.

It may obviously cause severe inconvenience if all the members of a class were to interview the office staff. However, to gain first hand interview experience, some domain expert can be invited into class to talk about their work and be questioned about it. Alternatively, students could make their own notes from a preprepared video taped interview.

If a question of this sort is used as an assessment, then we would suggest that the students hand in not just the completed task analysis, but intermediate notes and representations. The most important thing in determining the effectiveness of their analysis is the care with which they carried out the original observation and subsequent working.

Exercise 7.6

Consider the activity of making a telephone call. Record the actions in an HTA diagram or textually. Start off simply, assuming you know the number to dial, but then add more complicated situations: finding the number in an address book, or what to do when the number is engaged.

Answer

As with most of these exercises, this is an open ended question. Here is a simple version with some expansion, but one can look at alternatives such as public payphones.

0. make phone call
    1. pick up receiver
    2. dial number
    3. wait for reply
    4. talk
    5. replace receiver

Plan 0: 1 – 2 – 3
when answered — 4
when finished — 5

We now add looking up the number. The form this takes depends on whether we find the number in an address book or a telephone directory. If both fail, say if the call is long distance to someone not in a local directory, the telephone operator must be consulted. If students have some form of online telephone list, possibly in a palmtop computer, they could included the analysis of the lookup procedure. The next exercise, dictionary lookup, is very similar to looking up a number in a directory, and so further analysis of these branches is not included here. Note that contacting the telephone operator involves making a phone call, but the steps for this are not repeated in full!

0. make phone call
    1. find number
        1.1. look in address book
        1.2. look in phone directory
        1.3. ask operator
            1.3.1 pickup receiver
            …
    2. actually call
        2.1. pick up receiver
        2.2. dial number
        2.3. wait for reply
        2.4. talk
        2.5. replace receiver

Plan 0: if number unknown — 1
when number found — 2

Plan 1: if phoning friend — 1.1
if local call — 1.2
if 1.1 or 1.2 fail — 1.3

Plan 2: 2.1 – 2.2 – 2.3
when answered — 2.4
when finished — 2.5

Finally, we add the case when the phone is engaged. The simplest way to do this is simply to change Plan 2.

Plan 2: 2.1 – 2.2 – 2.3
if answered — 2.4 then when finished 2.5
if engaged — 2.5

However, looking at the second line it might suggest that we modify 2.4 to have two parts:

0. make phone call
   …
   2. actually call
      …
      2.4. successful call
         2.5.1. talk
         2.5.2. replace receiver
      2.5. failed call
         2.5.1. replace receiver

Plan 2: 2.1 – 2.2 – 2.3
if answered — 2.4
if engaged — 2.5

Plan 2.4: 2.4 then when finished 2.5

Exercise 7.7

The act of looking up a person in an address book or telephone directory is itself quite complicated. Get several friends to look up words in a dictionary. Observe closely their methods. You will probably have to develop shorthand notations to keep track of what pages they visit. Compare the strategies used by the different people. If they differ, try to abstract out the common parts of the task and the variable parts. Those with sufficient computing background can attempt to classify their methods in relation to known search algorithms: binary chop, linear search, etc.

Answer

In order to see the range of possible search strategies, the authors asked two subjects to look up words in a dictionary. One was a literate, but young, child and the other an erudite English graduate.

To obtain any sort of useful record, the subjects had to be asked to work slowly and even then keeping track of the search was difficult. We noted down the page numbers they paused at, but often searches included thumbing through a sequence of pages, at which point some form of specific notation would be useful. As an alternative to noting down the page numbers, noting the words at the top of the page could be used.

????	*note illegible, intended to convey that she turned the dictionary on its side and then opened it*
283 dog	*said that first cut was based on the position of 'd' in the alphabet*
flick backward	
overshoot to 'C'	
forward one page to beginning of 'D'	*'daltonism' on this page*
quick skim over page	
places finger on word	

Table 7.x1 *Adult looking up the word 'daltonism'*

right of centre	*comment by subject on position of 'p' in dictionary*
729 O	*gets to page in the middle of 'O' section*
turn over forward	
741	*page begins with 'para' words*
? C or K	*subject says unsure whether spelt with c or k says she'll look for 'c' first*
scan with finger	
sequential	*she explains that the presence of the 'para' words made the task confusing*

Table 7.x2 *Adult looking up the word 'parclose'*

The English graduate was also asked to talk aloud through the exercise. We discuss in Chapter 11 the possible effects this might have on the user's performance, but it is extremely difficult to keep track of the readers focus on the page otherwise. The notes below are tidied up considerably from our hand written scrawl! The notes in italics were added after. Unfortunately, in typesetting the notes, some of the spatial layout is lost. They do, however, retain the scrappiness of the original observations. Use of video can of course allow one to replay tasks at a slower rate and thus improve the detail of note taking.

The English graduate was asked to find unfamiliar words: 'daltonism' (Figure 7.x1) and 'parclose' (Figure 7.x2). The child was asked to lookup simpler words: 'dig' (Figure 7.x3) and 'orange' (Figure 7.x4). The transcripts do not include timings — the whole process took no more than 30 seconds. To give an idea of relative timings: whereas the English graduate homed quickly onto the correct page, this took the child somewhat longer, but, once they were on the right page, both were so quick that it is hard to give judgement on relative speed.

From this evidence, about the best general description we could have is:

525	*near middle of book*
429	*these first pages were quite rapid*
297	*page headed 'dualism'*
266	*correct page – luck!*
scan down and up	*sequential search item by item*

Table 7.x3 *Child looking up the word 'dig'*

236	*last page of 'c'*
266	
flick forward	*too fast to note pages*
800	*overshoot*
flick back	
700, 690, 699	*start of 'O'*
sequential pages	
703	
flick forward page by page	
717	*right page*
sequential search through page	

Table 7.x4 *Child looking up the word 'orange'*

0. look up word in dictionary
    1. find right page
    2. find word on page

Plan 0: 1 – 2

The performance of subtask 1 was far more efficient for the adult as she used a heuristic to find the approximates position based on her knowledge of the alphabet. The child knew the relative position of words in the alphabet, but not enough to be able to say 'p' is just after the middle as the adult did. The differences in subtask 2 was less marked, but mainly because it was too rapid to annotate. There was some evidence that the adult used a more direct method than the child who simply (but quickly) scanned the page entry by entry.

If individual students observe more subjects, or as they come together to make notes, they may be able to abstract more specific, but still general search strategies. For example, they may see something akin to binary chop. They should not look for a single strategy used by all people, but instead look for a small set of strategies, and then look for ways of telling how people choose one strategy or another.

Really, two observations, as described above, is only sufficient for a pilot study in order to sharpen up thinking. Looking at the notes may suggest shorthands, for instance, arrows to represent different kinds of scanning. It also prompts one to ask questions. What happens if when they scan the page at step 2, the subjects discover that they are still on the wrong page? Presumably this leads to a new 'find the right page' step. However, this subtask, essentially a fine correction, will be very different from the original search.

0. look up word in dictionary
    1. find right page
    2. find word on page
    3. adjust page

Plan 0: 1 – 2
if still wrong page 3 – 2

However, unlike the first HTA, this one is not based on *observation*, but on the authors' *imagination*. Given such refined expectations, the students ought to return to observation in order to explore further. Perhaps, they could deliberately choose words close to page boundaries in order to confuse the subjects. Alternatively, they could ask their subjects 'what would you do if you'd got the wrong page?'. You should emphasise to the students the difference between a psychological experiment in problem solving and a task analysis observation. For the former, we would be simply trying to verify our guess/theory. For the latter, we might have an idea, but our observations should be open ended as we want to find out what really happens, whether or not it agrees with our theories.

Although the question is clearly oriented towards procedural analysis, One could also do some form of knowledge based analysis. Clearly concepts such as 'position in the alphabet' and 'word at the top of the page' are important. Indeed, if we were designing an electronic dictionary such concepts may be more useful than emulating paper.

Indeed, one could extend this question into a design exercise – how to design the electronic dictionary. Assuming that we are not interested in totally automatic lookup, how would we design the interaction? Whereas in a book, one's normal action is to turn over one page backwards or forwards, in dictionary lookup one jumps into the middle, turns over several pages at a time and flicks through pages at high speed. This can emphasise that a question like 'how do we design an electronic book?' cannot be answered until we know for what purposes that book is to be used.

Chapter 8

Dialogue Notations and Design

Exercise 8.1

Complete the drawing tool STN in Figures 8.1 and 8.3 by writing dialogue descriptions for the text and paint sub-menus. For the text sub-menu assume that there are three options: centred, left and right justified. The text is entered by clicking at a location in the drawing surface and then typing. You may initially assume that typing a line of text can be regarded as a single user action. But later try regarding each character typed as an action. The paint sub-menu has two options: a pencil for free hand drawing and a paint pot for flood filling. The former is performed by holding the mouse button down whilst moving the mouse about to draw the line. The paint pot is activated by simply clicking the mouse over the area to be filled.

Answer

The STNs for the main menu and graphics subsystem are in the text. We only need to do the text-submenu and the paint-submenu. For each the overall structure is similar to the graphics submenu, that is, the user selects an option, the STN 'branches' and then there is a description of each option's dialogue.

First do the text sub-menu considering typing a line of text as a single user action. This is shown in Figure 8.x1. To consider individual letters being typed, we simply add a loop terminated by the user typing the enter key (Figure 8.x2). Finally, we do the paint sub-menu in Figure 8.x3.

Note that the fundamental actions for the pencil drawing are pressing and releasing the mouse button, rather than clicking it. Remember the more complex behaviour of the watch when we considered button press and release separately. It is worth discussing the expected behaviour of the package on other options during mouse depression. For example, when drawing lines, which position is registered the one when the button goes down, or the one when it is released. Encourage the students to try it out on different drawing packages. There is a similar issue

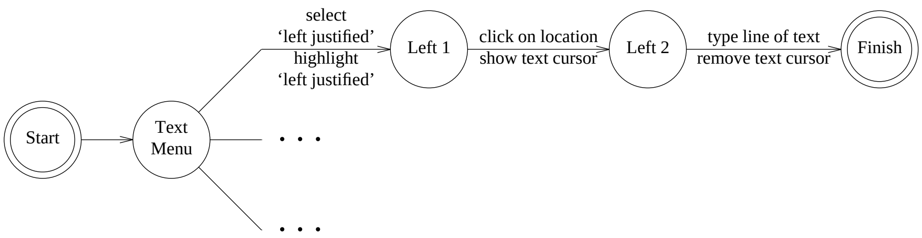

Figure 8.x1 *STN of text submenu*

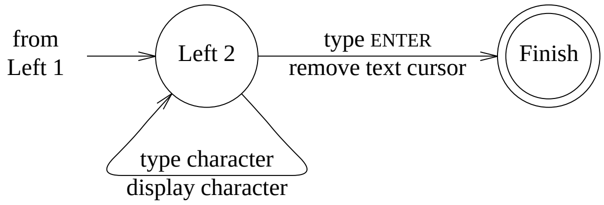

Figure 8.x2 *Character by character input*

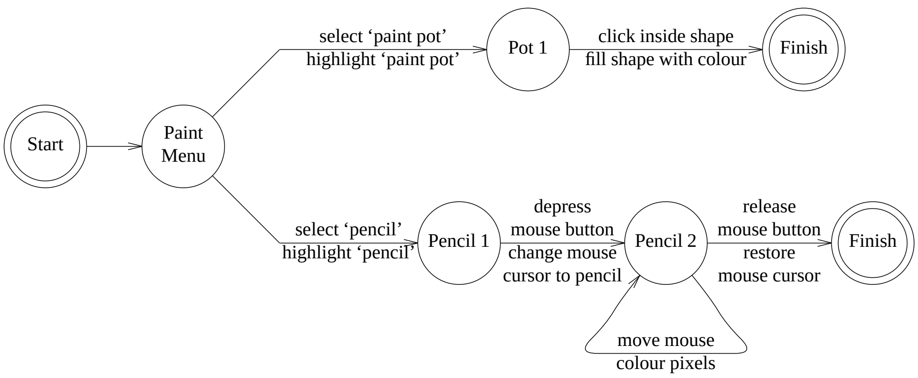

Figure 8.x3 *Paint submenu*

with menu selection and screen buttons. The scenario described in Section 9.4.5 is an example where the important location is where the mouse button is released. Again get the students to look at different GUIs and their detailed interaction properties. You may be surprised at the differences!

Exercise 8.2

> Repeat the above exercise using different notations, grammars, production rules, JSD or CSP. You will need to specify the whole system from the main menu to the individual sub-menu selections such as circle drawing. Note the problems you have with each notation.

Answer

Below are partial descriptions in BNF and production rules. As a class exercise, you could allocate different notations to different groups of students and then compare the answers in different notations.

BNF description

The BNF description is quite straightforward (if a little tedious). We give the description for the main menu, part of the graphics submenu and the text submenu. The line drawing is already described in section 6.4.1 as is the definition of `position-mouse`.

```
drawing-tool  ::=  main-menu + drawing-tool
main-menu     ::=  graphics-submenu
                 | text-submenu
                 | paint-submenu

graphics-submenu  ::= draw-circle | draw-line
draw-circle
        ::=  select-circle + choose-centre + choose-edge
select-circle ::= position-mouse + CLICK-MOUSE
choose-centre ::= position-mouse + CLICK-MOUSE
choose-edge   ::= position-mouse + CLICK-MOUSE

text-submenu
        ::=  select-left   + choose-location + type-text
           | select-centre + choose-location + type-text
           | select-right  + choose-location + type-text
select-left     ::=  position-mouse + CLICK-MOUSE
select-centre   ::=  position-mouse + CLICK-MOUSE
select-right    ::=  position-mouse + CLICK-MOUSE
```

```
choose-location ::=  position-mouse + CLICK-MOUSE
type-text       ::=  ENTER-KEY
                   | PRINTABLE-CHAR + type-text
```

Notice the different styles possible. In the graphics submenu, the option selection and the action selections are each packaged up into a named sub-dialogue (`draw-line` and `draw-circle`). However, in the text-submenu the top-level description of each option is included within the definition of `text-submenu`. Emphasise to the students that, with all notations, there is a wide degree of stylistic choice. Suitably named sub-dialogues can make the entire dialogue far easier to read, but, on the other hand, too many levels of abstraction can be confusing.

Also notice that the BNF description does not distinguish between selecting a menu option or choosing a point on the drawing surface. Both are described as `position-mouse` followed by `CLICK-MOUSE`. At the level of user movements and keystrokes, this is perfectly accurate, but loses the clearly perceived difference. This might be a good time to introduce your students to the UAN notation (Hartson *et al.* 1990)

Production rules

As the drawing tool is largely sequential, we would expect to find the production rule description rather cumbersome. The main problem is keeping track of how deep you are in the stack of menus submenus and options. Watch out for solutions where the higher level options are still active, when the lower level dialogues are operating. For example, the top level description might read:

```
Sel-graphics  →  <pop-up graphics menu>
Sel-text      →  <pop-up text menu>
Sel-paint     →  <pop-up paint menu>
```

This could exhibit strange behaviour. Imagine that the user is in the middle of drawing a line (using the first description in section 8.4.2). Then after selecting the first point (at which point the system event '`rest-line`' is active), the user goes back to the top level menu and selects 'text'. At this point the text pup-up menu appears, and the user is able to start putting text annotations on the drawing area:

```
Sel-left-just        →  left-start
C-point left-start   →  left-type <typing cursor on>
Type-line left-type  →  <put text left justified>
                                <typing cursor off>
```

The user selects left justified and goes to the drawing surface and clicks at a point. However, at this point there are two rules which can fire.

- `C-point left-start` – from the text rules (above), setting the location for typing.

- `C-point rest-line` – from the drawing rules in Section 8.4.2,putting another point on the line.

Whereas it is certainly the first of these that the user expects, the second is possible because the '`rest-line`' event is still active. Also looking back to the definitions in section 8.4.2, we see that rubber banding would still be on during the entire text menu selection process.

There are various fixes for this that you can discuss with your students. One way is to use 'clean up' rules as used in CCT in section 6.3.2. For example, we would modify the top level description so that it is not always active:

```
Sel-graphics main-active
             → graphics-active <pop-up graphics menu>
Sel-text main-active
             → text-active <pop-up text menu>
Sel-paint main-active
             → paint-active <pop-up paint menu>
```

The submenu options are then altered to update these activity events (the semantic actions have been omitted for clarity):

```
Sel-line graphics-active  → start-line
C-point start-line        → rest-line
C-point rest-line         → rest-line
D-point rest-line         → main-active
```

This now ensures that the user finished low-level dialogues before selecting new options at the top level.

Exercise 8.3

Develop the JSD diagram in Figure 8.12, expanding the various nodes until you get to basic operations such as 'prompt `"login"`' or 'user types in password'. Expand the 'delete employee' node using the dialogue style as described in Figure 8.11, and use your imagination for the rest.

Answer

This is a fairly simple exercise, and some of the nodes are expanded here. In Figure 8.x4, the login sub-dialogue is expanded, but it assumes that the user types the correct password during the login process. To change it to accommodate incorrect passwords would be inordinately complex for two reasons. Firstly, the JSD diagram does not distinguish between user choice (whether to add or delete a record) and system choice. This is evident in both the delete and change sub-dialogues (Figures 8.x6 and 8.x5). In each of these the actual updating of the file

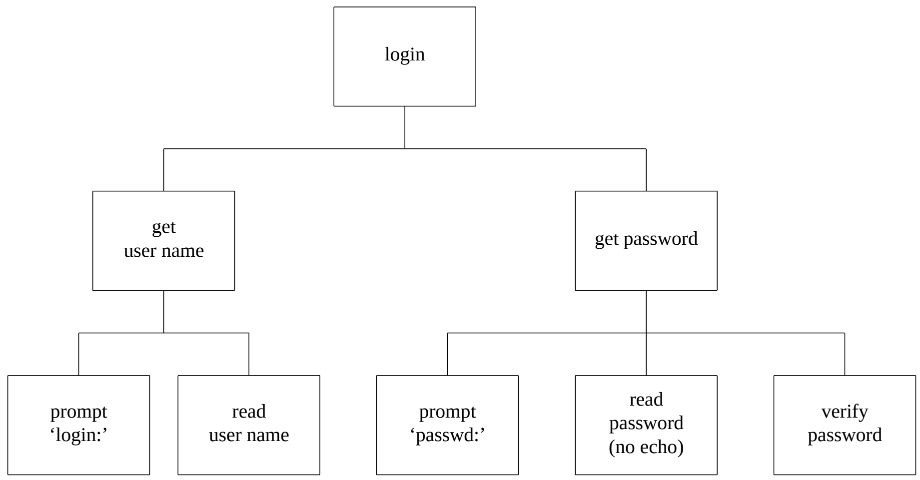

Figure 8.x4 *JSD diagram for logging in to personnel system*

is put in an 'optional' box as the user may have answered 'N' (no) when asked to confirm the update. Secondly, making the dialogue dependent on the password would mean that the rest of the dialogue would sit under an 'option' box with another box to abort. This was acceptable in the delete and change sub-dialogues, but if this were applied to the login sequence, the resulting dialogue would become degenerate and hide the normal hierarchical structure. This problem of whether or not to include exceptional cases and error behaviour is extremely complex. In early stages of design it is the normal cases which one wants to consider. Later on one might refine the dialogue descriptions, or alternatively annotate the normal description.

Exercise 8.4

Using CSP, construct a dialogue for one user with an application in one window of a multi-window system. Using this first general CSP description, provide a CSP description for a multi-window interaction using the parallel operator ($||$). How does this approach handle interference and information sharing between windows?

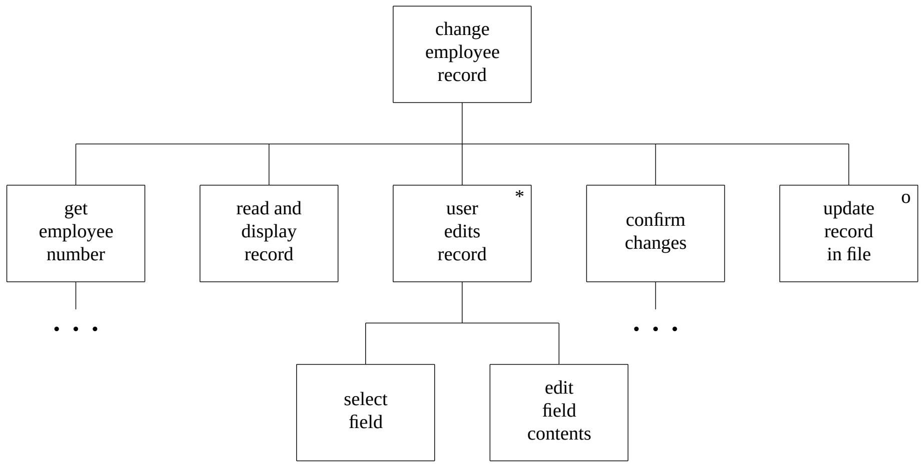

Figure 8.x5 *JSD diagram for updating a record*

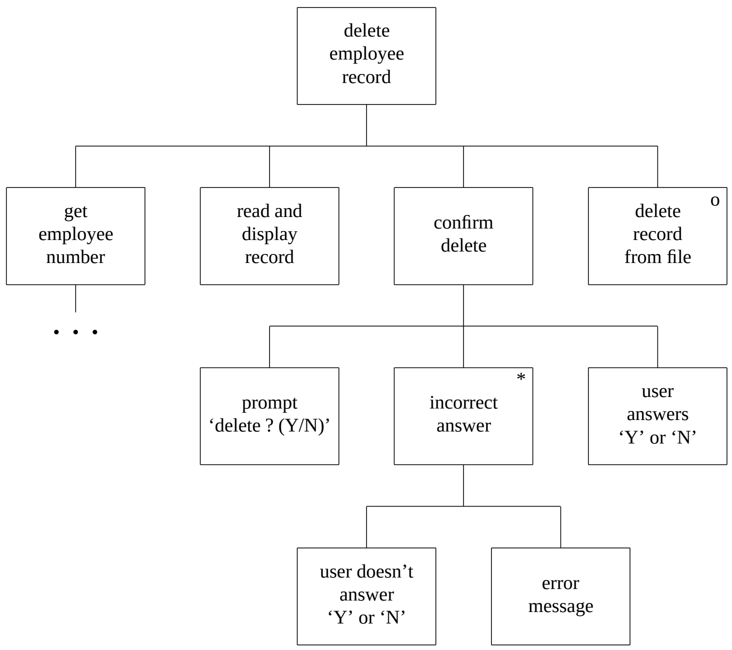

Figure 8.x6 *JSD diagram for record deletion sub-dialogue*

Answer

The intention of this exercise is not so much to produce a clean description, but to show how complex the business of talking about multi-window applications is. Most interface designers and programmers don't have to worry about designing the windowing system itself. However, it does help students to appreciate why areas like cut-paste buffers can behave oddly.

The applications can be anything. In fact, you may like to give the students some simple applications (e.g., the ones below), and then set this exercise where the emphasis is on modifications required to get the multi-window case working.

First of all, we will give two simple applications, a calculator and a database, then we will look at putting them together.

Calculator

In fact, calculator is a bit pretentious, this is really an adder! The user types in either 'quit' which exits the application, 'zero' which resets the adder to zero or a number which is added to the running total.

Adder = add-prompt → (quit? → **skip**
[] zero? → show-sum → Adder
[] number? → show-sum → Adder)

The semantic description is given in eventISL.

event: add-prompt =
prompt: true
out: "next number"

event: quit? =
uses: input
when: input = "quit"

event: zero? =
uses: input
when: input = "zero"
set: sum = 0
out: "reset"

event: number? =
uses: input, sum
when: is-number(input)
set: sum = sum + input

event: show-sum =
uses: sum
out: sum

Database

Another simple application could be a database, where the user actions are either 'quit', 'set *key value*', or 'find *key*'.

Database	=	db-prompt →	(quit? → **skip**
			[] set? → Get-key; Get-val
			[] get? → Get-key; print-val)
Get-key	=	set-key-prompt → get-key?	
Get-val	=	set-val-prompt → get-val?	

We won't bother with the semantic descriptions of the prompt events, or quit which is the same as for the adder. The descriptions of 'set?' and 'get?' are also similar to those for 'quit?', so are omitted.

event: get-key? =
uses: input
  set: key = input

event: get-val? =
uses: input, dbase, key
  set: dbase = db-update(dbase,key,input)

event: print-val =
uses: dbase, key
  out: db-lookup(dbase,key)

The functions 'db-update' and 'db-lookup' are assumed to add a new key-value pair and to lookup the value associated with a particular key, respectively.

Multiple windows

Having got these two description, putting them together is quite simple:

Window-system = Adder | | Database

However, it would be considerably more complex if we wanted two adders. Although this sounds a bit odd, if it were two text editing windows or two spreadsheet windows, this would be quite reasonable. We might attempt to do it simply as:

Window-system = Adder | | Adder

However, this would not work as the event names and the variables in the semantic description are global. The windows would both add to the same sum and it would be random as to which window responded to particular user events. This is an example of unintentional interference. The only way around this is to completely copy the description of Adder with new names for most of the events and variables.

Students can also be encouraged to look for ways of addressing deliberate sharing. For example, a common cut-paste buffer can be achieved using shared variables and explicit dependencies between dialogues can used internal events as described in section 8.4.3.

The modified dialogue below has an extra command 'copy' in the database, which is like 'get' except it also puts the value from the database into a shared buffer and invokes the 'buffer-ready' event, which tells the adder to add the value in the buffer to its running total.

Database	=	db-prompt →	(	quit? → **skip**
			[]	set? → Get-key; Get-val
			[]	get? → Get-key; print-val
			[]	copy? → Get-key; Send-val)
Get-key	=	set-key-prompt → get-key?		
Get-val	=	set-val-prompt → get-val?		
Send-val	=	set-buffer → buffer-ready		

The event 'copy?' is again a simple test of the input and 'buffer-ready' is described below with the modified adder.

event: set-buffer =
uses: dbase, key
  set: val-buffer = db-lookup(dbase,key)
  out: db-lookup(dbase,key)

The adder is modified to respond to the event from the database window.

Adder = add-prompt →
  (quit? → **skip**
  [] zero? → show-sum → Adder
  [] number? → show-sum → Adder)
  [] buffer-ready → show-sum → Adder)

event: buffer-ready =
uses: val-buffer, sum
  set: sum = sum + val-buffer

While your students work on this exercise, you can also discuss issues such as how to represent the changes in keyboard focus. This would probably require an extra keyboard multiplexer process.

Exercise 8.5

In the example of the digital watch in Section 8.3.8, what would be the dangerous states? Relate the lexical issues of the buttons for a digital watch to these dangerous states and provide some design advice. Does your own digital watch satisfy these criteria?

Answer

The time and alarm setting modes are dangerous states, in that we do not want to accidentally change either. Design advice would be to put some sort of guard which makes it difficult to accidentally get into the time setting modes, or alternatively, once within the modes, to make the buttons which advance the time difficult to press.

The watch in section 8.3.8 guards these states by requiring button 'A' to be pressed for two seconds before the mode changes. It might be easy to press a watch button accidentally, but is quite hard to hold it down. This solution would not of course work for a keyboard, where it is quite easy to hold a key down accidentally. Other watches guard these modes by insetting the mode change button or the buttons required to actually change the time. One has to press these buttons with a sharp instrument, such as a pencil – not an easy slip to make. Some digital clocks guard the mode by making you hold a button down continuously while you are changing the time.

Guarding the mode is far preferable to guarding the change buttons, as the latter makes it very hard to change the time when you want to do it.

In fact changing the time is often a problem for the *reversibility* of watch and clock dialogues. Typically there is a button to *advance* the time by one hour or one minute, but not one to put the time back. So, if you press the minute advance button once to often, you have to press it 59 more times to reverse the mistake!

Chapter 9

Models of the System

Exercise 9.1

Using the model-oriented approach with the example graphics program described in Section 9.2.3, specify the move operation as a schema which acts on the currently selected objects. Is the operation you have defined cumulative (two successive moves can be done as one move which is the sum of the other two) or is it 'forgetful?' Discuss the implications of the framing problem in your definition.

Answer

Students may not have a great familiarity with the Z notation used in this textbook, especially if they do not have a computer science background which emphasizes formal specification. Nevertheless, we will provide a model answer to this exercise in the Z notation and leave it to the instructor to decide whether an alternative language can be used to express the same ideas.

To accommodate the specification of the move operation, we will first need a function which allows us to add two coordinate points together. We just provide the declaration of this function here.

$$\_ + \_ : Point \times Point \rightarrow Point$$

The motivation for this function is so that we can define a move operation with a single argument, called *delta*? below, which will specify how far to move the centres of every selected shape. This approach, using relative movements, will allow for two consecutive moves to be defined with the overall movement being the sum of the individual moves. Another approach (which would be quite cumbersome to define for a set of selected objects) would be to specify the move operation as an absolute operation, the argument(s) to the move explicitly indicating the new value for the centre(s) of each selected shape. This operation definition would not be cumulative.

Initially, we might want to define the move operation simply to update the centre position of all selected shapes.

$$
\begin{array}{|l}
\textit{Move} \\
\hline
\textit{State} \\
\textit{State}' \\
\textit{delta}? : \textit{Point} \\
\hline
\forall \textit{id} : \textit{selected} \bullet \textit{shapes}'(\textit{id}).\textit{centre} = \textit{shapes}(\textit{id}).\textit{centre} + \textit{delta}? \\
\hline
\end{array}
$$

There are a number of details in this defined operation which are left (dangerously) open to interpretation. Many of these are framing conditions on the *Move* operation. For example, what happens to shapes which are not selected? The normal intuition on this operation would be that they stay put, but the above definition actually allows them to change in any way — including the possibility of disappearing altogether. To ensure that unselected objects stay the same as they were before the operation, we would have to add a predicate to the definition.

$$
\begin{array}{|l}
\textit{Move} \\
\hline
\textit{State} \\
\textit{State}' \\
\textit{delta}? : \textit{Point} \\
\hline
\forall \textit{id} : \textit{selected} \bullet \textit{shapes}'(\textit{id}).\textit{centre} = \textit{shapes}(\textit{id}).\textit{centre} + \textit{delta}? \\
\forall \textit{id} : (\mathrm{dom}\, \textit{shapes} - \textit{selected}) \bullet \textit{shapes}'(\textit{id}) = \textit{shapes}(\textit{id}) \\
\hline
\end{array}
$$

We also need to explicitly state that no new shapes are introduced into the system, i.e., that the domain of *shapes* remains the same.

$$
\begin{array}{|l}
\textit{Move} \\
\hline
\textit{State} \\
\textit{State}' \\
\textit{delta}? : \textit{Point} \\
\hline
\forall \textit{id} : \textit{selected} \bullet \textit{shapes}'(\textit{id}).\textit{centre} = \textit{shapes}(\textit{id}).\textit{centre} + \textit{delta}? \\
\forall \textit{id} : (\mathrm{dom}\, \textit{shapes} - \textit{selected}) \bullet \textit{shapes}'(\textit{id}) = \textit{shapes}(\textit{id}) \\
\mathrm{dom}\, \textit{shapes}' = \mathrm{dom}\, \textit{shapes} \\
\hline
\end{array}
$$

For those familiar with Z, it is possible to collapse the last three predicates in the above schema to one predicate using functional overriding from the basic Z mathematical toolkit (see Spivey, J.M., *The Z Notation: A Reference Manual*, Prentice Hall).

Another framing issue which has impact on the interactive style of this graphics system is specifying what happens to the selected set of shapes after the move operation. Our current definition says nothing about this. Two clear options are whether the set of selected shapes remains the same or becomes the empty set (no selected objects). For reasons of compositionality (for defining consecutive move operations or consecutive operations targeted at the same set of shapes) we choose the former option, and specify this below.

Move

$State$
$State'$
$delta? : Point$

$\forall id : selected \bullet shapes'(id).centre = shapes(id).centre + delta?$
$\forall id : (\mathrm{dom}\, shapes - selected) \bullet shapes'(id) = shapes(id)$
$\mathrm{dom}\, shapes' = \mathrm{dom}\, shapes$
$selected' = selected$

[**Note**: Experienced Z users would probably choose the functional overriding operator ($\oplus$) to more concisely express the first three predicates above.]

Exercise 9.2

Write a similar schema for *RESIZE*. It should have the following informal semantics. The width and height attributes are of the shapes bounding box. The resize operation should make one corner of the bounding box be at the current mouse position (supplied as the argument $current\_pos?$).

Hint: the width of a box is twice the difference between the x coordinate of the centre and the x coordinate of any corner:

$$wid = 2 * \mid centre.x - corner.x \mid$$

Answer

Strictly speaking, there are some errors in the Z usage for the hint in this exercise. The type *Point* is not a schema type, and so it is not meaningful to use schema selection, as in expressions like $centre.x$ and $corner.x$. The alert Z student should pick up on this error. However, spirit of the hint is still valid. To make things legal in Z, we can change the declaration of *Point* as follows.

Point

$x, y : \mathcal{R}$

To best specify this resizing operation, we will require to explicitly model the bounding box of a shape. We introduce a type, called *BoundingBox* which will represent the corners of the surrounding box which defines the extent of a shape. In fact, in Figure 9.1,we have drawn the bounding box for the circle but we have not made that information explicit in the specification in the text of this chapter. We omit details in this sample answer on "rectangular" constraints for the corners of the bounding box (e.g., that upper left and right corners have the same y-coordinate values but the x-coordinate of the upper left corner is strictly less than the x-coordinate of the upper right corner, *etc*.

$$
\begin{array}{|l}
\multicolumn{1}{l}{\textit{BoundingBox}} \\
\hline
\textit{upperleft}, \textit{upperright}, \\
\textit{lowerleft}, \textit{lowerright} : \textit{Point} \\
\hline
\textit{upperleft}.x \leq \textit{upperright}.x \\
\textit{upperleft}.y = \textit{upperleft}.y \\
\ldots \\
\hline
\end{array}
$$

We can add the bounding box as information known about shapes, by extending the definition of *Shape*. A constraint on the bounding box, motivated by the hint to this exercise, is that its width and height are the same as the width and height of the shape. In addition, the centre coordinates are related to the width and height. Note below that we use the names *height* and *width* in the definition of *Shape* to increase its readability.

$$
\begin{array}{|l}
\multicolumn{1}{l}{\textit{Shape}} \\
\hline
\textit{type} : \textit{Shape\_type} \\
\textit{width} : \mathcal{R} \\
\textit{height} : \mathcal{R} \\
\textit{centre} : \textit{Point} \\
\textit{bb} : \textit{BoundingBox} \\
\hline
\textit{width} = 2 \times (\textit{centre}.x - \textit{bb}.\textit{upperleft}.x) \\
\textit{height} = 2 \times (\textit{centre}.y - \textit{bb}.\textit{upperleft}.y) \\
\textit{width} = (\textit{bb}.\textit{upperleft}.x - \textit{bb}.\textit{upperright}.x) \\
\textit{height} = (\textit{bb}.\textit{upperleft}.y - \textit{bb}.\textit{lowerleft}.y) \\
\hline
\end{array}
$$

Resizing can be viewed as changing the size of the bounding box. At a relatively abstract level, we can simply specify the resizing operation with a single argument indicating the new bounding box for the selected shape. A precondition for the operation is that there is only one selected shape.

Resize

$State$
$State'$
$newbb? : BoundingBox$

$\#selected = 1$
$\forall\, id : selected \bullet shapes'(id).bb = newbb?$
$forallid : (\operatorname{dom} shapes - selected) \bullet shapes'(id) = shapes(id)$
$\operatorname{dom} shapes' = \operatorname{dom} shapes$
$selected' = selected$

Exercise 9.3

In Section 9.2.5, we said that the specification $Algebraic - Zdraw$ could be extended to say that a move, resize or unselect after a delete has no effect. The axiom for *unselect* looks like this:

$(ix)\quad unselect(delete(g)) = delete(g)$

Write two more axioms (x) and (xi) which say the same about move and resize. Now show use axioms (iv) and (v to show that (ix) implies both your new axioms.

Answer

The three additional axioms are:

(9) $unselect(delete(st)) = delete(st)$
(10) $move(p, delete(st)) = delete(st)$
(11) $resize(p, delete(st)) = delete(st)$

Axioms 4 and 5 are:

(4) $move(p, unselect(st)) = unselect(g)$
(5) $resize(p, unselect(st)) = unselect(g)$

To show that Axiom 10 is derivable from Axioms 4 and 9 we can argue as follows:

$$
\begin{aligned}
&move(p, delete(st)) \\
&\quad = move(p, unselect(delete(st)) && \text{by Axiom 9} \\
&\quad = unselect(delete(st)) && \text{by Axiom 4} \\
&\quad = delete(st) && \text{by Axiom 9}
\end{aligned}
$$

A similar argument shows that Axiom 11 is derivable from Axioms 5 and 9.

Exercise 9.4

A two-function calculator has the following buttons `0`–`9`, `+`, `-` and `=`. These would comprise the command set (C) of a PIE model of the calculator. The state (E) would consist of three components: the two current number being entered, the last number (to be operated on) and the pending operation (+ or −). The display (D) is simply a signed number. For this example, ignore the result.

Write down (semi-formally) the *doit* function updating the state for each user command and the *display* function relating the state (E) to the current display (D). To check your definitions: what does the display have on it after the user has entered '`2+3+`'? Most calculators would show 5, does yours?

Consider the displays after the sequences '`2+2`' '`2+`' and the effect on each of the additional user input of `3`. Does the calculator satisfy the transparency property?

Answer

The definition of the *doit* function could proceed as follows. We will use the Z notation in this sample answer. The character set is defined as:

$$C ::= 0 \mid 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9 \mid 0 \mid + \mid - \mid =$$

It will be useful to distinguish the digits and operations (+ and −).

$$\begin{array}{|l}
Digit : \mathcal{P}\, C \\
Op : \mathcal{P}\, C \\
\hline
Digit = \{\, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 0 \,\} \\
Op = \{\, +, - \,\}
\end{array}$$

The state is defined as an ordered four-tuple, consisting of the current number being entered,the last number (to be operated on) and the pending operation.

$$E == \operatorname{seq} Digit \times \operatorname{seq} Digit \times Op$$

The display is simply a projection of the current number being entered.

$$D == \operatorname{seq} Digit$$

$$display : E \to D$$
$$\forall (current, last, op) : E \bullet$$
$$\quad display(current, last, op) = current$$

The *doit* function is defined below by case exhaustion. The notation $o(c, l)$ means to apply operation o to the arguments c and l, so that $2 + 3$ would be written $+(2, 3)$, with $o = +$, $c = 2$ and $l = 3$.

$$doit : E \times C \to E$$

$$\begin{array}{llll}
\forall\, c, l : \mathrm{seq}\, Digit;\ d, d' : Digit;\ o, o' : Op \bullet & & & \\
\quad doit((c, l, o), =) & = & (o(c, l), 0, o) & \\
\quad doit((c, l, o), d) & = & (d, c, o) & \textit{if lastkey} \in \{+, -, =\} \\
 & & (10c + d, l, o) & \textit{otherwise} \\
\quad doit((c, l, o), o') & = & (o(c, l), c, o') &
\end{array}$$

The following table traces the execution of the calculator.

C	E	D
	(0,0,+)	0
2	(2,0,+)	2
+	(2,2,+)	2
3	(3,2,+)	3
+	(5,3,+)	5

Compare the trace tables for the 'programs' $2 + 3$ and $2 + 23$.

C	E	D
	(0,0,+)	0
2	(2,0,+)	2
+	(2,2,+)	2
3	(3,2,+)	3

C	E	D
	(0,0,+)	0
2	(2,0,+)	2
+	(2,2,+)	2
2	(2,2,+)	2
3	(23,2,+)	23

Notice that the display after the program 2+ is the same as it is after the program $2 + 2$, yet the behaviour of the subsequent 3 command is different. Therefore, the calculator as defined does not satisfy the transparency property.

Exercise 9.5

Imagine a calculator, normal except it displays A for 0, B for 1, up to J for 9. So the number `372` would appear as `DHC`. Does this affect the formal transparency of the calculator? Should it?

Answer

Formally, this would not affect the transparency property of a calculator. Any function from display to effect which held for the numbers would also hold for their alphabetic counterparts by just composing it with the code A for 0, B for 1, up to J for 9. Of course, most people would probably find using the encoded calculator quite a bit more difficult than one which used the normal ten digits. The important lesson here is that the formal expression of the transparency property does not indicate how difficult it is for a human to decide determine the function between display and effect. The formal expression just requires that there be some function. In practice, human users would probably judge the encoded calculator as unpredictable (that is, not transparent). The moral is that even though we can capture a usability property formally does not mean that we have captured the entire intent of the property. The transparency property would have to be augmented in order to measure the complexity from the user's perspective of the function between display and effect.

Exercise 9.6

> Can you suggest any improvements to the screen button feedback problem discussed in Section 9.4 that would distinguish at the interface between the two cases of hitting or missing the button? Is there any guarantee with your solution that the user will notice the distinction?

Answer

Some suggested fixes to the button feedback problem would be to have the function attached to the button be invoked when the mousekey is pressed down, rather than up. But this solution would not work for functions invoked on pop-up menus. Another possibility is to have some visible or audible feedback from the button associated with the invocation of the function on the mousekey release. This solution does not guarantee that the user notices the added effect unless their attention is focused on the added visual or aural effect. Another possibility, subtly different from the last suggestion, would be to associate the addition verbal or aural cue to the error case, when the mouse accidentally slips off the button between the press and release of the mousekey.

Chapter 10

Implementation Support

Exercise 10.1

In contrasting the read–evaluation loop and the notification based paradigm for interactive programs, construction of a preemptive dialogue was discussed. How would a programmer describe a preemptive dialogue by purely graphical means?

Answer

Creating a preemptive dialogue by graphical means is quite simple. In a graphic-based interaction, it is assumed that the user can only interact with parts of the system that are visible. In a windowing system, for example, the user can only direct input to a single, currently selected window and the only means of changing the selected window would be by indicating with some gesture within that window. To create a preemptive dialogue, the system can produce a window which covers the entire screen and make it the currently selected window. All user input would then be directed to that window and the user would have no means of selecting another window. The "covering" window thus preempts any other user action except that which it is defined to support.

Exercise 10.2

Recall the state transition diagram for font characteristics presented in Chapter 8 (Section 8.3.3). Compare different interaction objects which could implement this kind of dialogue. Use examples from existing toolkits (pull-down menus or dialogue boxes) or create a novel interaction object.

Answer

To recall the example, we want to allow the user to toggle between three font characteristics—bold, italics and underline. Pull-down menus and dialogue boxes are the most frequently used methods for implementing such a dialogue. A pull-down menu would have the three choices listed with some way of indicating whether the characteristic was active or not. Upon each invocation of the menu, the user could change the active status of one font characteristic. For example, they could turn on the bold feature or turn off the underline feature. The important feature of this dialogue is that more complex tasks (changing the active status of two or more font characteristics) can only be achieved by separate invocations of the menu. And often the only way the user can check the status would be by popping up the menu again (without invoking any action). This last point shows that the menu design alone does not support immediate display of relevant state changes as discussed in Chapter 4 under the synthesizability principle.

Within a dialogue box, the user can affect several changes at one invocation. Technically, it is not the dialogue box interaction device which allows for this, but the use of toggle buttons. Each font characteristic is represented by a box which acts as a simple on/off switch for that characteristic. As an aid to the user, the toggle button can change its appearance so that the user can know its status (on or off) by looking at it. A similar interaction widget is the checkbox, which would be labelled with the particular font characteristic. Clicking on the checkbox would put a mark in the box to indicate that the characteristic is on. Another click would remove the mark. In fact, a dialogue box need not be used to portray the set of toggle buttons or checkboxes, as they could be presented continually so that the user can always view the current font characteristic settings. The reason a dialogue box might be used is because there may be many font characteristics to choose from and the dialogue box will keep them grouped together and preserve screen space by allowing them to be hidden and used on an as-needed basis.

Exercise 10.3

Scrolling is an effective means of browsing through a document in a window that is too small to show the whole document. Compare the different interactive behaviour of the following two interaction objects to implement scrolling:

- A scroll bar is attached to the side of the window with arrows at the top and bottom. When the mouse is positioned over the arrow at the top of the screen (which points up), the window frame is moved upwards to reveal a part of the document above/before what is currently viewed. When the bottom arrow is selected, the frame moves down to reveal the document below/after the current view.

- The document is contained in a textual interaction object. Pressing the mouse button in the text object allows you to drag the document within the window boundaries. You drag up to browse down in the document and you drag down to browse up.

The difference between the two situations can be characterized by noticing that in the first case, the user is actually manipulating the window (moving it up or down to reveal the contents of the document), whereas in the second case, the user is manipulating the document (pushing it up or down to reveal its contents through the windows. What usability principles would you use to justify one method over the other (also consider the case when you want to scroll from side to side as well as up and down)? What implementation considerations are important?

Answer

There are many usability principles which can be brought to bear on an examination of scrolling principles. For example:

Observability The whole reason why scrolling is used is because there is too much information to present all at once. Providing a means of viewing document contents without changing the contents increases the observability of the system. Scrollbars also increase observability because they help to indicate the wider context of the information which is currently visible, typically by showing where the window of information fits within the whole document. However, observability does not address the particular design options put forth here.

Predictability The value of a scrolling mechanism lies in the user being able to know where a particular scrolling action will lead in the document. The use of arrows on the scrollbar are used to help the user predict the effect of the scrolling operation. If an arrow points up, the question is whether that indicates the direction the window is being moved (the first case) or the direction the actual text would have to move (the second case). The empirical question here is to what object do user's associate the arrow, the text or the text window? The arrow of the scroll bar is more closely connected to the boundary of a text window, so the more usual interpretation would be to have it indicate the direction of the window movement.

Synthesizabilty You might think that it does not matter which object the user associates to the arrow. They will just have to learn the mapping and live with it. In this case, how easy is it to learn the mapping, i.e., can the user synthesize the meaning of the scrolling actions from changes made at the display? Usually, the movement of a box within the scrollbar itself will indicate the result of a scrolling operation.

Familiarity/Guessability It would be an interesting experiment to see whether there was a difference in the performance of new users for the difference scrolling mechanisms. This might be the subject of a more extended exercise.

Task conformance There are some implementation limitations for these scrolling mechanisms (see below). In light of those limitations, does the particular scrolling task prefer one over the other. In considering this principle, we need to know what kinds of scrolling activity will be necessary. Is the document a long text that will be browsed from end to end, or is it possibly a map or a picture which is only slightly larger than the actual screen so scrolling will only be done in small increments?

Some implementation considerations:

- What scroll mechanisms does a toolkit provide? Is it easy to access the two options discussed above within the same toolkit?
- In the case of the second scrolling option, are there enough keys on the mouse to allow this operation without interfering with other important mouse operations, such as arbitrarily moving the insertion point or selecting a portion of text or selecting a graphical item?
- In the second option, the user places the mouse on a specific location within the window, and gestures to dictate the movement of the underlying document. What kind of behaviour is expected when the mouse hits the boundary of the window? Is the scrolling limited in this case to steps bounded in size by the size of the window, so that scrolling between two distant points requires many separate smaller scrolling actions?

Chapter 11

Evaluation Techniques

Exercise 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.

Answer

This exercise is intended to give the students a feel for using the technique of cognitive walkthrough (CW). CW is described in detail in the chapter and the same format can be used here. It is important to focus on a task that is not too trivial, for example, creating a style in a WP package. Also assume a user who is familiar with the notion of styles (and with applications on the same platform (e.g. Macs, PCs, UNIX etc.) but not with the particular word processing package. Attention should be given to instances where the interface fails to support the user in resolving the goal and where it presents false avenues.

Exercise 11.2

> 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. Subject is 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

Exercise 11.3

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.x1 shows an example questionnaire.

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).

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:

Table 11.x1 *Questionnaire to compare two systems*

Subjects user group

IV style of learning system

DV performance (measured as test score).

Design Between groups design.

Exercise 11.4

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?

Answer

The benefits of video include: accurate realistic representation of task performance especially where more than one video is used; a permanent record of the observed behaviour.

The disadvantages include: vast amounts of data which are difficult to analyze effectively; transcription; obtrusiveness; special equipment required.

The student can try transcribing a video record — conversation from a TV soap will do. The aim is to get the student to experience some of the difficulties of representing a visual record in a semi-formal written format. The students should discuss which parts of the video are most difficult to represent, and how important these parts are to understanding the clip.

Exercise 11.5

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.

Answer

The students should read carefully the description of the experiment in Section 11.5.1 to look for ways of reducing the possible interference. Three possibilities are:

- During the initial training period swop back and forth between learning the two sets of icons, with the aim of getting the subjects used to swopping between the two sets of remembered icons. However, this design could be argued to suffer the same flaws as the original. If the abstract icons had been taught in isolation perhaps they might have fared far better.
- One could invent a third set of 'random' icons (call them R). We could then interpose them in the experiment, that is present the icons in the orders RARN and RNRA. The intention is to swamp any transfer effect in the 'noise' of the random icons. It could be argued that our experiment then measures the robustness of the icon sets to such 'noise'!
- We could give the subjects multiple presentations, for example, ANAN and NANA presentation orders. This would not remove transfer effects, but does give us some way to quantify them. Imagine that in the ANAN group that the second presentation of the abstract icons was significantly worse than the first, but there was not a similar effect for natural icons in the NANA group. This would both give us positive evidence of a transfer effect, and perhaps give us some quantitative measure. However, even going from this additional evidence to a strong conclusion will be difficult.

Notice that all the above measures require additional subject time and one has to constantly weigh up the advantages of richer experiments against those of larger subject groups.

Chapter 12

Help and Documentation

Exercise 12.1

> Write a manual page for making a cup of coffee. Assume your user has no experience but will recognize a cup, a kettle, a spoon, etc. Swap your manual with a partner. Does your partner's manual give you sufficient instruction to make the cup of coffee? Discuss improvements with your partner and agree on a final version of the manual.

Answer

First you have to decide upon the level of granularity which you are going to work at. The aim of the exercise is to demonstrate that it is not as straightforward as it might seem to provide instructions even for a very familiar and well understood task. Given this, it is most helpful to assume that the user knows very little. The example solution (Table 12.x1) assumes that the user will recognize objects (perhaps they are labelled) and understands common actions and directions, but not the specific actions required here. Alternatively you could choose to assume that the user does know how to turn on a tap and open a jar.

N.B. The two options could be expanded further if required. Other alternatives could be included such as getting water from another source.

The discussion should focus on the assumptions that are made. The student should make a conscious decision as to what assumptions to make, rather than making them by default. This should ensure that the assumptions that are made are appropriate to the particular user.

The students could also contrast their answers here, based on the material in this chapter, with the 'making tea' manual in Chapter 7. How useful do they feel task analysis is in designing a manual?

Manual for making a cup of coffee

Required:	an automatic electric kettle, a jar of instant coffee powder or granules, a mug, a teaspoon, a mains water tap (or an alternative source of water), a mains electricity supply, milk (optional), sugar (optional).
CAUTION:	Electricity can be dangerous — avoid any contact between electric connections and water. Boiling water can scald — take care.
To boil water:	
	Ensure plug on kettle is not connected to mains electricity supply. Remove lid from kettle. Place kettle directly beneath spout of tap. Turn tap handle anticlockwise to release water. When water reaches mark labelled 'full' on kettle turn tap handle clockwise to close off water supply. Replace kettle lid. Move kettle to proximity of mains electricity supply. Place kettle's plug into electricity socket. Press button marked 'on' on top of kettle to switch kettle on. When the water has boiled, the kettle will switch off automatically and the 'on' button will return to its original position with a click. Remove kettle's plug from electricity socket.
To make coffee:	
	Take jar of coffee and remove lid by turning anticlockwise. Fill teaspoon with coffee. Place contents of teaspoon in cup. Replace lid on coffee jar and turn clockwise to tighten. Pour boiling water from the kettle into the cup up to approximately 1/2 an inch from the top. Add milk if required (to almost fill cup). Add sugar if required. Stir coffee mixture with spoon. The coffee is now ready to drink.

Table 12.x1 *Coffee making manual*

Exercise 12.2

Find a computer application that you have never used before. Attempt to learn to use it using only the on-line support. Is there enough information to allow you to use the application effectively? Is the information easy to find? What improvements (if any) would you suggest?

Answer

This is an investigative exercise for which there is no example solution. Possible systems to consider would be a word processor (for example, Word, WordPerfect, Word for Windows), a graphics package (MacDraw, Superpaint, CorelDraw) or a spreadsheet (Excel, Lotus). It may be helpful to provide the student with a list of tasks to learn to perform using the application. Include in this list basic tasks (creating a document, spreadsheet etc.) and more complex ones (creating templates, etc.). The aim is to get the student to think about the provision of help from the point of view of the solitary user trying to figure things out alone. The student should therefore make explicit any prior knowledge used to help interpret the system (for example, use of similar systems).

Exercise 12.3

What knowledge is needed to build an intelligent help system? Which do you think is most difficult to provide and why?

Answer

It requires knowledge of the domain and knowledge of the user at least. It may also require knowledge of teaching strategies and tasks. Knowledge of the user is usually most difficult to provide along with knowledge of the task. Even if the user can be monitored, interpreting the user's behaviour in anything other than coarse terms is difficult without access to his thought processes. However this level of interpretation, and generalization, can be helpful nonetheless (for example, a record of what the user has already done successfully or how often he has used the application). Context can be deduced from the user's current activity. This too can be difficult to determine however, unless it is a simple case where the tool in use indicates the activity being performed.

Domain knowledge in the general sense is probably the easiest to provide (although it is time consuming to do so). It is available within the system itself or from the designer.

Exercise 12.4

Look at as many on-line support systems as you can. Which do you find most useful and why? Try to assess them using the requirements discussed in Section 12.2.

Answer

This is another open investigative assignment. The set of requirements in Section 12.2. should provide the student with a structure for evaluating the systems. A scoring system can perhaps be worked out to aid comparison. Try to compare a general context sensitive help system (for example, Balloons help) with an application specific one. Also a hypertext based help system would provide an interesting contrast as would a manual based help system (such as UNIX man) and a prompt based one.

Chapter 13

Groupware

Exercise 13.1

> We discussed the use of a group pointer in a shared editor with a shared view. Consider the advantages and problems of using a group pointer when participants have different views. How do you show the pointer if it is outside part of the document you are working on? Think also about the issues when the system is a hypertext based co-authoring system. Is there any use for a group pointer in this case?

Answer

The purpose of this exercise is to encourage the students to think of the various scenarios which can arise during group editing and to encourage their creative interface thinking. Where suitable tools are available, the students can be encouraged to create mock-ups of their solution using a drawing package, or even envisionments of specific scenarios using HyperCard or a presentation tool. Below are some ideas.

We can start off with the shared editor shown in Figure 13.7. Imagine if your colleague wanted to point to the line starting 'Thinking about'. She could select the 'group pointer' tool at which point she can drag around an icon shaped like a hand. She moves this over the words 'Thinking about' and starts to talk about the text (using some other means of direct communication such as the telephone). As the phrase in question is on both displays there is no problem about where to display the hand, it is placed over the words 'Thinking about' on your display. (Note that the group pointer is at a different physical location on each display).

Arguably the group pointer is even more important when there are separate views then when the view is shared. With a shared view the participants can be assured that they are seeing the same thing and so can assume a shared context. With separate views the participants are likely to have various problems when they forget that the others' views are different. For example, they will have trouble with indexical expressions or with relative statements 'the thing at the top'. The presence of a group pointer can help the participants to gain a shared focus.

On the other hand, the very presence of the group pointer can encourage the participants in a false sense of shared context. One could therefore argue that when participants want to enter a period of shared focus and synchronous discussion that they also move to a shared view. However, for the rest of this answer we'll assume that the views stay different.

What do we do when the group pointer is outside of the view? For example, in Figure 13.7, your colleague might move the group pointer to the line starting 'More adaptable'. In this case, your screen cannot display the group pointer in its correct position. Various options suggest themselves.

One could have the hand icon appear at the very top of your display with the finger pointing straight upward, meaning 'see above'. Similarly if the group pointer were below the screen the icon could appear at the bottom pointing down. If such icons were displayed then the user can scroll the screen upwards until the group focus was found. Alternatively, one could arrange that clicking on the icon scrolled one to the correct position.

Instead or in addition to these 'out of view' icons, one could use the scroll bar and display the group pointer's position in it, thus allowing the user to easily see where it is and scroll there if appropriate.

In the hypertext case, the solutions depend very much on the particular style of the system. If the individual nodes are large, then even if two participants are acting on the same node they may have different views, in which case all the above measures become relevant. If the participants are looking at different nodes, then the first action of a group pointer should be to direct attention to the node of interest. In a hypertext where the node graph is shown, this can be achieved by displaying the group pointer on the node of interest. Alternatively, if the graph is not displayed, the presence of a group pointer could be shown by adding a 'go to the focus node' button, rather like the 'home' button in many HyperCard stacks.

Although it is not mentioned in the question, it is worth also considering the display of other people's insertion points. In Figure 13.7, although your insertion point is within your colleague's view, it is not displayed in any way. There is no reason why other people's insertion points should not be displayed (suitably identified) when they fall within your view and *visa versa*. In addition, one might want some indicator on your scroll bar to show where your colleague's current view is. Such indicators can enhance *group awareness* and be a focus for initiating direct communication.

Exercise 13.2

Find out how many different forms of direct computer-mediated communication are available on your system (start with email). Are they heavily used, and if so, where do they fit in the time/space matrix (Figure 13.8) and its refinement (Figure 13.9).

Answer

Obviously the answers to this and Exercise 13.3 depend on your own particular facilities, but here are some suggestions based on the authors' sites.

The question says 'starting with email'. In fact, on many systems several email interfaces are available. On one of the authors' system there are two command line interfaces as well as various GUI interfaces for different window managers (under X, on the Macintosh etc.). Although email is just one communication mechanism, it is used in a wide variety of ways. Its 'normal' use is essentially asynchronous/remote, or in terms of the refined matrix unsynchronised. However, delivery over a LAN is often virtually instantaneous and you can observe users engaging in effectively synchronous communication. The possibility of these modes of use is often determined by the style of the interface as is shown in section 9.4.4. Clearly, the way in which new mail is notified and the speed with which mail can be read both determine the pace at which the conversation can proceed. A further complication in some sites is that internal and external email have entirely different interfaces: sending mail off site may involve remotely logging into a gateway machine and then using some obscure mailer there, rather than the familiar workstation based tools.

In addition to email, you may have one or more text based synchronous communication tools. For example, many UNIX machines have two tools 'write' and 'talk'. The command 'write Alison' followed by some text prints that text on Alison's screen. She can respond by doing a 'write' back to the sender. The 'talk' command establishes a more continuous conversation, as described in section 13.3.1, where both participants screens are split in two, each half displaying one of the participants contributions. Although, both can be seen as synchronous/remote, they operate at different *levels of sharing* (section 13.6.2). Whereas 'write' only sends the text after the contribution is completed, 'talk' sends each user's contributions character by character as they are typed. That is 'talk' sits further down towards the bottom left of the diagram in Figure 13.10. Note also that 'write' can be used in a semi-asynchronous mode. If the other user is logged in, but not at their terminal, the message waits there until their return.

Another asynchronous communication tool commonly available is bulletin boards, such as Usenet news. As discussed in Section 13.3.1, these have different properties to email, being rather less personal.

In addition, to these standard tools, some of the authors' colleagues have permanent video connections between their offices. You may have similar facilities, or other computer augmented communications, perhaps video in a window on your workstation. Don't forget apparently low-tech things such as computers which automatically dial telephone calls.

Exercise 13.3

> Repeat the previous exercise, but this time look for shared data on your system. Is the data updated by one person and viewed by many, or have you files or databases which are updated by several people? If the latter, find out what methods are used to prevent two users changing the same data at the same time. There may be no mechanism at all, a computerised one (e.g., locking) or a social protocol (e.g., a floppy disk is passed around).

Answer

There is an argument that bulletin boards are a form of shared data, so should certainly be acceptable as an answer to this question, depending on their nature at your site and also on how well argued the answer is.

You are likely to come across a lot of loosely shared items, where only a small group access them. For example, the files which comprise the book were shared between the authors. Whether any sort of locking is used will depend on the facilities available in your installation, but typically such small group sharing is handled by social protocols (sometimes unsuccessfully!).

Here are some more specific examples from one of the authors sites:

There are centralised databases containing phone numbers and car numbers. The former is used in place of the internal telephone directory and the latter when you notice some lights have been left on in the car park! Both of these are updated centrally by one of the administrative staff. Changes are notified (by email) to the administrator. However, access to the information is available to all.

Each user has a 'plan' file in their local directory into which she can put generally informative remarks about her whereabouts. Typically, this might include a contact telephone number, regular appointments a simple diary, or indeed anything of interest. When another user enters the command 'finger Alison', Alison's plan file is displayed. As the file is owned by the individual user, normal access permissions apply and so typically a user can only update their own 'plan' file. However, through the 'finger' command the information is publically available.

There are also examples of publically updatable shared files. For example, some email groups are handled by having a file containing the email address of each recipient. Anyone can edit the file to add or remove their name from this list. (In fact, they can just as easily add or remove someone else's name!) On our system this file has no form of protection from simultaneous update, the assumption being that such updates are so rare that they will never happen (really).

The lack of shared data with locking is a reflection of the poor level of system support in the UNIX world. Users of PC based software should be able to find examples of locking of shared files or databases.

Finally, it is worth discussing the use of implicitly shared information. For example, the 'finger' command in UNIX also tells you whether the user is logged in and, if so, where. This is information known to the system, but made available to the users in such a way as to facilitate cooperation.

Chapter 14

CSCW Issues and Theory

These are both 'mini-project' exercises, so do not have definitive answer. here are some pointers to look out for.

Exercise 14.1

In Section 14.3.2, we discussed the highly contextual nature of the spoken word, including the use of deictic reference and indexicals, and the (officially) ungrammatical and fragmentary use of sentences. Try listening to social chat over cups of tea — collect examples of different forms of contextual utterance.

Answer

There are often objects of shared attention during social chat, perhaps a notice board, or magazines on a coffee table. These will be one obvious source of context. Make sure that when the students record statements such as 'I like that', they also record whether the indexical refers to a previous utterance or to something external. Encourage them to observe exactly how external references are made: for example, whether people point to a notice, refer implicitly to the article someone else is reading or use the direction of their own gaze.

In a small group, the students should listen particularly carefully for the use of the pronoun 'you'. Does it refer to the whole group (excluding the speaker), or to a particular person. If the latter, then how is it made explicit – eye gaze, body position? Other pronouns like 'we' or 'she' can be be equally contextual.

The students should also look out for implicit context, for example, 'a longer tail would improve the aerodynamics', spoken when the listener is looking at a picture of a kite. Although the sentence has no pronoun there is an implicit subject 'of the kite you're looking at'. This form of contextual statement can become arbitrarily obscure, for example, as one person meets another: 'it's in your pigeon hole' — referring to an overdue report.

As the students become more adept they can listen more carefully looking for the shift in conversational focus, and breakdowns where the context is not successfully negotiated.

Exercise 14.2

Go into an office or other place where several people are working together. Try to note down in as much detail as possible what they are doing and when. Do this with different foci: focus on the direct interpersonal communications, focus on the shared objects such as a calendar or document, or focus on one worker at a time. Whilst collecting data and when ordering your notes, look for breakdowns and misunderstandings, and for implicit communication through objects. Look also at a particular task over a period of time, and note the number of interruptions as a worker performs the task, or the way a single task is contributed to by several workers.

Answer

This exercise is similar to Exercise 7.5. However, whereas the task analysis in 7.5 is quite structured this exercise is not intended to produce the same form of precise task sequences etc. The intention is to expose the student the vast range of social situations within a typical work environment. However, the students can be directed to similar recording techniques, as described in Chapter 11.

In particular, this exercise is a good chance to introduce the use of video or audio-recording equipment if it is available. The students will soon learn how difficult it is to position a microphone so that it picks up more than one worker and so that it does not get drowned by the sound of a typist. Similarly, they can gain first hand experience of the problems of static camera positioning. If you are lucky, you may have access to multiple cameras and split screen recording equipment.

Although such equipment can be useful, its lack is not a disaster. Many ethnographic studies use only a pencil and paper, with perhaps some audio recording. However, developing appropriate shorthand then becomes essential (see Suchman's book for examples of appropriate notation).

Some forms of communication in the office can be quite subtle, perhaps 'overhearing' of conversations, or noticing when something is being written on a wall calendar. Developing an observing eye for such subtleties is largely a matter of practice.

Chapter 15

Multi-sensory Systems

Exercise 15.1

What factors are likely to delay the widespread use of video in interfaces? What applications could benefit most from its use?

Answer

Some of the factors are the costs in terms of hardware and software for compression and decompression; the slow speed due to the high bandwidth; the overall cost of equipment (for example, camera, video, CD); the lack of design tools to exploit video; the lack of specialist skills amongst designers. Many applications have been suggested as candidates for the integration of video. Educational systems, games and help systems are liable to benefit since information can be passed more clearly and memorably and new dimensions added. Other areas such as virtual reality can use video together with graphics in the creation of their artificial worlds. CSCW systems can use video to provide a face to face communication link between distributed workers (see Chapters 13 and 14). However, although these appear to be areas where video has a promising future, its use needs to be carefully considered and its consequences investigated. It may be that it will not fulfill its initial promise.

Exercise 15.2

In Section 15.11.1, we look at alternative interfaces for users with special needs. How could standard interface design be improved to provide better access to these users?

Answer

This depends on the particular needs of the user but in general the more an interface relies on a single channel for information (such as the visual channel) the more likely it is to be unusable by some people. Multi-modality and redundancy in the interface will alleviate this problem providing alternative views of the information and so not excluding a user group with restricted use of the primary channel. It is more satisfactory to encourage multi-modality as a design principle than to provide special systems or "add-onsùù to cater for extra-ordinary user groups. This can also benefit the average user since it reinforces the information over several channels. However this may not be possible in all cases: clearly, for example, a user who has limited control of hands and voice may need special equipment such as the Eyegaze system which will track eye movements. Where possible such specialized equipment should be integrated with standard applications to enable the disabled user to use readily available machines and software in common with other users. This is particularly important in the work environment.

Exercise 15.3

Think of a set of naturally occurring sounds to represent the operations in a standard drawing package (for example, draw, move, copy, delete, rotate).

Answer

This can exercise the imagination! Are there natural analogies? For example does the physical action say of drawing have a sound associate with it? The sound of a pencil on paper may be appropriate but is it identifiable? Similarly a photocopier whirring could represent the copy operation, and tearing paper delete. Rotate and move are more difficult since the physical operation is not associated with a sound. Perhaps direction and movement can be indicated by sounds becoming nearer or more distant?

Exercise 15.4

Experiment with Hypercard or another hypertext system if you have access to one. As you work through the system, draw a map of the links and connections. Is is clear where you are and where you can get to at any point? If not, how could this be improved?

Answer

This is an experimental exercise which requires access to a hypertext system. It can be used as the basis for a practical class, in which the students analyze the effectiveness of the system. Drawing he map has two purposes: one is to reinforce the overall structure of the hypertext; the other is to test the navigational support that is available. Whether it is sufficient will depend on the system under scrutiny, but possible improvements would be to provide an explicit map, escape buttons, explicit paths to core material. The system may of course incorporate such features.

Exercise 15.5

> Data visualization techniques have often increased our comprehension of phenomena: consider the effect that 3-D graphics has had on looking at complex models such as those of the atmosphere or the ocean, or in understanding the structure of molecules. What do you consider to be the areas that may benefit most from virtual reality visualization techniques?

Answer

This could be expanded to allow the student to carry out a brief literature review on the applications of virtual reality techniques. In general they are most promising in areas which cannot be explored in reality, either because it is physically impossible (such as in the manipulation of molecules discussed in Section 15.11.2) or physically dangerous, such as in space. Models can be physically manipulated and interacted with using virtual reality methods rather than simply observed. Chemical and medical research, meteorology, oceanography, seismology are all areas which could benefit from such techniques.

Exercise 15.6

> Virtual reality has found a number of applications in the games market. Is this a suitable use of such technology? Discuss the possible benefits and disadvantages of exploiting leading edge technology in a leisure market.

Answer

The leisure market is important to the computer industry and, being commercially lucrative, can provide an impetus for development which can then be exploited in other spheres. If virtual reality takes off in games it will result in the development of cheaper equipment and more robust techniques, since the games customer tends to be young (and not overly rich) but with sophisticated expectations. Indeed, many research projects in universities are currently using equipment developed specifically for games, since the more powerful, general purpose equipment is prohibitively expensive. Another possible advantage is that it makes the technology familiar and therefore accessible and acceptable.

A possible disadvantage is that the use of this technology in games may trivialise it so that it is not considered as a serious solution to other more weighty problems. However, this seems less likely than the advantages outlined above.

Exercise 15.7

What are the major achievements of computer vision? What are the major problems that remain to be solved?

Answer

Computer vision is being used in a number of application areas, where the images are used to provide input to the computer. For example, face recognition in security systems, monitoring airports and public areas for suspicious behaviour, analyzing scans and xrays, particularly to identify objects in packages, analyzing aerial photographs for cartography and military uses. The success of such systems has been limited and most computer vision is still at the research or experimental stage.

There are a number of problems still to overcome. These include coping with variation in images, such as orientation, shading and light changes, dealing with scenes which include partially occluded objects, speed of scene analysis.

Again this exercise provides the student with a focus fro a brief literature review on the subject. A textbook on computer vision or artificial intelligence is a good place to start.

Design Projects

15.1 Icon Design

15.1.1 *Aim and structure*

This is an exercise in design which should be attempted towards the beginning of the course. It fits in well after the introductory part of the book has been covered (Chapters 1-3). The aim is to encourage students to think about some of the problems involved in providing clear, unambiguous interfaces, even to familiar applications. It also helps the student to begin to evaluate designs critically. They should be able to draw on their knowledge of human cognition and of interaction styles but do not need to use any particular modelling or evaluation techniques. These will come later.

The exercise should be attempted in groups. There are two parts to it. First, each group should discuss the problem and try to reach a consensus on their designs. They can choose to draw upon their experience of other systems if they wish, although it should be pointed out that the developers of the new package do not wish to become engaged in law suits over copied designs! They should then sketch their designs as clearly as possible. It is easier if graph paper is available for this part.

Secondly, the group should display their designs and assess them and those of the other groups. This should be done critically but fairly — it is worth pointing out to them that their work is being assessed as well! The students should identify what they believe to be good designs and most importantly the reasons why they are good.

15.1.2 *Instructions to the student*

You are part of a development team charged with producing a new drawing package. You have to develop the full set of icons. This is a new product and so there is no set house style. It is to appear on a new platform and so does not have to follow any set of user interface style guidelines. The platform usually supports colour on a high-resolution bit-mapped monitor, but icons must be suitable for use on black & white as well as colour screens — design in black and white first then add colour. Text can be used if required. Each icon needs an unselected and a selected state.

The following is a list of all the functionality required: you may choose a form other than an icon to provide the function but you should justify your choice.

- freehand draw with pencil
- freehand draw with paintbrush
- smoothed freehand drawing
- draw rectangular box
- draw round-edged box
- draw ellipse
- fill or shade items
- move or resize item
- write text
- draw irregular polygon
- draw straight lines
- draw arc
- scale object
- rotate object
- edit shape

Icons should be designed on 64x64 grids. Try to draw them as clearly as possible and use colour if required.

When you have completed your designs, display them, marking them clearly with their associated functionality. Now look the designs produced by other groups. Try to assess them using the following questions to help you:

- is the meaning of the icon clear?
- is colour used sensibly and consistently?
- will the icon's meaning still be clear in black and white?
- what is the overall effect of the icons when viewed together?

- how original are the designs and is the originality effective or simply novelty for novelty's sake?

Discuss your assessment within your group and with other groups. How does your work compare to others? Make any adjustments you think necessary to your designs.

15.2 VCR Remote Control

15.2.1 *Aim and structure*

This exercise requires the student to design an interface for the remote control for a VCR. This involves first determining the tasks that the remote control must support (these may be a subset of those that the VCR supports) and second determining an appropriate dialogue design.

Students should be encouraged to work in groups and to utilize the modelling techniques for task analysis and dialogue design that they have studied (see Chapters 7 and 8). By the end of the exercise they should be satisfied that their design supports the tasks identified. It is assumed that most students will be familiar with VCRs and the tasks that users wish to peform with them. It may be helpful to show them various examples of VCR remote controls to get them started.

15.2.2 *Instructions to Students*

You are asked to design a remote control for a VCR which will support the most common tasks required by video users effectively. Your first job is to identify these tasks and the subtasks that are required to perform them. Use a task analysis method (Chapter 7) to produce a detailed description of these tasks.

Having identified the key tasks, you now have to design an interface to support them. Use a dialogue notation (Chapter 8) to describe the sequence of actions associated with each task and show how these relate to each other. When you are satisfied with your dialogue, produce a mock-up of the interface showing how the user will perform different tasks. Ensure that your design is consistent with your dialogue description.

15.3 Mobile phone

15.3.1 *Aim and structure*

This is a design exercise in two parts. The first part uses information from users to develop a design for a mobile phone. The second part uses different evaluation techniques to assess the design. There are two aims: to demonstrate how the views of users can be incorporated into design and to demonstrate the different information that can be gained by using different evaluation methods.

As with the previous project, students should be encouraged to use any modelling technique that will help them in their design. Again they should work in groups or pairs.

15.3.2 *Instructions to Students*

You have been asked to design a mobile phone taking into account the user views expressed below, determined from market research studies. There are a number of factors that have to be considered in the design, ranging from the way information is entered into the system and the manner in which is presented back to the user, through aesthetic judgements to the functionality that the system should offer.

15.3.2.1 *Design*

Read the user responses given below and then analyze them under the following headings:

Functionality: You need to determine the tasks which users wish to perform with the phone, and how they want to go about performing those tasks. You need to decide what features your phone should offer and what are dispensible. You are constrained by real-life costs and so will have to trade off size for weight and battery life, cost against functionality, and so on.

Try to identify the real issues that people consider when using a mobile phone. Try to see what factors will influence their decision about which phone to buy. It is these that must motivate your design. Make a list of the features that you feel your phone must have.

Interface: How are you going to present all this information to the user? What style of output is most suitable? What is the best way to interact with it? Will you present all the information at once? What input characteristics will your phone have: how will numbers be entered, functions accessed, and so on?

Having thought about these things, consider that you are also trying to keep the electronics inside the phone as simple as possible. Remember too that people will not carry a manual for their phone around with them, and so they will need to be able to understand what to do, or to be reminded by the interface itself.

Aesthetics: Consider issues of overall colour, size, shape, texture, of button spacing, layout, colour and feedback, display size and placement, information display.

You will use this analysis as the basis of your design. Modelling techniques such as task analysis can help to clarify this which will in turn make the process of design easier.

Having decided upon all the relevant information, now design your phone. You have to produce two views of the phone in all. The first is a picture of the phone as it appears to the user, with no additional labelling - make this as high quality as you can. Imagine it is the sort of picture that would appear in a brochure advertising the phone. The second view is of the same phone, but should be viewed from above. An additional set of pictures of parts of the phone should also be provided that should enable you to run through a typical interaction with a prospective user, showing the display state when they perform certain actions to achieve a certain function.

Either do all of these designs separately on paper, using colour, shading and different viewpoints as appropriate, or utilise OHPs to overlay the basic design. This is particularly useful for the second view of the phone as it allows you to update the display state without requiring redrawing the rest of the phone.

Again modelling techniques can help in the development of an effective dialogue. Consider for example using a dialogue notation to describe the details of your interface or performing a GOMS analysis of the main tasks supported by your design.

15.3.2.2 *Evaluation*

The second part of this exercise concentrates on the evaluation of your design and looks at two different evaluation methods: heuristic evaluation and cooperative evaluation.

The first evaluation method is "heuristic evaluation" (see Section 11.4.2) You will evaluate the design against a set of general usability criteria laid out below:

- system behaviour is predictable
- system behaviour is consistent
- feedback is provided
- the user's memory is not overloaded
- dialogue is task-oriented

These criteria are discussed in more detail in Section 11.4.2. The aim of the heuristic evaluation is to debug the design; to highlight the points in the design that are inconsistent or likely to cause users problems.

Exchange designs with a colleague. Try to evaluate their design using all of their design material and the usability criteria given above. During your evaluation, consider if there are other general criteria that are important considerations for this particular example; if so, include them. You are aiming to see where the design is successful and where it fails; it may assume too much, or too little, it may not offer any feedback, or present an unclear view of what is happenning. The user may be expected to remember too many things at once, or there may be different ways to do similar things. Make notes of your evaluation.

Return the evaluation and your notes on it to the original designer, and repeat the exercise on a different design.

At the end of this stage, you should have done two heuristic evaluations, and have received two independant evaluations of your design. Go through these evaluations, collating similar problems and resolving differences between reports, until you have an accurate summary of the evaluation.

"Cooperative evaluation" is another evaluation method, in which the user is encouraged to see himself as a collaborator in the evaluation rather than just a subject (see Section 11.5.2). In pairs, take it in turns to be an evaluator and a user for each design.

As an evaluator spend a few minutes thinking of some scenarios and tasks for the user to perform. For example, to look up a telephone number, to meter a call, or to dial a number. Include some complex tasks as well as some simple ones. Ask the user to step through these tasks using each design. Take notes on the user's actions, comments and any problems. The fuller the detail in the protocol, the better.

As a user, follow the evaluator's instructions. Try to give as much information as possible. Explain what you are doing at all stages and why. Answer the evaluator's questions as clearly as possible.

Look at the results from the heuristic and cooperative evaluations. Notice where they highlight problems in the design. Also look for instances in which the designallows tasks to be easily accomplished. Compare the two techniques and note anything which is identified by one technique and not the other.

15.3.2.3 *Redesign*

Reappraise your design carefully in the light of the evaluation process. Carefully look at things that cause confusion, at the functions that are there but may never be used, and at the comments made about the good and bad points of your design. Using your experience of others designs and the comments on your own, you have a great deal of information about what should be included, and excluded, from a new design. It may help to work in groups of four or five, for this final stage, though it is up to you. Pool your designs and try to come up with a new version of the phone. It should incorporate all the good features of the previous prototypes, whilst avoiding all the problems that they may have suffered. Once decided upon, each of you should produce a final labelled drawing of the ideal phone.

15.3.3 *Mobile Phone: User Responses*

A group of people were interviewed to obtain their views on the features they would expect to find in their ideal mobile phone. Some of the people currently use a mobile phone, others do not. Given below are the (edited) texts of their responses.

A: "I want to keep track of the number of calls that I make, and the cost of the calls. Since I use the phone for business it has to be small but powerful but can't cost a lot to buy. I don't want to fuss with that stupid airel thing that you have to flip up in order to use it and I want to be able to keep ringing people back if they're busy - as I said it has to be hand-held but that's it really."

B: "Ohhh - a mobile phone! Well, it just has to ring people and let people ring you, doesn't it."

C: "It must be able to store telephone numbers in as I won't have my telephone book with me but I dunno how I'd search for the numbers - perhaps a pad on which you could write them down would be a good idea - no that's daft really isn't it I mean it's supposed to be electronic I suppose what about a volume control for ringing. I remember someone's phone going off in a meeting it was really loud and she was so embarassed .....what about the buttons will it have press buttons or a dial? Oh I can decide well I like the dial personally I think it's aesthetically pleasing but some people like buttons I dunno really. Running them costs a lot of money so I suppose I'd like to be able to tell how much I'd spent that day on calls sort of get my bill as I go along I suppose. Will I be able to turn it off? Oh - I have to decide do I? Well I suppose I should have it with me and on all the time but what if I didn't want to be interrupted so I suppose I'd want to turn it off ..... but then I may miss important calls - can I have an answering machine built in as well or something so that it answers my calls if I'm not there - that would be nice."

D: "I wouldn't use one ever - yuppie toy!"

E: "Well, the buttons would have to feel nice as I hate those plasticy tacky ones yu get on those £8-99 phones. It's difficult to tell when the button has been pressed sometimes. Some sort of display would be good - I suppose it should have information on the status of things - battery maybe and the signal strength, th number I'm calling, the previous numbers so that I can scroll back and get them, and perhaps an address book thingy that has names and addresses and telephone numbers in. Hmmm that's a bit bulky right - I'd want it to be pocket sized to. Oh and the batteries should have to last for ages and ages and I want to be able to charge it up anywhere like in the car and so on."

F: "Oh - I'd never have one because what if I lost it? It's not the cost of the phone that's the problem it's that someone may ring up their friend in Australia and I'd have to foot the bill. It would be okay if only I could use it."

G: "Wow - mobile phones are great - I want one that rings people up, takes messages, will allow me to send faxes and receive them, will act as a calculator as well - I mean there's all those number buttons already right? - and an address book and small word processor - it's got to have a display and things right? so it could also be a personal organiser and interface to PCs and Suns and Macs and things and it must be small; that's for certain, it has to be small and the battery has to last for a week or so without being recharged and handheld - – no I said that - errrm that's it. Oh and I want to use it with the interface bit so as I can read my email on the Suns when I'm away and stuff."

H: "Easy: 1. see the number I'm ringing. 2. be able to redial easily when a number is busy. 3. see the battery strength. 4. see how much it's cost me. 5. have nice buttons to press with good feedback. 6. not have to reenter all a long number if I make a mistake. That's all"

I: "I'd like a system that has an address book in it, but it doesn't have to be too powerful. Then I want to be able to see what number I'm ringing and re-ring it if necessary and see how much the calls have cost. A good aerial will be needed because good reception is really important. A way of barring calls international numbers would be good and a way of barring others from using it would also be nice. I'd want to turn it off occasionally but I dunno what would happen if anyone tried to ring me."

J: "I like them, especially the grey ones - they're better than the black - and the pretty light that comes on that's good too dunno what it's for though."

K: "I don't like them - because I'm a bit deaf they're too quiet for me and I can't hear what people are saying."

L: "Ha! what would be nice is a phone with not too many buttons! They all have too many buttons nowadays and I hate that. Keep it simple, I reckon."

M: "I don't understand them. I never know which button to press to get the phone to ring and which to press to stop a call, and I can never work out how to answer it either - do you just pick it up or what? And what are the letters on the number buttons for? You know – the 1 has 'abc' on it and so on?"

N: "What's a mobile phone - a cordless one? Oh, one of those things. Naw, I don't want one of those - they're like half a housebrick.....oh, I can choose can I - well I just want the same as I've got at home - small, you know, but having it portable. That's all I want."