Closing the Loop: Modelling action, perception and information

Alan Dix

hci@hud – The HCI Research Centre, School of Computing and Mathematics
The University of Huddersfield, Canalside, Huddersfield, HD1 3DH, UK.

tel: +44 1484 472908 email: alan@zeus.hud.ac.uk

ABSTRACT

Visual interfaces to computer systems are interactive. The cycle of visual interaction involves both visual perception and action. This paper examines formal models of interactive systems and cognitive models of users. Neither completely captures the special nature of visual interaction. In order to investigate this, the paper examines two forms of non-visual interaction: mathematics for the blind and interaction by smell (nasal interaction). Finally three forms of more pragmatic design-oriented method are considered: information rich task analysis (<u>what</u> information is required), status–event analysis (<u>when</u> it is perceived) and models of information (<u>how</u> to visually interact with it).

KEYWORDS

Formal methods, cognitive models, aural interfaces, status–event analysis

1 INTRODUCTION

Visual interfaces are clearly very powerful and popular. Is this simply because they look good or are there deeper reasons? Also some visual interfaces are more successful than others, but what makes them so and how can we design visual interfaces which make maximum use of the medium? Without some sort of structured understanding design is an open-loop activity – even where the design process is iterative, each attempt is a shot in the dark.

The crucial difference between a visual interface to a computer system and well designed graphic art (say a poster on bill-board) is that computer interfaces are interactive. We do not just 'see' things on the computer system, but also manipulate them using their visual properties – "what you can see you can grab". Visualisation is about the look of things interaction is about the feel.

The first part of this paper will deal with models which help us to understand the nature of visual interaction. First looking at the formal models which have been the initial basis of much of my own work, and then at various cognitive models. Each model will tackle only part of the interactive process.

Proceedings of AVI'96 – Advanced Visual Interfaces, Gubbio, Italy. pp. 20–28. ACM Press.

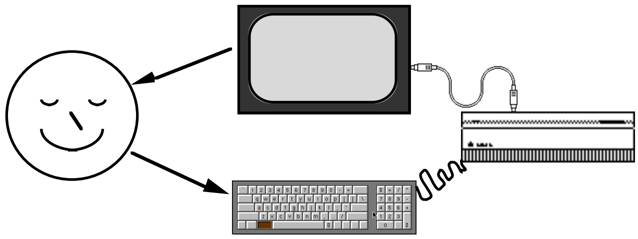

Figure 1. Visual interaction

Interaction is often viewed as a cycle where user goals are translated into user actions, which are processed by the machine to give some (visual) response, leading to a reformulation of goals ... (Norman 1988). Few models even attempt to model this entire cycle of activity, and we will see that both the 'visual' and 'interaction' aspects cause problems for many of them! Note especially that there is no mouse in figure 1. It is not at all clear that this cycle is the best way of looking at mouse-based interaction.

Given the complexity of visual aspects of interaction, the second part of this paper "pushing the boundaries" looks at non-visual interaction. We will try to see what distinguishes vision from other senses. Perhaps the success of modern interfaces is purely to do with their rapid interaction? By breaking beyond the boundaries of normal visual interfaces we can start to see what makes vision different and hence understand how to use it better.

Finally, in section 4, we will return to the issue of design and look at more pragmatic methods which attempt to capture more of the crucial aspects of the cycle of interaction. As with the models considered in the first part they cannot capture everything, but each focuses on some aspect of the holistic nature of visual interaction.

2 MODELS OF PARTS

2.1 Formal models of display–based interfaces

My own work in HCI began with the study of formal models of single user interaction. One of the earliest and simplest models was the PIE model (Dix and Runciman 1985), developed by Colin Runciman and myself back in 1984 and later developed into various specialised models for different aspects of interactive systems (Dix 1991). The philosophy of the PIE model is to describe an interactive system from the viewpoint of the user (but not necessarily in the language of the user!). Following this philosophy, it is a black-box model – only concerning itself with the behaviour of the system as perceived by the user but <u>not</u> the internal representations used in the programming or even design of it.

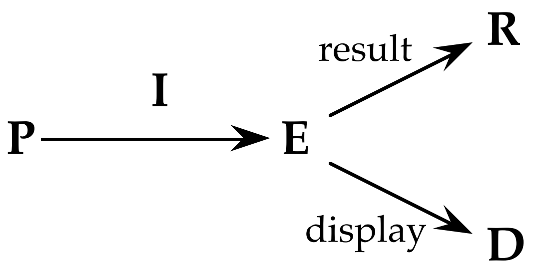

Figure 2. The PIE model

The PIE models the behaviour of the system as a function (I) mapping the trace of user actions (P) to their effect on the system (E).

$$I:\ P \rightarrow E$$

These names perhaps need a bit of explanation. The label 'P' originally was short for 'programs', it is now rather confusing, but a change would entail the model also changing its name! The set of effects, E, can be thought of as the state of the system, but in the special sense of the state as perceivable by the user. Unpacking this rather innocuous statement turns out to be a major task (see Dix 1991).

The effect of the user's actions is typically seen by the user in two ways. First of all immediately in the current display then later in the final results of the interaction, perhaps the printout of a word-processed document. These are represented by the functions 'display' and 'result' and can be thought of as the ephemeral and permanent effects respectively.

$$\begin{array}{ll} \text{display:} & E \rightarrow D \\ \text{result:} & E \rightarrow R \end{array}$$

In fact this particular formulation was driven strongly by the prominence at the time of WYSIWYG – "what you see is what you get". One of our goals was to formalise this idea so that we could go to a particular system and pronounce with certainty whether it truly was WYSIWYG or not!

Looking at the model we have the display: "what you see", and the result: "what you get". The phrase captures the idea that from what you see on the display you can know what you will get in the result. The simplest formulation of this is to demand that there is a function ('predict') from D to R.

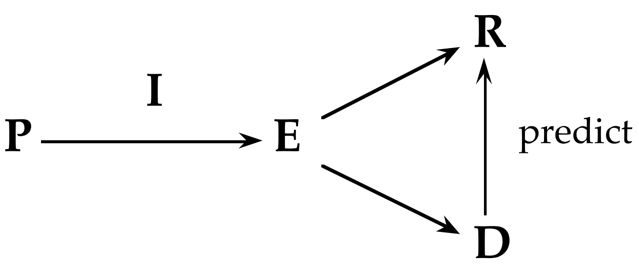

Figure 3. The PIE model

This function must satisfy a simple condition, which says that the above diagram 'commutes':

$$\text{predict} \circ \text{display} = \text{result}$$

This basically says that if you look at the display then use the function 'predict' to work out what will be on the printer, then you will be right!

So far so good, but things are never THAT simple. The above formulation is both too strong and too weak.

First of all it is too strong. It says that you can predict all of the result from the current display. This is unreasonable. At the moment I can see only one page of this paper on my word-processor. A WYSIWYG word-processor by the above definition would only cope with single-page memos! In fact, this is not too serious a problem and there are a series of more complex formulations which capture the fact that what is important is not necessarily what you can see now, but what you could see with a little exploration of the document.

The more serious problem is that it is too weak. The above formulation is based purely on the information available. It would find a system that displayed every character upside down as good as one that displayed them the right way up. In some sense the function 'predict' captures all the user's visual perception, but we have no limitations on the sort of functions which are acceptable. If no such function exists, then the system has fundamental problems, but if there is such a function it may be too complex to use, or even just plain silly.

From a practical point of view, this is not quite as bad as it seems as the sort of problem that the formulation does not cover are ones which are fairly easy to spot. However, it does suggest that some understanding of cognitive processes is also needed.

2.2 Cognitive models

Cognitive models obviously address the left hand side of figure 1 – the human. Different models address different aspects of human behaviour. We can roughly classify the relevant models into:

- psychology of perception
- models of action
- models of processing

These chop up the stages on the human side of the interaction cycle.

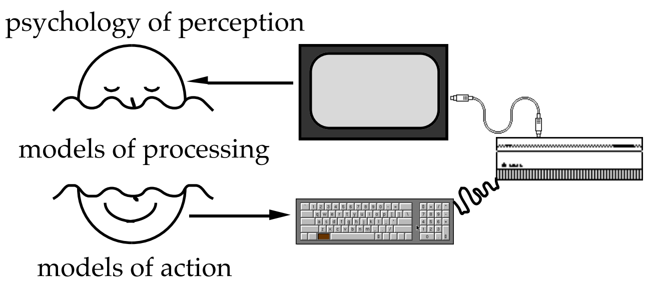

Figure 4. Visual interaction

The study of perception is well established, both within mainstream psychology and applied to interface design (Monk 1985; Dix et al. 1993). At a low level we know about the effects of flicker, the minimum sizes of objects which can be seen at different light intensities, the ability to distinguish colours, including the effects of common types of colour-blindness, the readability of various fonts, point sizes etc. The list goes on. At a higher level we have screen layout heuristics and graphic design, often based on established principles of paper-based graphics (Tufte 1990).

The models of action used in HCI have all been developed primarily for that purpose. For example, extensions to formal grammars have been used extensively to analyse the consistency of actions required to perform related tasks (Reisner 1981; Payne and Green 1986). A related form of action model is based on hierarchical goal decomposition, similar to that used in many forms of task analysis. The most famous example of this approach is GOMS (Card et al. 1983), which has been modified and built upon over the years. All these models tend to be highly non-visual. They start off with a high-level goal, decompose it into a series of sub-goals (methods in GOMS-speak), then decompose those sub-goals further until the level of physical operations is reached (keystrokes, mouse movements etc.). The direction is unequivocally outward from brain to fingers. Sometimes perceptual operators are included, but usually only for timing purposes.

Despite, or more likely because of, their limitations the GOMS family of models have been very successful in predicting timing properties of systems, indeed, predictions based on one such model (John 1990) showed that a proposed system for telephone operators would be slower than the existing system and thus saved NYNEX many millions of dollars.

These action-based models typically assume minimal mental activity. Indeed, as soon as we begin to think, to be really human, we cease to be predictable and models break down. Some techniques do attempt to capture elements of the way we think, for example the user models built into some intelligent tutoring systems. These are most successful when applied to constrained domains such as mathematics or programming. In other words symbolic domains, not visual!

A different kind of processing model which does capture more of the movement between perception and action is ICS (interacting cognitive subsystems) which uses an architectural model of human cognition in order to predict interference between different processing activities within the brain (Barnard 1985; Barnard and Teasdale 1991). As with all models it obtains its power by abstraction, it identifies where visual information is processed (at least in a logical sense), but does not attempt to address the understanding of visual information, or the way in which actions are based on it.

In recent years there has been a minor revolution in the way (some) researchers think about cognition. Traditionally there has been a cerebral fixation – thought happens inside the head., but this is perhaps not the case. Imagine we want to know whether a table would fit in a different part of the room. We might measure it and then do the relevant calculations in our heads. Alternatively we might simply move the table around to see where it fits. In the later case our thinking involves our interaction with the world. Even when we do mathematical calculations we rarely do it all in our heads, instead we use paper, fingers, or an abacus. In a social situation things get more complicated as people solve problems together in interaction with one another. This recognition that thinking happens not just in the head, but in interaction with the environment is called distributed cognition (Hutchins and Klausen 1991), or situated action (Suchman 1987).

In the cognitive modelling community this same spirit, although sometimes explicitly rejected, is also evident in recent work in display-based cognition. Models are being developed which take into account the dependency of human action on what is seen and sensed, for example, Payne's work on adding display based features in a SOAR modelling framework (Payne 1994).

In summary, the available models are either very low-level, giving predictive power at the expense of generality, or purely high-level heuristic knowledge. Although display-based cognition is putting the emphasis onto the whole loop of interaction, the real breakthrough is that the importance of human senses has been recognised, rather than the particular properties of visual interaction.

3 PUSHING THE BOUNDARIES

Recently my daughter was examining an old clock. It was unreliable and about to be thrown out. She wanted to see how it worked, but it was in a sealed case. In order to get inside we had to break the case. In fact, it is often the case that you have to break something to see how it works. Let's see what happens if we break some of the assumptions behind modern visual interfaces.

3.1 "You can use it with your eyes shut": Advanced Aural Interfaces

Some years ago a common phrase heard in user-interface circles was "you can use it with your eyes shut". This was supposed to suggest to you just how easy to use a particular user interface was. Strangely enough the phrase was current at just the time graphical user interfaces were becoming popular and was frequently applied to them. Of course, whereas this could be a phrase to apply to an older command-style interface it seems hardly appropriate for an interface where visual display is central. Perhaps the various specialists were behind the times; or perhaps, as we saw in the last section, they were too enamoured with GOMS-like models of performance.

Although the phrase may bring a wry smile to the modern designer, it is a serious matter if you are visually disabled.

Remember the days when text-based interfaces were the norm, whether command line or full screen character terminals. Everyone had to remember large sets of arcane commands. The information available on-screen was limited and viewing different aspects of data involved complex navigation between screens. Although this may have been complex for the sighted user, it was not substantially <u>more</u> complex for the blind or partially sighted. With a screen-reader and a reasonable memory the blind user could compete with a sighted user. In such ways computers have liberated many disabled people. Indeed, from the other end of an email message who knows how many eyes, legs or fingers the person you are talking to has.

In contrast, the modern interface is based on recognition rather than recall. You are no longer expected to remember commands, simply look them up on the menu. Where keystroke alternatives are supplied[0] designers are far less worried about making them mnemonic and consistent. But that doesn't matter, you only need to remember keystrokes if you are a 'power user'! Displays now have many windows showing different applications and different aspects of the same data simultaneously. Finding out what you want no longer requires complex interaction, just a glance at the appropriate window. Of course, making such heavy use of the visual sense makes things far more difficult for the blind.

Even building screen-readers is substantially more difficult: with a character-based interface the reader could simply ask the operating system what character was at any screen location, with bitmap displays reading text is effectively a form of OCR. happily, screen-readers for Windows are available, but this is only the beginning of the problem. How does one interpret multiple overlapping windows, menus, dialogue boxes, and the use of different typefaces and fonts? There is as yet no definitive answer, although substantial progress has been made over several years on the use of sound: both computer generated speech and non-speech sound (Edwards 1989; Edwards 1993).

The MATHS project is an European TIDE funded project to build a mathematics workstation for the blind. When dealing with text, hearing a paragraph or sentence read to you is acceptable. Mathematics is far more difficult. Even simple formulae are difficult for a human reader to say unambiguously and clearly. Consider:

"the square root of x minus one minus x squared minus 3"

What is squared: the x, "one minus x"? How much of the formula is rooted? The printed formula makes all this clear using spatial layout:

$$\sqrt{x} - (1-x)^2 - 3$$

In fact, good readers use vocal clues to help disambiguate: leaving small pauses, changing the pitch of their voices (prosody) and adding small phrases "all squared" (Stevens and Edwards 1994). On a more complex formula the sighted mathematician will constantly scan and rescan the formula to see what it means. Furthermore, manipulating mathematics means you have to compare different formulae rapidly moving your eyes between them. How can you provide such facilities in a non-visual interface?

Answering such a question forces an understanding of the fundamental nature of sight. Let's look briefly at three factors:

- Gestalt – Looking at the typeset formula, one can just 'see' where the limits of the root and square are. We get a similar feeling when we see a rising graph, or a network diagram.
- Matching – Given an equation, we can rapidly look for occurrences of 'x' to see, for example, if all the 'x's have been collected together on one side of the equation.
- Scanning – When dealing with large formulae, or complex diagrams, we cannot take it all in at once in a gestalt fashion. This is partly a cognitive limitation and partly due to the small angle of sharp vision. When reading text or looking at a picture, our eyes constantly flick from place to place (saccades)

At first the gestalt nature of vision seems most important. However, for complex formulae this breaks down. For the blind mathematician hearing a formula being read, the complexity of formula at which gestalt understanding is possible is perhaps smaller, but for both blind and sighted a more complex interaction with the text is necessary. Consider the formula:

$$12zx + 3x - 4x^2 + y = -4y + 7zx + 3x + 2y^2$$

The sighted reader can quickly look back and forth at the 'x' terms and see that they cancel. The algorithm goes something like:

① look at each term on the left in turn
② look for a matching term on the right of the equation
③ if they cancel remove them both
④ otherwise subtract one from the other
⑤ continue from ② with the next term on the left

We can imagine interfaces to allow such non-visual scanning, However, the difference is one of pace (Dix 1992) – the rate at which one interacts. For the sighted mathematician the pace is governed by the rate at which we can move our eyes and fixate on a new position. The computer interface is limited by the rate at which the user can issue navigation commands, but perhaps more critically by the rate at which parts of the formulae can be spoken.

Notice that I haven't focused on the two dimensional (or 2.5D) nature of vision, compared to inherent linear nature of hearing. This is important within the gestalt of vision, but when we start to scan the visual field, the glances we receive come in a linear fashion. We can easily have 2D aural interfaces with spatial information from stereo effects and up-down left-right navigation commands (perhaps by mouse or joystick). Both are ultimately linear glances through a planar space, the issue is the pace with which we interact with that space.

3.2 Following your nose: Advanced Nasal Interfaces

Although not a dog-lover by nature, I have married into a family of canophiles. Having dogs thrust upon one, it is natural to wonder what sort of perception of the world a dog has. Whereas vision dominates much of our view of the world, dogs also depend strongly on their noses and sense of smell. Some animals, such as moles, depend even more on smell rather than sight.

So what does the world 'look' like through your nose? Dogs spend a few seconds sniffing at each place, so we'll compare it with what we see of the world through our eyes when we spend a few seconds looking around. When we look around we see a snapshot of the world as it is <u>now</u>. We may see a large area, but only how it as at a moment. If we want to know what happened five minutes, an hour or a day ago, we must rely on out memory. In short, sight gives

[0] Which is by no means universal, especially on the Macintosh platform – the bastion of visual interaction!

us a snapshot through space of a single time and we must rely on memory to give is information about other times.

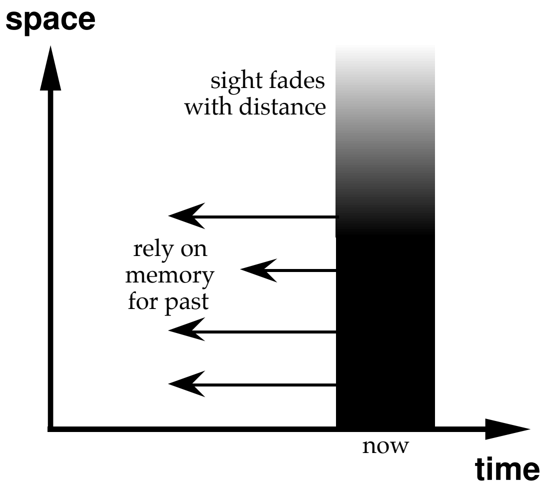

Figure 5. Visual perception

Consider now our dog sniffing. The smell of a place consists of the odours left by all the animals that have visited it over a period. Furthermore, the sensitive nose can determine not only what has been there, but how stale or faded the smells are and hence when the visits were made. That is, the dog perceives a protracted history of a single location. If the dog wants to know about another location it must move and smell there as well. However, in smelling at the new location it is no longer able to smell the first location. It must remember. In short, smell gives us a snapshot through time of a single place and we must rely on memory to give us information about other places.

This sounds rather off the point. Am I suggesting designing interfaces for dogs (canine–computer interaction?), or adding smells to human interfaces? No, the application of this sort of thinking is rather more prosaic! It has been previously noted that one difference between graphical interfaces and command line interfaces is their treatment of history. If you are using a graphical interface, looking at the screen tells you about the state of the system now, but you have to remember how you got there. On the other hand with a command line interface, the transcript gives you a trace of what you have done and what parts of the system were like at various times; however, to find the state of the system now requires exploration. Sounds familiar?

So command-based interfaces are rather like sniffing and GUIs more like seeing. Which is better? Obviously we are well adapted to act and respond in visual environments. We work from the state of the world as we perceive it now, acting to change it in the ways we desire. Many of our basic skills are based on reaction and moment-to-moment assessment of the situation. This is why graphical interfaces are engaging and 'easy to use' – they build on our innate abilities.

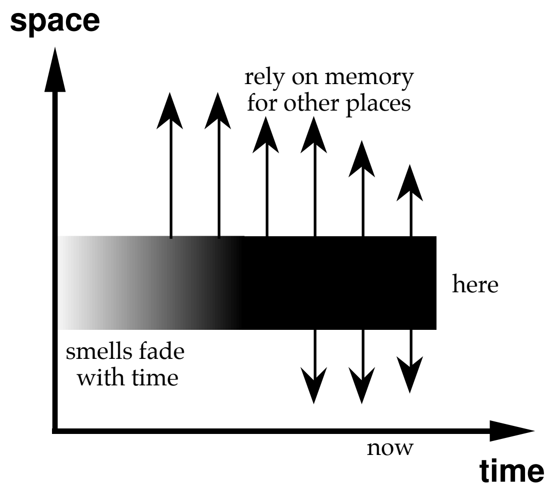

Figure 6. Nasal perception

However, much of culture and civilisation depend on a reaching beyond the here-and-now, an awareness of the past: the reasons for things and the way in which they have come to be. A sense of history is central to all complex human activity, but it is difficult to manage and to 'visualise'. That is why human-kind has had to spend many thousands of years developing the methods for recording and reasoning in this way. An example of this is the difficulty in designing interfaces for undo mechanisms. If these are more than a single step then the user is presented with a <u>textual</u> list of 'commands' that have been executed in the past. Suddenly the user has to view the world in the same way as an old fashioned command line interface. How are the mighty fallen!

If we look wider afield, other animals have different balances of senses. Bats using sonar obtain something close to a snapshot of the world in true three dimensions – they have an accurate indication of distance as compared to our use of clues such as hiding and binocular vision. Probably their sense of what they are 'seeing' is less detailed than vision, but in addition to the a full depth dimension, they can also detect the speed of objects in their environment. We have to estimate speed more crudely using successive views of an object. Is it moving from left to right? Is it getting closer? Again this has practical parallels in the design of environments for scientific visualisation such as the virtual wind tunnel (Bryson and Levit 1992), where we want poor 2.5D human vision to perceive a true 3D vector field.

The examples continue. Consider whales. Their sonar travels over large distances, so the snapshot they receive is neither of a single moment, nor of a single location, but a sort of cone through space time. The further away something is the further into the past they are looking. The time scales here are in the order of no more than tens of seconds, but it is remarkably similar to the experience of astronomers peering outwards towards the birth of the universe.

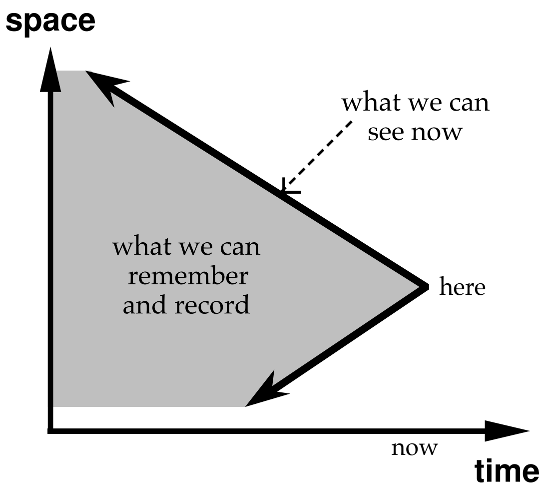

Figure 7. Whales and astronomers

In fact, every sense has limits: smells fade with time, we can only see to the horizon. Each sense differs in its local gestalt, but when we want to build a picture of the wider world: beyond the present, beyond the horizon; the distinction between time and space fades. Our lives and experience are linear and we eventually rely on memory and its aids: paper and electronic records. Larger and larger screens, immersive virtual reality, Hi-Fi quality stereo sounds cannot give us everything at once, the nature of our existence is one of exploration, discovery and recall.

4 CLOSING THE LOOP

Having gained some insight into what makes visual interaction 'visual' let's look at some more pragmatic models which can be used during the design of visual interactive systems. We will consider what information is required at any stage during an interactive task, then look at when information and events will be perceived by the user and finally look at models of how information is presented and interacted with.

4.1 What

The MATHS project needs to determine what navigation mechanism is most appropriate. In order to determine this one needs to know which parts of a mathematical expression the blind user will want to visit during a typical interaction. In other words: what information is required at each stage during the solving of a mathematical expression? There are some existing models of sighted mathematics, most notable by Larkin who is working within remit of the display-based cognition and has constructed lisp models of algebraic manipulation, focusing on the use of the written equation (Larkin 1989). Of course, this was for sighted use and the program assumes all of the equation is visible all of the time. Linehan and McCarthy have used 'Wizard of Oz' techniques in order to obtain a more detailed analysis of information needs during mathematical manipulation (Linehan and McCarthy 1995). This involved getting university students and school children to solve equations written down on a paper that only the experimenter could see. The subjects could ask the experimenter to read parts of the problem and to write down intermediate results, but could neither write on nor read the paper directly.

These techniques are not only of use for the design of aural interfaces for the blind. Their case is merely an extreme, the aural 'window' on the world is very small, but all displays are finite. This is evident when interfaces which functioned well on a desktop machine are shoe-horned into a hand-held computer. However, large the display you can never get everything in it. Sometimes we can get away with being lazy with visual interfaces because there is sufficient screen space, but then only because a mess of stacked windows gives a larger virtual screen.

For virtually any system a detailed analysis of information needs will be useful, if only for frequent scenarios of use. This could take the form of a detailed walk-through or perhaps a hierarchical task analysis (Shepherd 1995) annotated with the information needed to perform each sub-task. This can then be used to assess whether the required information is available in any given design for a visual (or aural) interface.

4.2 When

If the interface is purely driven by the user, then our job as interface designers may be simply to ensure that information is available when it is needed. However, when there is any form of sporadic or externally generated events things are not so simple. We must ensure that users have information presented to them and brought to their attention at a suitable time. This is one of the outcomes of status–event analysis.

Status–event analysis covers a range of techniques that myself and others have worked on over several years. It is based on the simple distinction of phenomena into events – things that happen – and status – things that are. Events we are used to: the ringing of an alarm clock to wake you up in the morning, keystrokes and mouse clicks sent to the computer and various beeps and tones it makes in return. Status is a little more complex and refers to anything whose value you could sample: the time on a watch face, the position of the mouse on the table, the display on the screen. It is interesting to note that many models used in user interface design are predominantly event based (even the PIE model for its input!), yet for visually intensive systems both the major input device, the mouse position, and the means of output, the display, are both status phenomena (Abowd and Dix 1994; Dix et al. 1993). It is typically the case that if you attempt to model a status phenomena with events or vice versa the result is less than ideal!

This distinction, although simple has surprising analytic power. It can be used to describe human–human interaction, human–computer interaction and computer–computer interaction. Most important many of the same phenomena arise in all these different situations. An example of this is polling.

Any change in status can be regarded as an event, for example, when the clock hands pass 10 o'clock. However, agents may not notice this change straightaway, there is typically a lag between the actual event and the perceived event. There is only a small number of ways in which the agent can discover the status change, one of which is polling, periodically examining the status. This is what we do as humans when occasionally glancing at the clock and what computers do internally at various levels.

Furthermore, one of the ways in which agents, human and computer, communicate events to one another is using status. One agent changes the status (actual event) and at some time later the other agent notices the change (perceived event). An example of this is the behaviour during the delivery of electronic mail. Many of us have some sort of mailtool that displays an icon on the screen. When mail arrives it changes the icon to say so. The icon may be an image of a mailbox or an in-tray:

– no mail yet

– mail has arrived, but when?

Has the mail arrived just when the icon changes, or has there been a lag? In fact, this is often the culmination of a complex multi-stage process. Under UNIX it involves two status-mediated events depicted in Figure 8. Let's start with mail arriving at the system (of course, it was initially sent by someone, but that is another story). This is an event. Some level of software agent notices this event (sendmail under UNIX) and adds the mail message to the end of the relevant user's mailbox file. This is a change in the status of the file system. The users mailtool agent periodically monitors this file, perhaps examining it every 30 seconds – a polling behaviour. Next time it looks the file has grown and so it knows that mail has arrived – a perceived event for the mailtool agent. It then changes the icon – a change in status of the screen. The user occasionally glances at the screen (a sporadic polling behaviour) and some time later notices the icon has changed (perceived event for the user). Only at this stage has the mail arrival been successfully notified to the user.

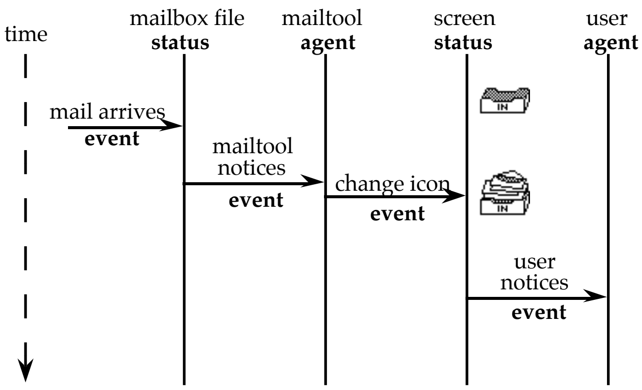

Figure 8. Mail has arrived!

Notice how similar behaviours happen between sendmail and the mailtool (computer–computer interaction) and between the mailtool and the user (human–computer interaction). Also note that a simple answer to the question about the lag between arrival and notification may have missed the human perception stage. Effective design will involve both aspects. Finally, and most important, note how the same analytic framework has been used throughout the stages of human–computer interaction. Rather like the 'predict' function in section 2, we have to rely on some heuristic input to the process to determine how often and reliably the user is likely to poll the screen, but this is a highly focused question. The status–event analysis is bringing formal analysis of the system and informal heuristics of the interface together within a single framework.

This and other aspects of status–event analysis have been used to analyse small scale properties of on-screen buttons (Dix and Brewster 1994), medium scale properties of the timeliness of status information in networked systems (Dix and Abowd 1995) and the progress of long-term office procedures (Dix et al. 1996).

4.3 How

Finally let's look at models of how information is presented.

There are a variety of models of visualisation coming from different communities. For relatively simple data, rules have sometimes been derived for semi-automatic graphical layout. For example:

> if the x data is ordered and both x and y are continuous then use a line plot

Complex scientific data is not so easy! Many techniques have been developed, using colour, three-dimensional effects, flow lines, contours, arrows, and increasingly virtual reality. How does one choose which technique to apply to a particular data-set? One way to at least get some handle on this problem is to find some representation of the dataset and of the visualisation technique so that one can at least tabulate the possibilities. An example of this is the work of Brodie who represents the dataset and the visualisation using the same notation (Brodie 1993). For example a scatter plot is represented as:

dataset: $O^{V_2}(N_1)$
visualisation: $\{O_2\}_0$

This says the data set is a function from a one dimensional set of nominal values 'N_1' to a two dimensional vector 'V_2' of ordinal values 'O'. The visualisation is a zero dimensional restriction (the curley brackets) of a two dimensional ordinal field (the screen).

Although this seems quite complex, it does not even attempt to deal with computer data, such as trees, graphs, hypertext and data-base results! More important it is a model of visualisation, not visual interaction. That is it is primarily interested in the static display of data. Interaction introduces extra dimensions. On the one hand it can add new methods of viewing complex data. This is evident in the use of virtual reality techniques which at very least allow one to grow the normal visualisation space from two to three dimensions. However, the extra temporal dimension can be used in other ways within both two and three-dimensional displays. During the process of preparing an (unsuccessful!) funding bid some years ago, my colleagues and I coined the term temporal fusion to refer to a variety of such techniques which use temporal behaviour to portray information, often under the control of the user. One generic method is sequential temporal fusion.

For example, two maps of the same area showing different information (say political and physical features) the maps are too complex to overlay, but by flicking back and forth between them one can obtain some feeling for common features. In contrast, concurrent temporal fusion occurs when related things happen to two or more simultaneously visible objects. For example, we might see two graphs the trail of an aircraft through space, the other showing its distance from base through time. A point can be moved along the two graphs and allow the user to see the relationship between the two.

Finally, we get on to the most complex case of all, the filtering or modification of possibly complex information via a visual interface. One can apply the same sort of techniques, modelling or taxonomising the information domain and the different sorts of interaction technique.

An example of this is Ahlberg and Truvé's recent classification of interaction widgets (Ahlberg and Truvé 1995). They consider basic data to consist of tuples of integers, reals and strings and looks at different sorts of selectors: individual values, ranges or sets. This allows a mapping of the design space and highlights gaps where no current technique fits. This sort of two-dimensional taxonomy grid is often surprisingly effective as one can take neighbouring techniques and see if they can be modified to fit an empty location – new widgets from old.

In similar work at Huddersfield, we have been considering ways to classify the growing number of browsing and searching techniques for complex data structures. This time rather than the data itself, we have focused on the aspect of the dataset used for retrieval:

- values of attributes – as in traditional relational databases queries
- unstructured keywords – as in many bibliographic searches
- taxonomy or hierarchy – as found in file systems and multi-level library classifications
- direct reference – as found in hypertext links and network or object-oriented databases
- semantic association – measures of closeness between related items

Note how these do not map directly onto the data structure as, for example, database records could be accessed by looking at records with similar attributes – semantic association. On another axis we can put different visualisation techniques:

- spatial – points in space, e.g. star-field displays (Ahlberg and Shneiderman 1994)
- structural – grids, trees and graphs, e.g. cone-trees (Robertson et al. 1991)
- summary – statistical information, e.g. how many items in different classes
- listing – linear listing of items
- individual – only single item or record shown at once

For example, the standard World Wide Web interface uses direct reference navigation and individual item visualisation.

Individual item visualisation is perhaps not very interesting in terms of visualisation, unless each item is itself complex. However, there are suggestions for adding CSCW (computer–supported cooperative work) awareness features to the web so that you can see who has recently accessed pages and hence perhaps make contact with people of similar interests. This may not be heavy on visualisations, but is remarkably similar to the situation we found in section 3.2. So, the culmination of years of international network development, the first global hypertext, and the state of the art developments in CSCW have brought us not to an advanced visual interface, but to the cyber-nose!

5 SUMMARY

Visual interfaces are not just about visualisation, but also about visual interaction. The fact that we can investigate, navigate and modify visualised information gives extra opportunities and problems. Interaction is often viewed as a cycle and we saw that different models only covered parts of the loop. Closing the loop typically requires both technical and human-oriented input into the design process.

The PIE model is able to capture some of the dynamics of interaction and generic information requirements, but the information (in the formal sense) oriented nature of the properties means we need additional human-oriented analysis. Although there are many cognitive models of various aspects of visual interaction, they do not seem particularly better at capturing the full richness of visual interaction, although the growing study of display-based cognition holds great promise.

We looked at what happens when interaction is not visual. Aural interaction shows that the crucial thing about visual displays is that we can rapidly navigate them with eye movement – a form of low-level interaction. Nasal interaction reminds us that in being able to see space, we typically loose a representation of history – both may be necessary in complex systems.

Armed with a fresh perspective on the nature of visual interaction, we looked at pragmatic models which may help in design. Mathematics for the blind forces an analysis of information requirements. However, in visual interaction, especially when screen space is limited, we also need some sort of rich task analysis which tells us what information is required at each point during interaction. The broad descriptive nature of status–event analysis has enabled us to understand aspects of the entire human-computer cycle from both a formal and informal perspective, bringing the two together within a common framework. It has proved especially powerful in analysing when information becomes available and is noticed by the user. Finally, we looked at models of information itself which help us to decide how particular information should be represented and interacted with at the user interface.

REFERENCES

G. Abowd and A. Dix (1994). Integrating status and event phenomena in formal specifications of interactive systems. *SIGSOFT'94*, New Orleans, ACM Press. pp. 44–52.

C. Ahlberg and B. Shneiderman (1994). Visual information seeking: tight coupling of dynamic query filters with starfield displays. *Proceedings of CHI'94*, Boston, ACM Press. pp. 313–317.

C. Ahlberg and S. Truvé (1995). Exploring terra incognita in the design space of query devices. *Engineering for Human–Computer Interaction*, Grand Targee, USA,

P. Barnard (1985). Interacting Cognitive Subsystems: A psycholinguistic approach to short-term memory. *Progress in the Psychology of Language* , Ed. A. Ellis. Hove, Lawrence Erlbaum Associates. pp. 197–258.

P. J. Barnard and J. D. Teasdale (1991). Interacting Cognitive Subsystems: A systematic approach to cognitive-affective interaction and change. *Cognition and Emotion,* **5**: 1–39.

K. Brodie (1993). A classification scheme for scientific visualisation. *Animation and Scientific Visualisation* , Eds. R. A. Earnshaw and D. Watson. London, Academic Press. pp. 125–140.

S. Bryson and C. Levit (1992). The virtual windtunnel: a environment for the exploration of three-dimensional unsteady fluid flows. *Computer Graphics and Applications*.

S. K. Card, T. P. Moran and A. Newall (1983). *The psychology of human computer interaction*. Lawrence Erlbaum.

A. Dix and G. Abowd (1995). Delays and Temporal Incoherence Due to Mediated Status–Status Mappings. *SIGCHI Bullitin,* (June 1995)

A. Dix and S. A. Brewster (1994). Causing Trouble with Buttons. *Ancillary Proceedings of HCI'94*, Glasgow,

A. Dix, J. Finlay, G. Abowd and R. Beale (1993). *Human-Computer Interaction*. Prentice Hall.

A. J. Dix (1991). *Formal Methods for Interactive Systems*. Academic Press.

A. J. Dix (1992). Pace and interaction. *Proceedings of HCI'92: People and Computers VII*, Cambridge University Press. pp. 193-207.

A. J. Dix, D. Ramduny and J. Wilkinson (1996). Long-Term Interaction: Learning the 4 Rs. *CHI'96 Conference Companion*, Vancouver, ACM Press.

A. J. Dix and C. Runciman (1985). Abstract models of interactive systems. *People and Computers: Designing the Interface*, Cambridge University Press. pp. 13-22.

A. D. N. Edwards (1989). Soundtrack: an auditary interface for blind users. *Human–Computer Interaction,* **4**(1): 45–66.

A. D. N. Edwards, Ed. (1993). *Extra-ordinary Human–Computer Interaction*. Cambridge, Cambridge University Press.

E. Hutchins and T. Klausen (1991). Distributed cognition in an airline cockpit. *Cognition in communication at work*. Eds. Y. Engestrm and D. Middleton. CUP:,Cambridge.

B. E. John (1990). Extensions of GOMS Analysis to Expert Performance Requiring Perception of Dynamic Visual and Auditory Information. *Proceedings of CHI'90*, Seattle, ACM Press. pp. 107–115.

J. Larkin (1989). Display based problem solving. *Complex Information Processing: The Impact of Herbert A. Simon* , Eds. D. Klahr and K. Kotovsky. New Jersey, Lawrence Erlbaum Associates.

C. Linehan and L. McCarthy (1995). *A Task Analysis of Students doing Mathematics: Contributing to the design of the input and manipulation language*. MATHS Project Internal Report, IR-15.3, University College Cork.

A. F. Monk, Ed. (1985). *Fundamentals of Human Computer Interaction*. London, Academic Press.

D. A. Norman (1988). *The Psychology of Everyday Things*. Basic Books.

S. J. Payne (1994). Acquisition of display-based skill. *CHI'94 Conference Companion*, Boston, ACM Press. pp. 299–300.

S. J. Payne and T. R. G. Green (1986). Task action grammars: a model of mental representation of task language. *Human–Computer Interaction,* **2**(2): 93–133.

P. Reisner (1981). Formal grammar and human factors design of an interactive graphics system. *IEE Transactions on Software Engineering,* **7**(2): 229–240.

G. G. Robertson, S. K. Card and J. D. Mackinlay (1991). Cone Trees: Animated 3D Visualisation of Hierarchical Information. *Proceedings of CHI'91 Conference of Human Factors in Computing Systems*, ACM Press. pp. 184–194.

A. Shepherd (1995). Task analysis as a framework for examining HCI tasks. *Perspectives on HCI: Diverse Approaches* , Eds. A. Monk and N. Gilbert. London, Academic Press. pp. 145–174.

R. Stevens and A. Edwards (1994). *Analysis of audio approaches*. MATHS Project Internal Report, IR-6, University of York.

L. A. Suchman (1987). *Plans and Situated Actions*. Cambridge, Cambridge University Press.

E. R. Tufte (1990). *Envisioning Information.* Cheshire, CT, Graphics Press.