User Interface Design as Systems Design

Harold Thimbleby,[†] Ann Blandford,[†]
Paul Cairns,[†] Paul Curzon[‡] & Matt Jones[⋆]

[†]*UCLIC, University College London Interaction Centre,*
26, Bedford Way, LONDON, WC1H 0AP, UK
[‡]*Interaction Design Centre, Middlesex University,*
LONDON, N14 4YZ, UK
[⋆]*Department of Computer Science, University of Waikato,*
Private Bag 3105, Hamilton, New Zealand

EMail: *h.thimbleby/a.blandford/p.cairns@ucl.ac.uk,*
p.curzon@mdx.ac.uk, always@acm.org

When designing complex systems, it is standard systems engineering practice to carefully design the interfaces between subsystems. Yet when designing human/computer systems, the interface between human and system is not usually thought through in such terms. Instead, the human is often given wide access to arbitrary parts of the system, and the result is a complex human/computer system that fails in various ways.

We illustrate this argument with a case study of a public walk-up-and-use rail ticketing system. We show that the interaction imposed on the user is inappropriate to the user's task needs; we show how user interface problems arise through access to organisational conventions that are of little interest to users. Furthermore, the wide interface is beyond the resources of the rail organisation to manage.

Conversely we show that an interface designed to hide irrelevant complexity (exactly as one would do approaching user interfaces as a systems engineering design problem) can have a beneficial impact on the user experience, including improving the reliability of the total system.

Keywords: Design, graphical user interfaces, interaction design, personal technologies, public user interfaces, system complexity, systems engineering, walk-up-and-use.

1 Introduction

Standard good HCI practice is to find out about user tasks as one of the first steps in design, otherwise it is tempting to start design from the functionality provided by (or planned to be provided by) the system. The system functions may interact in interesting and complex ways, but the HCI issue is to support the user task, rather than give access to the underlying system *per se*. One of the most successful user interfaces to general purpose systems is the graphical user interface, and its success in providing access to PC functionality means it is frequently a candidate for use in other applications, which are supposed to support much simpler tasks; the same is true of web-style user interfaces, which provide such easy access to the world they 'must' provide easy access to simpler systems! Unfortunately neither a GUI nor a web style interface constrains the designer much, and does not motivate finding a clear view of the task, because it is possible to support almost any functionality and hence almost any task.

This paper argues that proper design, based on well-known systems engineering principles, can transform the user experience by hiding irrelevant complexity. We support our abstract discussion with a case study of rail ticket vending machines as an example of complex public technology, but the principles underpinning the discussion apply very widely across a range of technologies and contexts of use. We show how appropriate design can eliminate conventional user interface problems for the task in hand, including problems of implementation quality. We will see, further, that having eliminated conventional problems, latent issues become salient: user acceptance now depends on solving less technical but more social problems.

1.1 The Abstract Systems Design View

Many goals cannot be achieved by simple systems. Most interesting systems involve people interacting with each other and often with complex computer systems, and all within some bigger system that is intended to achieve goals such as "staying in business, whilst adhering to regulatory and legal requirements." It is well known that interacting subsystems can exhibit complex behaviour that no subsystem alone may be able to exhibit, such as deadlock, livelock, non-determinism and feature-interaction (multiple features combining in unexpected and usually unhelpful ways). In the specific case of human-system interaction, there are particular problems such as closure errors (including post- and pre-completion errors), which are psychological representations of the abstract system concepts like deadlock.

To avoid such undesirable problems arising in new systems it is crucial that the interfaces between subsystems are well-defined and understood and, moreover, simple. An interface specifies how subsystems interact with each other, and a good interface deliberately hides details that are irrelevant. Once interfaces are well-defined, it is possible to change subsystems (for instance, to make them more efficient) without detriment to the overall system behaviour, provided the interfaces are maintained. In fortuitous cases, of course, one may improve systems by changing interfaces, but in practice it has been found that carefully managing the separation of concerns of different subsystems through well-defined interfaces is crucial to successful design. More importantly, when interfaces are well defined, it is possible

to reason clearly about a subsystem without having to know everything about all other subsystems: because their details are hidden, their details are irrelevant. Conversely, many system failures can be attributed to circumventions (be they accidental, overlooked or in hindsight unwisely exploited), of what were supposed to be well-defined interfaces. These systemic issues are no less true of human-system or human-computer interaction than of conventional technical systems engineering.

The value of good interfaces between subsystems has been known for a long time (Parnas, 1972). However this knowledge has not been extended to user interface design; for example, a comprehensive textbook on software engineering (Sommerville, 2001) includes a thorough discussion of interface design but the chapter on user interface design does not take a systems approach, being entirely conventional user-centred design.

Typically the interface between the human and the technical system — the *user* interface — is deeply entangled with the internal details of the system. *This is exactly the wrong way to do it.* User interfaces are often designed to control as many features of the system as possible, so this entanglement is inevitable. Indeed there is a line of thought within HCI (Norman, 1986) that one central role for the user interface is to communicate the underlying system model to the user. Interfaces are also deeply entangled with the social, perceptual and cognitive details of the human: hence the lively debates about design, colour, screen layout, culture, and so forth. These issues have a significant impact on the success of the combined human-system. It is a truism that there are no easy design solutions, and one often has to build prototypes and iterate design to identify feasible improvements.

In human computer systems, there are usually many parallel interfaces (in the systems sense). For example, if interaction using the primary interface deadlocks, there may be interfaces available that support 'reset,' 'cancel' and 'timeout': and these operations may be initiated by other subsystems in the computer or in the user (or even bystanders helping the user). Users, of course, usually reflect on their behaviour and can walk away or take other action. In some contexts, avionics being a case in point, with hard real time constraints, the design of the control interfaces is very difficult if they are not to create, rather than solve, further problems of interaction. There is a great deal of interest in designing systems so that users' awareness of interaction (and likely consequences) does not lead to panic or other extreme actions outside the specification the primary interfaces.

Technology is changing, and ideas about the best ways of designing user interfaces are also changing. For many tasks, GUIs are easier to use than command lines, and for a while they have been the best way of interacting with general purpose PCs. We have excellent visual computing skills, and GUIs exploit these skills (e.g., recognising overlapping windows (Friedhoff & Peercy, 2000)) to raise the level of complexity that is easy to interact with. However, the recognised strengths of GUIs do not mean they are best in general. Indeed, because GUIs are so good, they can give the surface impression of providing good solutions while actually forcing the interaction to an inappropriate level.

2 Case Study

Many cities world-wide depend on public walk up and use ticket vending machines for the smooth running of their transportation infrastructure. As a case study, we take a ticket machine installed in early 2001. The machine is a realistic representation of the state of the art in walk up and use technology[†] and is used by the public to undertake real and important daily tasks. Ticket machines are a serious application to study: not only do they raise revenue for operators, but difficulties with them can mean users miss trains and have their lives disrupted, which has a further impact on others. There is also a "problem amplification": not only do individual users have problems, but when they do, they delay people waiting in the queue behind them. A point a rail employee made to us was that passengers have to learn how to use the ticket machine while they are trying desperately to *use* it to get a ticket before their train comes.

The ticket machine considered here has numerous user interface problems; thirty years of relevant usability research and practice has had very little obvious impact on its design, even though incorrect operation of the machine can result in the user being unable to travel or being fined for travelling with incorrect tickets. An alternative view is that the designers are concerned with usability, but that they are only empowered to address surface-level usability concerns (such a colour and button layout) rather than the complex usability concerns that might be regarded as systems engineering. The design well illustrates the endemic poor usability in the current every-day computing infrastructure, and the lack of impact of deep usability concerns on designers, manufacturers and those charged with installing new systems.

Although some work has been done on analysing the interactive problems of public technology (e.g., Stanton & Baber (1997)), much work involves attempts at adding sophisticated interactive capabilities to ticket machines and kiosks. This work has included adding emotional avatars (Christian & Avery, 1998) and new input techniques (e.g., for sign language interaction). Our analysis suggests this approach, at least if not guided by more systems-level thinking, is misguided. Our case study shows very clearly how organisational issues cause bad user interface experiences that are not hidden from the user in conventional approaches.

2.1 Methodological Issues

The context of work studied here is enormously complex, and there is no established methodology for investigating issues such as the interaction between usability, organisational and regulatory conditions. There are also ethical challenges: if the work were commissioned by the interested parties then the independence of researchers would be compromised; conversely, those parties should have a right to answer any criticisms of their systems. We have tried to engage in discussion with the main interested parties (the manufacturers and the rail operators).

The ticket machine manufacturer has been very co-operative with us, though they work under constraints. They provide hardware to their customer's requirements. In turn, much of the hardware they use is constrained by external

[†]There is a difference between "state of the art" relative to the potential of current technology and "state of the art" *in practice* as actually used on the British national rail system. We mean the latter here.

issues, such as financial regulation beyond their control. For example, there may be a better way of users interacting with the credit card facilities, yet this is supplied as a sealed unit for security reasons, and it cannot be changed. The rail operators have declined to comment.

New evaluation techniques for difficult-to-study contexts and applications are needed (Cheverst et al., 2000). Some have further argued that the definition of 'usability' needs to be extended for these new contexts to include notions of intimacy, beauty and social acceptance (Thomas & Thimbleby, 2002). The work reported here is based partly on the experiences of the authors and exploratory investigation of the system interface, partly on periods of observation of others using the ticket machines in question, and partly on discussions with various stakeholders: users, railway staff and the hardware manufacturers. All examples discussed in the body of this paper have been witnessed by the authors or have been experienced during everyday use by at least one of them; the scenarios are not contrived.

A JavaScript simulation has been made of the ticket machine user interface that is adequate to explore the issues raised in this paper, and from which most of our claims can be double-checked. (The simulation obviously doesn't provide tickets or change.) It can be used in a browser over the web (e.g., for evaluation) and the complete source code is available. See `http://www.uclic.ucl.ac.uk/projects/tvm` for the simulation, which for copyright reasons may be used for non-commercial research purposes only.

2.2 A Brief History of Passenger Transport Systems

Before explaining the case study further, it is helpful to briefly review the organisational history in which the user's task is situated. In the next section, we shall see how many user interface problems arise because the interface gives the user access to the intricacies of the broader ticketing system implementation; this provides the background to show, as we do later, how the interface can be narrowed to conceal this irrelevant detail.

To stay in business, transport systems must collect revenue from travellers (or from other sources, such as subsidies) and, in turn, must provide an acceptable service to those travellers to ensure repeat business. Given human nature and the unreliability of transport systems, relying on trust is insufficient: some passengers do not pay, and some modes of transport fail. Thus tickets were invented as a reliable token of the contract between passengers and transport operators. In Victorian times, railway transport became very sophisticated, and the range and type of tickets expanded. Tickets were printed on card by mechanical devices; as more uses for tickets became apparent (e.g., for distinguishing between contracts with different railway companies) different colours of tickets were introduced. And so on.

Today, over a century later, many interacting layers of complexity have been accreted. Rail companies introduced many sorts of ticket fares, depending on modes of travel, and numerous loyalty schemes — involving identity cards in addition to conventional tickets — such as discounts for family travel at certain times of day. Passengers may use credit cards to pay for tickets: these transactions are heavily regulated. In the United Kingdom another layer of complexity was introduced by the privatisation of the railway network, resulting in many competing organisations

and authorities with overlapping interests.

The British railway ticketing system can seem impenetrable to passengers: there are many options and restrictions on tickets and forms of travel. In our terms, the complexity that passengers see arises because the interface between passengers, as one component of the transport system, and the railway companies, as another component of the system, has been defined so that passengers "see" and interact with the internal complexity of the transport infrastructure. Almost all of that is irrelevant to the needs of the passenger, namely to travel economically and in good time.

Rail companies provide interfaces between passengers and the complex ticketing systems: they provide human ticket clerks. The human clerks are supposed to provide optimal ticketing (and other advice) for passenger travel requests. Unfortunately ticket clerks are an expense and managing their costs, in turn, encourages further complexity in ticketing. If clerks cannot always be present, then new forms of tickets have to be introduced (e.g., "permits to travel"). Stations now have ticket clerks, automatic ticket machines, *and* permit to travel machines (and on major stations, like Kings Cross, there are several sorts of ticket machines to cater for different rail operators)!

There is a straightforward business case to automate the ticketing process, and there are substantial potential gains. Passengers will be pleased to have shorter ticketing queues, for instance, and rail companies will be pleased to better manage the demands on ticket clerks in order to save costs, and perhaps to improve ticket clerks' morale.

All this, the history and business case, is well known, though the systems complexity might not be widely appreciated as a highly relevant factor of user interface design.

2.3 The Conventional Solution

A conventional user interface design solution to rail ticketing would be to automate as much as possible of the ticket clerk's role, and automatic ticket machines have been widely used since the 1980s. Ticket machines are intended to replace the clerk for a given fraction (e.g., 80%) of passenger ticketing needs. As might be expected from our introductory comments, automating the existing interface between passenger and complex rail system cannot avoid the complexities apparent in the original interface itself, unless the specification of that interface is simplified. So, not only do we find the expected deadlock and other symptoms of bad design in our case study, we also find that the new automated system is itself so complex as to introduce its own quality control problems. It is incomplete and unreliable.

The particular ticket machine discussed here was installed on Welwyn North station (north of London's King's Cross station) in early 2001. Similar models are installed throughout the UK. The machine has a rectangular purple label saying it is "Easy to use."

ASCOM, the hardware manufacturer, has installed over 30,000 ticket vending machines around the world (`www.ascom.com`, Sept. 2001), with half still in use, handling over 40 million transactions a day. With this scale of operation any usability problems will have a huge aggregated social impact. In Britain, the 1000+ ASCOM

machines have an annual revenue well in excess of £100 million: bad user interface design can be costly; conversely, incremental improvements to user interface design has worthwhile payoffs. If users make mistakes — even if in any one location particular types of mistake are experienced infrequently — the overall cost to rail operators will be huge. On individual stations the usability of a ticket machine has an impact on both traveller and rail staff satisfaction. There is also the knock-on loss of reputation when travellers see other users having difficulties.

2.4 The Ticket Machine

We now explore the design of the current ticket machine user interface: we show how user interface problems are entwined with system complexity.

Figure 1 shows the ticket machine kiosk discussed in this paper. The main buttons are on each side of the screen, and there are two permanently-labelled buttons: one for information, towards the bottom left of the screen, blue with a white italic *i*; and one for cancelling (towards the bottom right of the screen, in red, and labelled explicitly as CANCEL. The *i* button provides help, yet there is no explanation of this (perhaps on the principle that users should recognise *i* as the international standard sign meaning 'information').

The interface employs a standard desktop (GUI/web) paradigm, but is not, in fact, one. The on-screen labels look like buttons, but do not act like them. Likely surface damage would make a touch screen unreliable (so buttons would probably still be needed). Many passengers we spoke to said they thought the screen was touch sensitive, and they found out the hard way that it has buttons off to each side. One user was observed to repeatedly press the screen; many press the screen for the final "purchase ticket" step. The designers have applied a widely recognised interface paradigm in an inappropriate context (possibly because the early prototypes they worked on *were* GUIs on their own computers, without a representation of the final context of use).

The actual buttons are placed some distance from the screen (see Figure 1) because physically reinforcing the machine against vandalism (e.g., on unattended stations) is a major design factor: a steel frame means buttons cannot be closer. It is likely that if the screen information looked less like buttons, and were carefully aligned with the markers linking the buttons to the screen area, the mistake of thinking the screen was touch sensitive (as well as the obvious alignment mistakes) would be less likely. For instance, the rectangular button images could be changed to arrows pointing towards the buttons users must press, in forms like Purchase Ticket ⋙ and possibly animated. Whether this would be an effective idea could easily be checked by experiment, but the point is that the interface needs to be designed for the actual context of use, with all the ensuing constraints.

The top-level screen, which appears either if a user presses CANCEL or after a short time-out, provides some common destinations, such as King's Cross (a major rail terminus in central London).

2.5 Examples of Use

We now give a list of design and usability problems encountered with the ticket machine. The point is to illustrate how the interaction takes place at the wrong level

Figure 1: The ticket vending machine. The screen (diagonal 25cm, underneath the '↓ 1' arrow) is at eye level for a standing adult, its centre 1.47m off the ground. There are five hard buttons on each side of the display screen (looking like triangular arrows) with meanings as displayed on the screen. There are two buttons with fixed meanings, for information (blue, on the left) and for cancelling (red, on the right). There are ten unlabelled buttons beneath the display that do nothing. The user can insert coins, credit cards or paper currency (underneath the '↓ 2' arrow). A slot at the lower right, that looks like it might accept proprietary tokens, is apparently used as an alternative slot to return rejected notes. Tickets and change are obtained at the bottom of the machine, under the '↓ 3' arrow.

for the user, leading to many unnecessary difficulties and breakdowns, and also to set the scene for discussing the potential advantages of solutions based on designing *system* interfaces more appropriately.

2.5.1 Buying a Ticket: Mismatch of Task Structures

First, we present the complex set of steps a user has to perform to achieve the most important goal: buying a ticket.

One of the authors has a routine task of buying a return train ticket for his daily work commute from Welwyn North station (in the countryside) to New Southgate station (in London suburbia). Because he is a regular traveller, he has a Network Card (a sort of loyalty card), which allows certain discounts on certain journeys.

New Southgate is not a destination mentioned on the top level screen of the ticket machine, so the user will select Other Destinations and select through the alphabet (through several screens) until New Southgate appears. If the user makes no mistakes, this takes 10 button presses.

None of the choices shown at this stage of the interaction mentions a network

card discount, and one button is unused — so it would appear that all choices the machine supports must have been catered for. If the user selects the choice More Fares, they get no network card choice in the next screen either.

The first time the machine was used, it became necessary to ask for help from a rail employee, who asserted that there was a network card option. With help from the official, we persevered, and discovered that from the screen with the spare button, you could choose the Cheap Day Return option and get more options, *including* discounts. This is an example of unnecessary complexity for the user: discounts are not an after-thought, but a central part of the task of getting the most economical ticket for the journey.

However on the screen providing network card discounts, there is now no Previous Screen option, so the user cannot change any other part of their choice. The hard key called CANCEL remains, of course, but using it at this stage takes the user back to the initial screen. The key could be better called RESET or START AGAIN, as it resets the entire interaction rather than cancelling the last step (which the soft key Previous Screen does on many screens).

If the user selects Railcard discounts[‡] from here, the next screen confirms that they have selected a ticket to New Southgate, but does not confirm what sort of ticket it is (is it a single, season, a return, or what?). There are two network card choices, neither of which match our user's network card. The user selects one, and is now ready to pay. The screen does not confirm what sort of ticket the user is getting, and there is no choice to return to a previous screen to revise (or confirm) any previous choices.

The user can now put in cash or cards and get the ticket. However, most of the screen stays steady as the cash is inserted, and no special feedback is given to reassure the user that cash is being entered successfully. (We have watched people insert notes, and jump around frustrated, since the machine appears to have eaten their cash but does not appear to be doing anything helpful.)

These usability difficulties indicate a lack of understanding of even the simplest user tasks (as perceived by users), or of the importance of context-relevant feedback on the state of the machine. The overall task structure reflects the complexity of the organisational task model, rather than the relative simplicity of the user's.

2.5.2 Inadequate Help: Inadequate System Testing

If the help button *i* is pressed when a user is trying to find network card discounts, screen displays nonsense:

```
Help not implemented for ContextID 0011.
```

Eighteen months later, we rechecked, and found *another* screen with no help implemented for ContextID0047! It is difficult to understand why help has not been implemented properly. As well as being a comment about bad user interface design (including lack of testing), this error is a sad comment about the quality of software development, and poor maintenance (this problem has been present, despite user

[‡]Note the inconsistent capitalisation.

interface upgrades, for over 18 months, and has survived the introduction of multi-lingual internationalisation — a process which one might have supposed would have reviewed all system texts).

The following text used to be shown when the [*i*] button was pressed when the machine was at its top level:

```
Using This Machine
1. Select your destination.
2  Select the ticket type required.
3. Select Railcard discount if applicable.
4. Pay by card or cash.
```

The interaction designers seem to think that ticket type and railcard discount are different things — a point we return to below. Note also that this screen is only accessible from the top level — at a point where the user has not yet become aware that there is a likely confusion over ticket types and discounts. As is widely recognised, users only read the manual as a last resort, and are therefore unlikely to read this information *before* starting (this information is not available later in the interaction).

Recently, the help system has been extended: the top level screen is now a choice of four languages, which then go down to a multiple choice screen, which provides details of railway regulations and other details. The help screen shown above is now three button presses down — and the simple typos are still there. Someone who needs this help will need help finding it.

In each of the four languages (English, French, German, Dutch) available, one option, "`Collecting pre-ordered tickets`" is in English in every case. Unbelievably, its help screen (in all four languages) is `Help not implemented for ContextID HM08.`

We are concerned that regulations have been quietly inserted (for instance they discuss how to obtain ticket refunds for incorrect tickets issued by the machine) without telling any user. Whether the rail operators would dismiss usability problems by citing the disclaimers hidden in the user interface remains to be seen; a £5 'administrative charge' can be levied in any case — a use of regulations that permits profit from bad design.

2.5.3 Sometimes It Does Not Work

On several occasions, a few seconds after the ticket machine shows "insert payment" (right at the end of a 10-or-more key press interaction to buy a ticket) we have found it unable to actually issue a ticket. It just says "Your cash is being returned," when no cash has been put in the machine which can be returned! One reason for this problem is that tickets can be printed on two stocks of paper. Suppose the ticket machine has run out of normal ticket paper. The user interface is designed so that type of ticket is one of the last choices a user makes, so it is only at this stage that the machine knows whether it can print the requested ticket. If this is the case, the interaction sequence could be re-designed so that choice of ticket type is made earlier in the interaction, or the machine could display on the initial screen (just as it does when there is no

change) that it has restrictions. Why not simplify the ticket regulations — which should be part of the system design. How ever one looks at it, this is a poor feature of user interaction design.

Similarly the machine sometimes will only accept credit cards (perhaps due to running out of change or the cash box being full) or will only accept coins. The user is only told this at the end of an interaction, having already spent time selecting the ticket to purchase. Worse, at the appropriate stage of the interaction, the screen briefly indicates that all payment methods are available, before removing those options that are not available. The delay in correcting the screen is long enough for a user to read it and start inserting coins or cards, no longer looking at the screen.

Sometimes the machine has limited change. It will allow a user to try and buy a ticket, and if they overpay it will return their change. Since the rail operator already has ticket credit notes, why does it not allow a user (who may well be in a rush to catch a train) at least the option of having a credit note printed?

Sometimes the machine will allow a user to get to the penultimate screen, then it displays a screen saying the fares are not available.

In all cases, the user interface is unimaginative, and not based on a clear view of the user's task and the real time pressure users work under. The user is led down garden paths, perhaps as long as 20 button presses, and is then frustrated.

2.5.4 Multiple Ticket Problems: The User as Enemy

Suppose the user is part of a family group and wants to buy a ticket for themselves and the party travelling with them. It takes about 13 key presses to buy one ticket. Unfortunately the ticket machine then reverts to the top level screen, so it is not possible to say, as it were, "same again." Parents travelling with children cannot easily buy cheaper tickets for them. Each and every individual in a party to the same destination must repeat the whole process again. This will take a considerable time — and will provide more opportunities for errors, which in turn will increase the time required to complete their tasks.

The machine would have been easier to use with an Another one button ("repeat last purchase," "again"... the label needs some evaluation to be sufficiently unambiguous), properly integrated with the (existing, but not fully working) Previous Screen button, so that further tickets could be bought easily, just allowing the user to select whether the new ticket is for an adult or a child.

We understand that features like this, which make it easier for the user, also make fraud easier for thieves. Suppose a stolen credit card is being used: the Another one button would permit rapid multiple ticket purchases. From this point of view, it is important to slow down purchases! Losses from just one organised gang amounted to about £25,000 per week, so limiting fraud is a serious design concern.

Changing the ticket machine's user interaction for multiple purchases also impacts on wider issues. The ticket machine must be approved by the Rail Settlement Plan authorities (e.g., regulating how monies are divided between the numerous train operators, for instance as happens with tickets that cover surface and underground travel on different companies). New styles of dispensing multiple tickets is not just a local user interface issue; it could mean seeking new approval; this is a slow and expensive process that has to be offset against the benefits to users (and the increased

revenue it may or may not generate).

As in other domains (such as computer security (Adams & Sasse, 1999)), the user appears to be viewed as 'the enemy': the task is made difficult for the many to protect the organisation against the fraudulent behaviour of a few. This entrenched attitude is further exacerbated by the complexity of the change process. There has been no strategic usability thinking.

2.5.5 Costly Errors and Costly Tickets: Misfits with User Goals

Unlike interfacing to a human operator, mistakes cannot be rectified once a ticket is purchased.

Although the ticket machine "knows" how ticket prices vary with the time of day, it never tells users that if they wait a few minutes, or if they travel on the next-but-one train, they can buy cheaper tickets — routine advice that human ticket offices give. (Travel before 10am is more expensive because workers have no choice but to travel at those times; tickets are cheaper later in the day when the rail company wants to attract discretionary travel.) We have found that the machine's clock is sometimes wrong (it has been up to 8 minutes slow), and it will therefore sell over-priced tickets. It is tedious for passengers that, although the last train before 10am has left and therefore it is impossible to travel before 10, the ticket machine will happily sell over-priced tickets until it thinks it is after 10.

Some cheaper tickets cannot be bought at all. The ticket machine only sells tickets from the local station. For example, if the user starts travelling before 10am, buying a ticket to an intermediate station and a continuation cheaper ticket from there (for after 10am) to the destination is something human ticket offices provide (saving about £5 on a return trip to London), but cannot be achieved from the machine.

Sometimes a traveller will want to travel as far as they can without exceeding the cash they have available (e.g., to get as close to home as possible when they don't have enough cash). The ticket machine forces a user to choose their destination first. If the destination costs more than the user has, they then have to try other sorts of tickets or other, closer, destinations. This is extremely hard to do with a ticket machine as opposed to a human ticket office (especially as destination stations are organised roughly alphabetically rather than in order along the route).

It might be better if a user could put in as much cash as they can afford for the desired destination, and then ask the machine for cheaper options (e.g., single tickets rather than returns, or for destinations one or two stations short).

The ticket machine knows the time, and gives the user cheaper ticket options when they are apparently permitted. Unfortunately if a train is running late, a user may have bought a ticket at the cheaper rate which is not permitted on the train actually used. The passenger is then subject to a supplement or penalty fare, as well as embarrassment.

In all these cases, we see a particular organisational orientation towards the user's task: the task is framed only as 'buy a ticket from this station using the next train to a specified destination station,' without any recognition that users often frame their tasks in other terms, such as 'travel from here to there within a reasonable time at low cost,' or 'travel as far as possible in that direction for the available cash.' Although simple observational studies in a ticket office or passenger interviews

would establish the range of common user tasks, this might be difficult to integrate easily with the current organisational view, so the user is forced to adapt to the organisational perspective — to work at the wrong level.

2.5.6 Post-completion Errors

Post-completion errors occur when a user finishes their task before the interaction is complete (Byrne & Bovair, 1997). The classic example occurs in some designs of cash machines (ATMs), where a user, wanting to get cash, inserts their card, types their PIN, collects their cash, and walks away. They have finished but the machine is left holding the card. A simple redesign, widely known, does not give the user their cash until they have retrieved their card: this ensures completing the task completes the interaction.

The ticket machine suffers the same problem. On one occasion we had to wait for a woman to get her ticket, and she walked off with it. We asked her to come back and collect her change, which had dropped into the ticket slot the moment after she had turned her back on the machine. This is a standard post-completion error.

We interviewed the woman who had bought the ticket. She blamed herself for leaving her change behind. However, her task was to buy a ticket; the ticket machine gave her a ticket, so she picked up the ticket and turned away. If the train she had wanted was already at the station, she would immediately have been running towards it and not even had a chance to hear the change clunk over the noise of the train arriving. She had completed the task she had set out to do (buy a ticket) but the user interface had not finished its part of the task.

The ticket machine hardware can drop change before, or at the same time as, the ticket itself; the actual order used is a software design decision. We understand that one reason for this decision is that return tickets are printed on two separate tickets (the "out" and "return" parts), and the designers did not want users to leave part of the ticket behind. Leaving change till last, they must have reasoned, sometimes — when there is change! — reduces the risk of users leaving without one half of their ticket. Again, trying to improve the user experience at the machine level has implications for the wider design of the entire process: why are return tickets printed on two pieces of paper?

The problem is that the ticket machine is providing an interface to the old system. When human ticket clerks issued tickets, the advantages to the rail operator of multiple tickets might have been worthwhile compared to the trivial problems of issuing them. When the user interface of the ticket machine merely copies this procedure, it exposes the user to intricacies that are now inappropriate and indeed cause predictable errors.

2.5.7 Limited Awareness: Pre-completion Errors

The converse of a post-completion error might be called a pre-completion error: the interface obstructs the user completing the task.

The ticket machine sometimes displays the soft button Purchase Ticket, but when it does, it is not possible to pay. In fact, the user is supposed to press a button at this point — because the ticket machine is otherwise "unaware" that they are trying to pay (it has no sensors on its coin and credit card slots, which could have been used

to avoid the button press).

We understand that this user interface difficulty is caused by regulations: users must confirm before purchasing. One might have thought that trying to pay was tantamount to confirmation — but changing the user interface now with financial implications could require a costly new regulatory approval process. (If the financial systems were designed with usability as a priority changes would already be underway.) Of course, pressing a button called Purchase Ticket is hardly confirmation that the user wants *that* ticket.

The first time we encountered this problem, we assumed the machine was not accepting credit cards (maybe its communications line had failed) so we hunted around and found a £20 note. By the time we had correctly inserted it into the note slot, the machine had timed out. Despite the user being very active with the machine it times out, because it has no sensors to know the user is still busy with it. We have seen users walk away in disgust at this stage. (Incidentally, paying by £20 note gets, say, sixteen £1 coins as change, which takes a substantial time to drop into the ticket slot, and even longer to retrieve as the slot is an awkward 'pouch' without enough room to grab more than a few coins.)

2.5.8 Limited Accessibility: Identifying User Groups

Users with poor mobility have to get close to the ticket machine, and wheelchair users will find the high screen and high coin slots awkward. Blind or partially sighted users will not be helped that the buttons *always* make a beep, regardless of whether the machine is working or not. (Thus it is pointless rote-learning a sequence of button presses, since there is no way to know whether they are working.)

2.5.9 Bad Organisational Interface: Design for Maintainability

A discussion with a rail employee revealed he had not been trained to use the machine, let alone explain it to anyone else. The ticket machine has an internal user interface when the door is opened. There is an operator's manual, but he hasn't seen it. He has to empty the cash from the ticket machine from time to time; the first time he did this he set off police alarms, since he didn't know how to use the security interface! We understand the machine's manufacturers provide training courses but these have not been widely taken up by rail companies.

3 The Problem Restated

A computer system has been introduced into a complex system (i.e., the existing railway ticketing system, including its regulations) without taking the opportunity to reappraise the design of the systemic interfaces to the user. The interface design appears to be motivated by providing an "easy to use" interface to replace the current human interface, and was thus driven by (organisational) functionality rather than by (user) task analysis.

A user interacting with the system still has to contend with all the historical intricacies of the original complex system, as well as coping with (what is for them) a new interface to the GUI — itself implemented in a confusing way. In fact, the computer ticket machine has increased the complexity of the interface to the transport system. Fixing the conventional user interface problems that have been created is

a new problem, and one that is at or already beyond the limits of the designers' resources (they may not have been funded adequately to do a good job, or they may be unable to deal effectively with the complexity of the design task set them). The overall system goals have got lost in trying to solve the wrong problem at the wrong level, though nevertheless at a level that can be represented as a pushbutton graphical user interface. Unfortunately, the conventional user interface problems are quite interesting in their own right (they are also apolitical), and might divert professionals from tackling the bigger issues of design. It may be that, despite the new interaction problems, the automation of ticketing has obtained some gains in cost-effectiveness for the rail operators: there would then be little incentive to seek better solutions.

3.1 Summary of Concrete Design Problems

The following points briefly summarise higher-level issues from the discussion above.

- The software user interface encourages users to make errors of well-known and easily avoidable categories (e.g., post-completion errors), and fails at the end of interactions (e.g., when it has run out of change, or does not know fares). The user interface over-serialises: the user has to break down their task in the unpredictable order the machine specifies rather than an order appropriate for the user's specific task — the machine is not *permissive* (Thimbleby, 2001) (see also below). The user interface is misleading for partially-sighted users. (For example, the keys beep regardless of whether they do anything.) And so on.

- The user interface is inconsistent and, for example, has missing help texts. The recent introduction of multiple languages has not been checked. The quality of the user interface is very poor, and indicates the software developers are unaware of (or unwilling to apply) well-known international standards (ISO 9000 series, or ISO 13407 on user-centred design processess, *etc*) and undergraduate textbook techniques. Evidently, the programmers did not comply with the British Computer Society's codes of conduct (British Computer Society, 2001).

- In contrast, the hardware interface is very tightly constrained by concerns of vandalism, robustness and credit card security. We understand some of the hardware components (e.g., credit card verification) cannot be changed — though this is itself another usability problem, which should have been addressed much earlier.

- Rail staff have not been trained.

- The design increases rail revenue by discounting user needs. (Users who fail using it in the time available but who still travel will be fined; users who buy the wrong ticket can get no refund; and so on.) Apparently, the user interface designers did not consider the users' tasks or their frequency (e.g., the machine does not support buying multiple tickets to the same destination).

- Trying to improve the user experience leads to larger issues and challenging fundamental assumptions on rail travel and regulation.

The list reflects a variety of shortcomings in the design. A motivator to improve usability could come from HCI standards produced by organisations such as the ISO. The ticket machine has a label "easy to use." Farbetter, for all concerned, if this claim was validated by reference to some international standard. ISO WD 20282 is a potential candidate. The European Union is also working towards standardisingpresentation aspects (terminology and so on) of tourism information services. Such efforts might further improve a user's ability to understand the interface superficially. However, although conventional usability practices and standards will reduce user frustrations, we are advocating a fundamental shift in usability design.A more rigorous engineering-based design process would avoid many of them.

4 A Solution

The case study lists problems and attributes them largely to bad interface design. We must now turn from criticism to constructive suggestions, based on designing the interface to be as restrictive (in the system interface sense) as possible.

To summarise: the user's task is to travel; the rail operator's task is to raise revenue from passenger travel. The user can achieve their goals by getting on and off appropriate trains at the appropriate stations; the rail operator requires to know how to bill the user, where they start their journey, where they end it, and some time information (to allow for differential pricing). Conventional solutions, embedded in the historical necessity of "ticket thinking" has led to layers of complex interactions between large inflexible (institutionalised) subsystems. For instance, as we saw, eminently justifiable (and very rigid) rules about credit card transactions have a direct and deleterious impact on the user experience. Such rules cannot easily be changed, and certainly cannot be changed within the design brief of the ticket machine design. The crucial problem is that the interface exposes the user to this unnecessary detail and complexity. Graphical user interfaces (and web-style interfaces) are adaptable and flexible: they merely encourage this wrong level of design.

In contrast to GUIs, which have been so successful for personal computer systems, personal (small, ubiquitous, location aware, ...) technologies (PTs) seem like solutions looking for problems. In some sense they must be, because they are new and opening up new possibilities that have not been thoroughly explored. On the other hand, a characteristic of personal technologies is the elimination of the conventional user interface. Much interaction with personal technologies is implicit: they infer, through sensors and communications, what the user is doing. The consequence is that using PTs almost automatically restricts the interface: anything that can be inferred by the PT need no longer be part of the explicit user interface. In particular, as fewer things are explicitly mentioned in the interface, the interface necessarily becomes more permissive.

To develop a deeper understanding of the design issues, we now consider the simplest kind of PT appropriate to the ticketing task: one that simply provides user authentication, and can transmit that authentication data over a distance of, say, 3

metres. With that, we can propose a radically different kind of design solution that addresses most of the identified problems. However such a solution introduces, or rather emphasises, new issues: in hiding system complexity, issues of social complexity emerge as new usability challenges that are masked, or less central, when working with existing technologies. (In turn, 'society' can be considered a system, but to do so is beyond the scope of this paper.)

We assume the user and their PT are registered with the transport operator, with arrangements for billing (e.g., charging to the user's account, or to that of a parent under agreed conditions). There might be agreed limitations on the use of the PT to reduce the risk to the owner of the PT being stolen and used without proper authorisation. We can propose a scenario of use ...

If there are PT entry channels at stations, the user's starting station can log their entry into the system. At the destination station, the user passes through an exit channel, collecting a receipt showing the journey details and cost, so that they can check the bill when it arrives. If at any point there is a choice of a cheaper or more expensive option, this should be clearly displayed. For example, before the time when fares reduce from peak to off-peak, there should be a clear indication before the user enters the system to show that entering after (say) 9.56am will cost less than entering now. Similarly, if there are alternative train operators offering different services at different rates, there might be clear indications that the passenger is electing to use a more expensive service that will be surcharged. There would have to be additional PT detectors on such trains to bill for the higher grade service; the same principle would apply to those travelling first class. For regular commuters, there can be discounts (as currently implemented in 'season tickets,' or in other ways, such as 'rail miles' benefits).

Such a solution would address the problems identified earlier, as follows:

- There would be no end-user interface that suffers from problems such as inconsistency of interaction, spelling mistakes, missing help text, *etc*. A PT solution can hide a range of complexity in the existing system irrelevent to the user task: credit card equipment regulations, multiple ticket stock, adult/child distinctions, returns/singles/season tickets, change or the lack of it, penalty fares, vandalism, multiple operating companies, time restrictions, *etc*.

- As noted, the current hardware interface is constrained by concerns of vandalism and security. Radio transmitters can be located in less accessible places and be more easily protected against the weather. They would therefore require less servicing, and no routine servicing (such as repairing breakages, replacing ticket paper, emptying cash boxes). This would remove the need to train station staff in their maintenance.

- With radio communication with the ticket machine, any user interaction could always be completed, so there would be no risk of post-completion errors, provided the user did not have to collect anything physical from the ticket machine (e.g., a hardcopy receipt) at entry: the receipt at exit would only be for the user's records. The user task is defined by the user's actions; a family

group can be detected by the system through information stored within the system (on familial relationships and ages of children).

- There would no longer be an issue of over-serialisation (non-permissiveness) of the user task, or over the system adapting to users making regular journeys.
- The system would work as well for blind and partially-sighted users as for anyone else. It would also work for less mobile users who would not need to find and get directly close to the machine itself.
- Users would be charged for the journeys they actually make, and not suffer from buying an incorrect ticket by mistake. In fact a lot of complexity has moved from the paced interactive task to an unpaced review of the bill. (There is no reason why billing companies should not compete and therefore provide a better service than the current non-competitive, uninterested in users, system aspires to.)

A PT approach is easier to implement and to design thoroughly. A major problem with conventional user interfaces is that programs are serial, but users may work out what they want to do in almost any order (hence the requirement for permissiveness noted above). Often user interfaces are badly implemented, partly because of the combinatorial interaction possibilities. In contrast, the radio protocol for a PT is very simple (above the error correction layers). The main implementation task is maintaining a database of users and billing appropriately — a task that is already performed effectively by most utility (gas, electricity, *etc*) companies, and which should therefore be better understood that the current ticketing task.

From the rail operators' perspective, one of the disadvantages of PTs might be a loss of revenue from people making mistakes or buying more expensive tickets than they need to. However, it is likely that this would be more than offset by reduced costs of providing conventional tickets and reduced fraud (particularly fare evasion).

From the user's point of view, such a system presents some new challenges:

- There may be situations where the PT does not work. For example, one might be wishing to 'treat' someone else to a rail outing (for example, a children's outing): a conventional ticket would have to be bought for people not covered by PTs.
- The user may lack confidence in the system — for example, about billing or about whether they are actually authorised to travel — and, without a paper ticket, may feel vulnerable to over-charging or other error.
- Probably most fundamentally, such a solution raises privacy issues: as users are tracked through the system, their current locations may be traceable. Whether the issues are much greater for this use of PT than they are for established technologies such as mobile telephones remains to be seen.

The current ticket machines are intended to address 80% of user needs, leaving the remaining 20% to be dealt with in the traditional way by a clerk. One difficulty a

user has is knowing from the outset whether their current task is an “80%” one or a “20%” one: the user may try to use the machine and eventually give up, or may go straight to the clerk, which defeats the point of providing the machines. If the same approach is taken to PTs, they are not being proposed as the total answer to all travel billing needs, but as a way of dealing with most traveller needs, while leaving some to the ticket office. The ticket office might sell ‘single use’ PTs or what look like traditional tickets but with embedded disposable chips.

There are two areas in which the system may break down: failures associated with users, or associated with the technical infrastructure. Currently, the rail companies can cope with passengers who have mislaid their season ticket: such passengers can formally request a refund of the fine (but only a limited number of times over the life of the season ticket). Clearly the “non-functional PT” excuse will have to be treated equally leniently. Using network communications, the official should at least be able to check whether the passenger claiming to have left their PT at home in principle does have a PT. In the second case, where the PT infrastructure fails, simple safeguards (e.g., as are used in automatic bank transactions) can ensure the passenger is not charged, is refunded, or obtains compensation, as the case may be.

5 Conclusions

Despite more than thirty years of usability research, public interactive kiosks, exemplified by the ticket machine of our case study, are difficult to use, causing the public frustration and worse. We have analysed one such state-of-the-art system to show how fundamental and far reaching the problems can be.

The “invisible computer” has become a popular slogan, and personal technologies certainly make computers invisible to users. Books have been written on the invisible computer (Norman, 1998) and on the invisible future (Denning, 2002). From our arguments it should be clear that invisible computers are a side-effect of some good design, not a cause. We want to support the user’s tasks and experience, with as little irrelevant interference from the inner details and workings of the system as possible. Our approach is much deeper than just making the computer invisible: it means making unnecessary and irrelevant complexity invisible, thus making parts of the system invisible (which may imply making the computer more invisible). We support the user’s task by defining the system interface, not to control the computer (as a GUI does so well) but to support the operations required by the user. In the case of travel, the user’s task is obvious, and — as the case study made clear — personal technologies can directly support the user’s task and almost completely hide the intricacies of the rest of the system.

Hiding system complexity can solve or avoid conventional usability problems. We argued this abstractly; we showed that exposing complexity in our case study led to poor user interface quality. Poor quality is due to the user interface giving the user inappropriate access to what could have been hidden parts of the rail system. We argued that hiding the intricacies of the system makes the user interface trivial to use, and we showed a practical way to do so using personal technologies.

Personal technologies can improve user interfaces by narrowing system

interfaces. The original ticket machine is a wide and flexible interface to the rail system: System ⇔ User. Using a personal technology (as we described it) greatly narrows the interface and hides much system complexity: System ↔ PT ↔ User. In doing so the PT defines an interface that creates a separation of concerns that allows all interaction to be much simpler than with the original and, specifically for the ticketing system solution, for the user to interact at no time or order critical points.

Better interfaces highlight social problems. Having solved conventional user interface problems, social problems still remain. The simplest PT solutions raise privacy problems, which have no obvious "solution" apart from the public becoming accustomed to the out-weighing benefits of simple and reliable technology. Developers of more complex infrastructures need to be aware of the far-reaching social implications their designs may have.

On the basis of current implementation standards, simply getting anything to work reliably remains a problem. The ticketing machines studied here had problems partly because they were too complex for their designer/system interface. Unfortunately many of the schemes proposed in the PT literature are also complex (e.g., requiring complex network protocols). Trying to achieve a reliable user interface may force organisations (in this case, rail operators) to re-design systems holistically, rather than just attach user interfaces.

Good design will help transform society. Even within the limited scope of the case study it is clear that many conventional commercial working practices (such as the use of complex fare structures) will become counter-productive as better design — perhaps driven by the simplifying technologies of PTs — become more widely taken up. Travel, in particular, gives people benefits. Users may be willing to trade the obvious increased usability for the more hidden surveillance by PTs: the PT infrastructure is likely to become much wider than transport, to provide other business or even state benefits (e.g., combining the PT with a national ID card). For international travel, now with heightened security concerns, the tradeoffs are stark. The central and right concern of HCI, valuing the user, clearly now has a very important role to play in leading to a more peaceful and better world.

In short, user interface design is not about building user interfaces (such as GUIs) to an existing system, or to a computerised version of an existing system. The "system" is much larger than the computer that may run the user interface. User interface design is about defining interfaces, in the systems sense, that appropriately support the user's tasks and which conceal everything else. Doing so may result in a specialised interface that does one or a few things well, but therefore is much easier to implement reliably and easier to achieve satisfaction for the user. Perhaps one reason why user interfaces are so bad, and have been for so long, is because user interface designers have been concentrating on solving the wrong problems.

Acknowledgements

Harold Thimbleby is a Royal Society-Wolfson Research Merit Award Holder and acknowledges their generous support. We are grateful for some very helpful meetings with ASCOM Autelca of Gümligen-Berne, Switzerland, who manufacture the hardware but not, in this case, its software. This research was partly supported

by EPSRC Grant GR/R71467/01.

References

Adams, A. & Sasse, M. A. (1999), "The User is Not the Enemy", *Communications of the ACM* pp.40–46.

British Computer Society (2001), *British Computer Society Codes of Conduct and Practice*. Version 2.0.

Byrne, M. D. & Bovair, S. (1997), "A Working Memory Model of a Common Procedural Error", *Cognitive Science* **21**(1), 31–61.

Cheverst, K., Davies, N., Mitchell, K., Friday, A. & Efstratiou, C. (2000), Developing a Context-aware Electronic Tourist Guide: Some Issues and Experiences, *in Proceedings of CHI'00 (Conference on Human Factors and Computing Systems)*, pp.17–24.

Christian, A. D. & Avery, B. L. (1998), Digital Smart Kiosk Project, *in Proceedings of CHI'98 (Conference on Human Factors and Computing Systems)*, pp.18–23.

Denning, P. J. (ed.) (2002), *The Invisible Future*, McGraw Hill.

Friedhoff, R. M. & Peercy, M. S. (2000), *Visual Computing*, Scientific American Library.

Norman, D. A. (1986), Cognitive Engineering, *in* D. A. Norman & S. W. Draper (eds.), *User Centered System Design*, Hillsdale NJ: Lawrence Erlbaum, pp.31–62.

Norman, D. A. (1998), *The Invisible Computer*, MIT Press.

Parnas, D. L. (1972), "On the Criteria to Be Used in Decomposing Systems into Modules", *Communications of the ACM* **15**, 1053–1–58.

Sommerville, I. (2001), *Software Engineering*, 6th. edition, Addison Wesley.

Stanton, N. A. & Baber, C. (1997), Rewritable Routines in Human Interaction with Public Technology, *in* E. Hollnagel (ed.), *Proceedings of ECCS'97: European Conference on Cognitive Science*, Paris: EACE, pp.20–25.

Thimbleby, H. W. (2001), "Permissive User Interfaces", *International Journal of Human-Computer Studies* **54**(3), 333–350.

Thomas, P. & Thimbleby, H. (2002), The New Usability: The Challenge of Designing for Pervasive Computing, *in International Conference on Computer Communication*. In press.