# Design for a fax

*Harold Thimbleby*
Middlesex University
Bounds Green Road
LONDON, N11 2NQ
Email: harold@mdx.ac.uk

We argue that button-controlled devices can be made better to improve their usability, and that there are routine ways to do so that can be effectively employed in the early stages of the design process.

By way of example, we examine the design of a 1993/4 domestic phone/fax/answerphone machine, and show that alternative designs have quantitative advantages over the original. We point to many strange—obscure, undocumented, non-functional—features, and raise questions about the process that led to the design, and what role either human factors or computer science played in it.

## Introduction

The purpose of this article is to analyse the way a push button device's functionality is organised, and to show that such analysis informs design choices and raises interesting design questions. We can perform such analysis with or without without empirical user testing, so it is an approach that should appeal to industrial designers. We will show that alternative designs optimise various usability criteria.

For concreteness, this paper takes a particular device, the BT DF200, a domestic telephone, with fax and answerphone features.[1] Although this risks the paper seeming like a product review, it has useful advantages. Each example in this paper is based on the same device, so results may be compared. Our approach is shown to work with a real device, rather than a idealisation. And since the DF200 is a commercially available product it

[1] BT is a name of British Telecommunications plc.

would be possible for other researchers to repeat our work, or perform other sorts of experiments on it and to compare results.

Given the DF200 as a specific device, here analysed in some detail, we shall show that its design is inefficient *given* the choice of functionality its designers have chosen for it. We cannot be sure, but its design is so peculiar, it seems likely that no usability work was done for its design. The DF200 could easily be significantly improved (our work with other devices shows that this is typical; the DF200 is not an especially bad device).

The important points we make in this paper are covered by the following queries:

- Why is no serious usability design done in mass-market devices? Even small usability improvements would have major impacts, since the gains would be multiplied over the many users. This paper shows that significant gains are easy to achieve.

- If any usability *is* influencing the design of devices, exactly what is it achieving? Are manufacturers confusing marketing and market appeal for usability?

- Are consumers so uncritical of usability issues that usability does not matter?

- Has usability been so narrowly defined so as to exclude the sorts of issues raised in this paper?

For the sake of brevity, this paper does not question the general design approach, that is, of providing a push button style of user interface. There are many design considerations that could have led to alternative approaches (e.g., Norman, 1990). For example, the volume control could have had a better affordance than the current DF200 approach of using two side-by-side buttons. If the volume had been controlled by a slider or a knob, no doubt the designer would not have been able to even consider using the volume decrease button (*but not always*) as a sort-of delete key! We may charitably assume that market expectations, manufacturing costs, and other conventional reasons are sufficient to justify the style chosen. Even so, the chosen functionality itself, regardless of its packaging, is questionable; modifying the functionality and using an improved engineering approach would have led to considerable gain. We shall substantiate this claim in this paper.

It is no wonder that other research in user interfaces emphasises *explaining* systems, since they are so obfuscated. The FAX ASSISTANT (Gibbs & Tsichritzis, 1994), a video multimedia system using an animated human assistant to help people learn how to use a fax, is a case in point. The logical design of the DF200 is certainly so obscure that we can excuse the technical authors who wrote its manual: the DF200's bad manual is a symptom of an obscure and unusable machine, rather than a cause of the machine's unusability. A separate paper discusses the design of the DF200 manual and general ways to make it better and more reliable (Thimbleby & Ladkin, 1995).

*Personal comments*

I myself find the DF200 very difficult to understand and remember how to use without referring to the manual, and I had to expend considerable effort in determining how it actually worked for the purposes of this paper.[2] Though my (limited) ability to understand the fax during use may affect detailed numerical results below, the general issues raised do not depend on an exact understanding of the actual DF200—at least, any errors in my understanding will be consistent and will affect all examples equivalently.

The complexity of the DF200 itself and the effect of the inaccurate manual on my perception of it would call for caution in applying the results here to redesigning a "new DF200." If one wished to do that, obviously there are better ways of understanding a device than reverse engineering it without access to the technical details that are or should be known to its manufacturers.

Many of the design problems we explore here have been discussed elsewhere, and for a considerable period. For example, (Thimbleby, 1978) is a review of menu selection (i.e., the method underlying the DF200's function selection); and (Witten, Cleary & Greenberg, 1984), a decade old, is a well-known relevant development. There is a *vast* literature on Human-Computer Interaction—even if we only consider that part of the literature that predates the design of the DF200 by, say, a decade, which its designers should have been well aware of. Although this literature certainly relates to the design of the DF200 (and similar devices) there is no evidence of its being recruited to the design process.

The DF200 is made by a French company, Sagem, which also manufactures the Exocet missile and other military hardware. This paper does not consider how easy or safe military hardware is to use, but I can raise the question, if military hardware is usable, why can't a simple domestic product be designed to similar usability standards?

Maybe BT or Sagem *have* done some empirical usability testing and have determined that the design of the DF200 is acceptable to a majority of users (cf. Nielsen, 1993). If this is so, then the discrepancy between the implications of the findings reported here and BT's empirical work would deserve examination and explanation. I think, if users find the DF200 'usable,' this is probably because they understand the device so poorly they are unable to arrive at an informed critical opinion of it. If so, this calls into question all user-centred design approaches. Alternatively, I imagine, any such studies with the DF200 evaluated users' willingness to *buy* the fax or its particular mix of features—not users' ability to *use* it, or even their ability to understand much of its manual.

Though this paper is not intended as a product review of the DF200, readers may be interested in other aspects of its user interface that we do not analyse in the paper; the Appendices cover various non-systematic issues of its design—of necessity these are product-specific comments. For example, it has commands that don't work, commands that are not documented, and misleading "garden path" features.

---

2 The shortest check of every transition in the FSM used for this paper's analysis requires more than 516 key presses error free, the length of the optimal Chinese Postman Tour of the FSM.

## The DF200 functionality and its analysis

The DF200 has 49 functions of interest to us, organised as a tree (i.e., a hierarchy of choices) giving access to options and data-entry points. The DF200 has several other functions, such as standard telephone functions, which we will not analyse. The basic structure is easy to analyse; for our purposes a small Finite State Machine is adequate. Our analysis will suggest that alternative organisations may have been preferable. The DF200's specification, as used here, has been presented elsewhere (Thimbleby & Ladkin, 1995).

Our method is very simple: we calculate the cost, as a number, of the user doing tasks with the DF200 and compare the numbers with the corresponding costs of a alternative designs that support the same functionality as the DF200. The general method is described in (Thimbleby, 1994).

The DF200 can also be simulated interactively, using the method described in (Thimbleby & Addison, 1995), an approach that *automatically* gives it *correct* manuals and intelligent interactive help. We have also discussed elsewhere how the user manual can be correctly developedeven under changing design requirements, as would happen in iterative design of a product (Thimbleby & Ladkin, 1995) .

Though we will not modify the original functionality for our analysis, we should point out that the functionality *is* questionable. For example, there are three types of print summaries for phone number abbreviations. These and other functions might have been combined, providing a simpler and more rational design.

### *Definitions*

Various functions on a device take differing numbers of keys to perform. Entering an identifier takes two or more key presses, depending on how long an identifier the user wants to enter. We say that a user **accesses** a function, meaning they are then able to enter what specific data they wish, as opposed to **activating** a function, when the fax acts on the user's already entered data. In all cases on the DF200, activating a function takes one more press than accessing it and entering the data (or selecting from a menu).

Numerical key counts (unless stated otherwise) refer to numbers of key presses required to access a function starting from standby mode.

In practice, a designer would take a weighted normalised sum of such numbers—and others—and hence choose the best design for the intended tasks. For example, frequently used commands are likely to be memorised by the user so the their impact on the user is how costly they are to *access*; whereas infrequently used commands have to be found, so for these the cost of *finding* the commands is more important than how costly they are to use once found. Suitable weights would be chosen to represent the users' task mix.

To illustrate: the DF200 has an average function access of 5.9 key presses but a complete search cost of 117 key presses. In other words, the DF200 favours familiar users, for they can achieve an average performance ten times better than users who have to search for commands. As we shall see below, the complete search cost of 117 is an under-estimate of what a user would be expected to achieve on the DF200, for the search task is so complex that users would be expected to make many errors and so take more key presses than the minimum.

An alternative design (proposed below) has cost of 24.5 and 49 respectively. This alternative, on average, is harder to use for any task, by about 19 presses, so it is over three times slower for familiar users. On the other hand, if the user is unfamiliar with the device, on average they will only spend 24.5 presses searching for what they want—which is a significant improvement over the DF200, which requires about 140% greater effort. Moreover, this alternative design is extremely simple, and therefore has merit because it is easier to learn how to use. So this alternative design would likely be much better for a "walk up and use" environment than the DF200 is. We can assert this without close analysis of how a fax/answerphone might be used in practice. If, additionally, we knew (established or assumed) how the frequency of use of different functions varied, then we could quantify precisely how the second design was favourable. In fact, it is tempting, and may be useful to guess weights: the DF200 has some commands that are used any time a fax is sent, and some used very rarely, such as resetting its clock, which is only necessary after a power cut.

We are interesting in measuring how easy or difficult a device is to use. Define the cost to the user of activating a function $f$ as $c_f$ A function with low cost is better than a function with high cost. We can measure cost in many ways; cost is the converse of utility. The more often a user requires a function the greater its utility. The more a function contributes to the user's business, the greater its utility. The utility could also be related to marketing considerations—there would be little point having a selling feature that salesmen found difficult to use! Using the notation $c_f$ is reminds us that the cost of an operation *to the user* is not known unless the subjective factors of $c_f$ are agreed and established.

From a design specification, regardless of what a user does or prefers, we can easily and objectively measure the cost of accessing a function, $b_f$ measured in button presses. This cost clearly influences the time it takes a user to do something, since the user is physiologically limited in their button-pressing rate; also, the more buttons that need pressing in sequence, the greater the probability of error. We can account for these effects by a function $T_f$ that maps the button press count to a time for the user. Note that $T_f$ depends on $f$—it may be that the user is more anxious about the use of certain functions than others, which would affect $T_f$ differentially.

The idea is, of course, that $T\left(b_f\right)\times c_f$ is a measure of the cost of accessing feature $f$.

From these considerations, the average badness of a design (its meanness?) measured as a time is

$$\frac{\sum T_f\left(b_f\right)c_f}{\sum c_f}$$

where the sums are over all functions $f$ provided by the device. A designer should seek to minimise this measure to make the design more

'efficient.' However the average cost is not the only measure of design quality. For example, max $T(b_f) \times c_f$ is the worst case.

Average and maximum are just two of many statistics we might consider; which statistic measure the designer should optimise depends very much on the users' tasks and environment.

Even considering a battery of statistics, a keystroke level model (Card, Moran & Newell, 1983) such as this is a very simple design cost model. It does not easily cater for error avoidance techniques in the design, because it is simply based on button press counts. The DF200 has no features for error repair in any case (e.g., it has no undo button).

More buttons mean a more complex physical space for the user, and fewer buttons would mean a more complex logical space for the user; properly weighing such trade-offs rapidly gets into deep psychological waters, and generally will raise design questions that cannot be answered *a priori*, which is one of our aims.

All such considerations introduce uncertainty, and make it harder to compare designs. Clearly, a different set of functions influences the relative cost of the functions to the user. Rather than get bogged down it details, it is easier to take $T_f$ and $c_f$ to be constant. We then obtain the simple average:

$$\frac{\sum b_f}{n}$$

where $n$ is the number of functions.

Perhaps misunderstanding this equation has led to many designers to increase the number of functions of a design in the hope of improving its usability—by increasing $n$ to make the cost less! The point is: you should not simply increase the number of functions, but should also consider the structure of the design, and hence modify $b_f$. In our examples below, we do not change the number of functions nor what they do; we only modify $b_f$, and we show what considerable variations are possible.

In the absence of empirical evidence the analysis below assumes every function has uniform weight. This shows how such analysis might be of benefit even before a (novel) system has been built, and hence one where empirical work would have been difficult. Our purpose is to show that design facts are readily available to support many useful design analyses and trade-offs.

## Analyses

We shall consider only three simple tasks: how long it takes the user to perform a command knowing how to do it; how long it takes to find a command; and the worst case time to perform a command. (By 'time' we are taking $T$ as giving one second per key press.)

| Design | Average time | Maximum time | Complete search |
|---|---|---|---|
| Actual DF200 | 5.86 | 13 | 117 |
| Linear search | 24.5 | 49 | 49 |
| Direct function | 1.9 | 2 | (1.9) |
| Binary tree | 5.69 | 6 | 53 |
| Basic trie | 9.29 | 17 | 49 |
| Hybrid trie | 3.18 | 5 | 49 |

These figures should be read in conjunction with the explanatory text. For example, the rather high average cost of the linear search design is the consequence of having a very simple user interface, which might be a worthwhile trade-off.

It is perhaps unfair of me to point out that my DF200 has failed several times and has had replacement parts: each time it has been repaired it has changed its design (see Appendix 2). Evidently, it is possible to modify the DF200 design. There is no technical obstacle to improving it.

### *Function access on the actual DF200*

The average cost of a function access is 5.86 presses *error free*, provided the user knows how to access the function (the worst case is 12 presses from standby, or no more than 13 presses from anywhere). These figures assume the user knows how to get to the function, perhaps by following exact instructions in the user manual (an assumption that is questionable).

If the user has to search for the function, then a systematic search of the functions amounts to a travelling salesman problem, and the user is unlikely to do this efficiently. If the user knows that the functions are organised in a tree, and the STOP button can be used to return from any point to standby, then they *might* embark on a depth first or breadth first strategy to search. Either approach takes a maximum of 117 key presses, and on average the desired function would be found half way, at 58.5 presses (plus one more press to activate it).

If, despite being so systematic, the user missed seeing the required function the first time they located it, they would have to 'go round again,' taking them another 117 presses (175.5 presses on average) *if* they spotted the desired command the very next time it was accessed. If they *don't* recognise when they visit a level for the second time, they could spend forever going around in circles.

### *Alternative: linear search*

The DF200 clearly poses serious problems for users who know there is a particular command they want, but do not know where it is to be found. An obvious design approach to overcome this is to organise the commands in a simple linear sequence; searching for a command now becomes trivial, especially if the fax is given a button called, say, FIND COMMAND .

There are 49 functions in the DF200, and they can be searched linearly using two existing keys: for example, OPTIONS starts the search, and subsequent presses of it progress through the list of functions. Pressing START COPY activates the chosen function. (We could use the volume decrease button because it looks more like 'scrolling' than the OPTIONS key, but this would make no difference to the analysis.)

On average, the function the user searches for will be half way through the list. So a user has to press 1+49/2=25.5 keys. In the worst case, they have a very simple and reliable procedure for accessing any function in a maximum of 49 presses.

If the functions are sorted most-frequently used first, then the average can be weighted, and much better results than 24.5 would be expected in practice. On the other hand, if the list was sorted like this, then a user might despair before finding an infrequently used command that in any case they were not too sure about using! It might be better to order the commands alphabetically, so that the user always has a sense of progress through the list. Doing so would then raise questions of judicious command names; related commands could have similar names so that they are close to each other in the sorted list.

*Alternative: direct function number*

Suppose a list of all functions was shown on the front of the DF200, together with a number, 01 to 49. Every function could therefore be accessed with exactly three key presses, say [OPTIONS] -*digit-digit*; moreover, the user would not need to remember what the functions were called or how they were organised, as the summary would be an adequate reminder.

We can do better by using a more compact numbering scheme, numbering the 48 functions as 0, 1, 2, 3, 4, 50, 51… and all numbers to 93, and realising that as the digit keys currently do nothing when the handset is down they could be used for accessing commands, without requiring the [OPTIONS] prefix. These modifications would bring the average access down to 1.9.

It takes a maximum of 2 key presses to access a command; this implies sometimes the user will have pressed only one key and not completed the command access. What should the fax do in these circumstances? The naïve solution is that after a time-out (the DF200 takes 30s), the fax would revert to standby. But it is also possible that the first key press was accidental—should the user wait 30s for this error to be rectified? What happens if the first press was an accident and went unnoticed by the user?

A better solution is to arrange that the command accessed is the result of the *last two* key presses, as opposed to the *precisely* two key presses counting from standby. This ensures that the user need never worry what an earlier key press was. Regardless of whether the fax is in standby, the user can enter two keys for the required command and it is accessed—even when there were prior key presses ‘partially’ accessing a command. For example: suppose command 43 is NUMBER OF RINGS, and the user has already pressed [5]. Pressing [4] accesses (but does not activate) command 54—the user would have to press [START|COPY] to activate it—and, next, pressing [3] accesses NUMBER OF RINGS.

The DF200 provides a delete key (curiously a double, and unlabelled, meaning for the decrease volume button): this key can also delete function key digits. It may not seem very useful or important to correct a two-digit sequence; but the main advantage of this facility is to increase the frequency with which the user can use the delete key, and hence to increase their familiarity and skill with it. Specifically, it generalises the meaning of the key so that it depends less on modes.

*Alternative: search by binary tree*

The DF200 uses two keys, OPTIONS and START|COPY to access each function. Two keys are sufficient to perform a binary tree search, say with OPTIONS going left and START|COPY going right. If we do this with a minimal height tree, we find the average function access is 5.69 key presses, and the maximum is 6 (where over half of the commands are to be found). By the simple expedient of linking every command together at the leaves of the tree, every command can be accessed by getting to the (say) left-most command (5 presses) and then continuing left from there (with a maximum of 48 more presses).

Like the actual DF200, this scheme relies heavily on the user understanding the organisation of the tree. We suggest the tree might be organised alphabetically. Even an alphabetic organisation groups commands helpfully for the user: thus ACCESS (set free or protected) is adjacent to ACCESS CODE, rather than in unrelated places, as on the DF200.

*Alternative: search by alphabetic trie*

Using more than two choices speeds searching a tree, provided the increase in choice is systematic. If we use lettered keys to choose the next letter of the function name (that is, using a trie), we achieve an average function access of about 13. However, most choices in a trie are unique, and taking advantage of this reduces the numbers of button presses to 9.29 (max. 17). So far as the user is concerned, the fax's commands are accessed by using a simple dictionary lookup. A trie necessarily achieves an organisation at least as sensible as the DF200's choice of command names!

The full trie requires 27 symbols using the DF200 manual's exact names for each command, but by mapping the commands onto the letters that each of the DF200's keys *already* uses and ignoring punctuation, only 8 keys are required. With this transformation, the average and worst cases are unchanged. So we do not even need to change the keyboard.

So far as the user is concerned, they now type the command (by pressing the digit keys with the right letters) and pressing START|COPY when the desired command appears in the LCD. The LCD would appear to work like an old-style command-completion user interface. If the user wanted a direct search (e.g., not knowing how to spell a command) this can be achieved by pressing OPTIONS, which would take the user onto the alphabetically next command in sequence (and wrap around at the end, so no command is ever missed).

As has been pointed out by Knuth (Knuth, 1973), a combination of a tree and a trie achieves better results than either. The same is true for the DF200. The top level of the trie splits the commands into 7 sets, but if binary trees are used in each of these seven sets then the overall average becomes 3.18 (max. 5). Exactly how one proceeds with this variety of design ideas will depend on wider issues and trade-offs, particularly between speed of skilled use against memorability, error-recovery and—perhaps primarily—what the design looks like, for that is what sells it.

These methods *sound* obscure, but that does not mean they are obscure to use. On the contrary, we have evidence that these techniques are surprisingly easy and effective.

**Note**: because it has significant usability advantages, the technique described in this section is subject to a patent application. We plan to discuss full details at greater length in a subsequent paper

## Variations

### *Search by prediction*

To some extent, but not consistently, the DF200 supports search by prediction. That is, if some command $c_a$ has been accessed, some command $c_b$ is immediately easier to get to than by returning to standby. The fax has 'predicted' that the user will wish to access $c_b$ after accessing $c_a$. It is possible for a sophisticated gadget to try to predict $c_b$ given the past choices of the current user, but most systems make static assumptions about the typical user's behaviour.

An example from the DF200 is as follows: if the user sets the network type (public or private), they can *then* very easily specify whether it uses pulse or tone dialling. In fact, activating the network command itself accesses the pulse/tone command. Here the designer's prediction has a dramatic improving effect on the efficiency of a certain sequence of actions.

On the other hand, if the user changes the answerphone mode from fax/answer to answer only, then the manual recommends recording a new announcement (so it no longer says sending a fax is possible). Yet recording a new announcement is not easier to reach after the change; in fact, needing to record a new announcement is presumed to follow listening to the current announcement, and it has to be reached from standby.

Another example arises after a power failure. The fax prints a "Check date and time" warning when power returns, yet setting the time is still as difficult to do as ever: it has to be done from standby, as usual. It could have been made easier: the printout could have said: "Check date and time. Press START COPY to do this easily." Here the prediction is based on the world's actions on the fax; prediction can be based on more than just the user's direct actions.

A different sort of example is represented by the DF200's REDIAL. The designer has made the assumption that redialling the last number is an activity the user will often wish to do (because the number might have been engaged); in fact, because this prediction is so specific, it can be achieved with a single button press. It is interesting, then, to observe that the same reasoning did not lead to a REDO command that not only could redial, but could take the user back to the last command they attempted—for instance *just in case* the user wanted to check their changes or to correct them.

Note that a predictive design would not change the simple statistics used in the analyses above. Conditional probability distributions of the user's tasks would be required to show the advantages of a predictive design; alternatively, the design could adapt to the particular pattern of use being made of it by a specific user.

The standard problem with predictive interfaces is that just as a user gets used to how a device works, it changes trying to make its use more efficient. Thoughtful design could also separate the predictive part of the design from the conventional; for example, a button called GUESS might get the device

to try and guess the most plausible next command, but the design would otherwise be unaltered.

### *Error-prone use*

The previous analyses assumed error free behaviour. Of course, for key press sequences of 12, even if the user knows what is going on, there will be times when the wrong keys are pressed. It is easy to perform the analysis on the assumption that the user makes key press errors with probability *p* using Markov models. In the present case, a Markov analysis does not reveal much other than to confirm that key sequences to access a command get increasingly longer with the length of the error free sequences. This is another reason to prefer easily accessed commands.

Suppose the user does not know where a command is, and is prepared to search randomly—on the wise assumption that an intended systematic search might forever miss a command if the user did not understand the system's actual structure. Take the alarm setting command on the DF200, which is the easiest command to access (3 key presses): pressing keys at random requires at least 27 presses to have a better than even chance (50.08%, in fact, for exactly 27 presses) of finding the alarms function. Whereas, if the direct function number interface was used (see above) instead, *any* command could be found with better than even chance after just 26 presses.

After 27 presses, the user of the DF200 has a slim chance of accessing the alarm command; after just 2 presses on the function number interface the user has a certainty of accessing some command, and therefore of obtaining an example of how *all* commands are accessed. A user can 'play' randomly with the interface and learn how all of it works; this is not possible on the DF200, and the user would have to be quite persistent to get any success from which to generalise (somehow) its mode of operation. So there are some user interfaces that are better than the DF200 even if the user does not know how they work!

## A city is not a tree …

Christopher Alexander's classic paper, "A city is not a tree," (Alexander, 1965) makes an eloquent argument for not designing cities as trees, but rather as semi-lattices. A tree-like organisation isolates activities (industry, education, health, housing, …) into separate areas without overlap. This means people have to travel between areas, and their lives become compartmentalised—with increasing problems as they become older, and isolated into regions specialised to old peoples' needs. However, the tree structure suits designers because of its conceptual simplicity.

We see a similar effect in the design of the DF200. Its functionality is organised as a tree, with each function isolated into its area. Unfortunately, the area it is isolated in is the specific and unique area the designer thought appropriate. Unfortunately the user may not see it like that!

In the case of the DF200, the designer decided that printing was one area and defining one-touch fax numbers another, separate, area. How, then, does a user print one-touch fax numbers? (In the print area.) The designer decided that setting the machine to behave as an answerphone would reside in the *initialisation/parameters/answer param/answer* area; but to record the answerphone's message (which a user might well wish to do at the same time) resides in *answering/announcement recording*—somewhere completely

different. Similar points could be made about access codes, abbreviated key names, baud rates, and so on.

If the fax was a city, Alexander would recommend it was a semi-lattice. This would be appropriate for a fax: a function such as 'print one-touch fax numbers' can be in *both* 'print' and 'one-touch fax number' categories, and the user could find it easily however they classified it. The drawback of this organisation is that it increases the number of key presses required for a systematic search when the user has no idea how a function is classified, though the trie approach demonstrated that the user interface can still appear very simple (even if trie is not a familiar name to the user!).

The point is, and here the analogy with Alexander's city argument is overwhelming, that designers, whether designing cities or faxes, grossly over-simplify, and not for the benefit of their products' users.

### The computer science view

If a program is implemented as a transition network, then transitions can be added in *ad hoc* ways. The DF200 is a simple FSM and it might be conveniently implemented in this fashion; if so, this would explain the incoherence of the design. Better programming practice suggests, instead, factoring out abstract operations (e.g., what each user action does) and implementing them top-down. This approach might have led to a collection of functions implementing buttons, and a data structure representing the state space. Done like this, it would have been quite hard to make the user's actions do different things in different places in the state space.

When people design programs, standard practice is to define abstract types, operations, modules … the details vary, but generally the idea is to design top down, and to rigorously specify what is intended so that the outcome may be proved and/or debugged against the initial ideas. The reasons for doing so are to better manage complexity.

The design of the DF200 suggests that the user operations were not considered as part of the program design, moreover the program design was probably such as to conceal any relation between the program and the user's language for operating the device.

There is a vast literature on algorithms. The user's task, so far as analysed in this paper, is *searching*.[3] The DF200 has a database of commands, and to perform any task the user must locate the corresponding command. The designers of the DF200 chose to use a peculiarly structured tree, with no obvious advantages: we showed that its maximum depth, average depth and total path length were all sub-optimal. It seems the operations available to the user to support any search were simply not considered.

Every book on AI or algorithms of which I am aware has a section on searching, and all of them provide better ways of doing it than the DF200 employs. The designers of devices like the DF200 seem to be unaware of standard algorithms. They are certainly unaware that they ought to provide users algorithms, or allow users to use effective heuristics at least as carefully chosen as they would chose when designing the insides of the device.

It seems the DF200 user is not treated as well as a computer would have been to do the same tasks. Surely the user deserves to be treated at least as well as a computer?

---

3 It is still called searching even if the user knows where to look.

## Limitations to the approach

There are several limitations to the methods presented. This section briefly dismisses them as inconsequential to the wider perspective.

*The model of the DF200 was approximate.* Product designers should know exactly what they are designing, and there should be no need to approximate a proposed design. The approximation used here, being a simplification, possibly gives an advantage to the DF200 when compared to the alternative designs proposed.

*The analysis has no empirical foundation.* Rather, it made no assumptions (e.g., cultural assumptions) about user behaviour. The results are conservative and may be quantitatively compared, without risk of being affected by users' changing behaviour across design alternatives.

*The analysis assumes error-free behaviour.* Similar analyses could be made making more realistic statistical assumptions about users' error distributions (Witten, Cleary & Greenberg, 1984). However, given the lack of empirical data and the inaccuracy of the DF200 model used, it was inappropriate to do so in the present case.

*The analysis ignored time.* The analysis did indeed ignore all temporal issues. We ignored time-outs, which are clearly significant design features: all time-outs on the DF200 return the user to standby, and therefore the user has to start again. In our experience the time-outs (30 seconds) seem very brief and frequently interfere with reading the manual. For example, the explanation of programming keys (p23) refers the user to two other pages in the manual (p12, p14); likely, by the time the user has found, read, and returned to continue reading, the fax will have timed-out. Nevertheless, by ignoring time we made our analyses clearer, the assumptions simpler, and their limitations more obvious.

*The analysis was trivial.* Well! Yes it *was* trivial—but in the sense of easy to do. So why wasn't it done by the manufacturers? If a designer can't analyse even a trivial system, then what hope does a user have of understanding it or any of the more complex systems that are widely available?

*The work is inapplicable to real systems.* No; that is why we chose to analyse a specific product, the DF200, rather than a 'generic' product, possibly simplified. We analysed a real product, and one that is currently being sold.

*The analysis ignores interpersonal differences, psychology, social context …* Certainly it would be useful to inform the analysis with human factors, however to do so runs the very serious risk of making profound assumptions that are hard to identify. The analysis here has *obvious and explicit* limitations, and that is an advantage.

## Conclusions

The DF200 shows a degree of arbitrariness that makes its analysis awkward. In fact, we performed an analysis of a simplified machine[4] because the actual machine was not documented clearly enough to understand its structure, nor was the actual device easy enough to manually explore systematically with any hope of certainty. The analysis nevertheless showed quantitative inefficiencies, and indeed confirmed the difficulty of a systematic exploration of the DF200 by hand. It is tempting to jump to the conclusion that the DF200

---

4 For example I omitted all details of the DF200 debug and trace commands, which in any case are not documented.

was specified in a very disorganised fashion, probably in very low level hardware terms. Whether that specific conclusion is probable, a plausible inference is that there is no deliberate relation between the DF200's design and any usability requirements.

Besides critiquing specific details of the DF200 design, we have shown that alternative designs would have offered quantifiable usability trade-offs. These trade-offs and costs are extremely easy to explore, and in a proper design exercise might be enhanced by empirically obtained weights. It would be extremely interesting to learn from BT (or their suppliers) what their design process actually entailed, what usability engineering was involved, and how this was reflected in the outcome, the DF200 design itself.

The best is the enemy of the good. We could of course construct a better design approach, but it would entail a research project rather than be a method that could be applied in industry today. The limitations of the approach suggested here are obvious, which is why they are easy to accommodate. A more sophisticated approach would still have limitations, and would probably be much harder to use in the early design stages, where it would be most useful. And as we tried to be more realistic to particular users and tasks, we would start to raise wider questions, such as cultural assumptions, which the designers may be unable to address objectively, being themselves embedded within a culture.

Most importantly in this paper, we have shown that basic computer science, namely *designing* algorithms for the user, can be used to improve user interfaces: using routine computer science, we showed that useful usability analysis for a real device is feasible and results in productive design insights.


## Acknowledgements

Matthew Jones and Gary Marsden provided many useful insights and encouraged me to publish this paper.


## References


C. ALEXANDER, "A city is not a tree," *DESIGN*, **206**, pp46–55, 1965.

BT (undated), *DF200 User guide*, (Apparently coded: UM DF200 23175349-2), British Telecommunications plc., London.

S. K. CARD, T. MORAN & A. NEWELL, *The Psychology of Human–Computer Interaction*, Lawrence Erlbaum Associates, 1983.

S. J. GIBBS & D. C. TSICHRITZIS, *Multimedia Programming*, Addison-Wesley, 1994.

D. E. KNUTH, The Art of Computer Programming, **3** (Sorting and Searching), Addison-Wesley, 1973.

J. NIELSEN, *Usability Engineering*, Academic Press, 1993.

D. A. NORMAN, *The Design of Everyday Things*, Doubleday: New York, 1990.

H. W. THIMBLEBY, "A Note on Menu Selection," *Computer Bulletin*, Series 2, No. 18, pp20, 21 & 23, 1978.

H. W. THIMBLEBY, "Formulating Usability," *ACM SIGCHI Bulletin*, **26**(2), pp59–64, 1994.

H. W. THIMBLEBY & P. B. LADKIN, "A Proper Explanation When You Need One," in M. A. R. Kirby, A. J. Dix and J. E. Finlay (editors), *BCS Conference HCI'95*, People and Computers, **X**, pp107–118, Cambridge University Press, 1995.

H. W. THIMBLEBY & M. A. ADDISON, "Hyperdoc: An Interactive System Development Tool, in M. A. R. Kirby, A. J. Dix and J. E. Finlay (editors), *BCS Conference HCI'95*, People and Computers, **X**, pp95–106, Cambridge UNIVERSITY PRESS, 1995.

H. W. THIMBLEBY & I. H. WITTEN, "User Modelling as Machine Identification: New Design Methods for HCI," in *Advances in Human Computer Interaction*, **IV**, pp58–86, D. Hix & H. R. Hartson, editors, Ablex, 1993.

I. H. WITTEN, J. CLEARY & S. GREENBERG, "On frequency-based menu-splitting algorithms," *International Journal of Man-Machine Studies*, **21**(2), pp135–148, 1984.

## Appendix 1. Basic features of the DF200

The DF200 is a small desktop fax unit, with an external mains power supply. The fax has 39 buttons: 12 for dialling out, 10 for abbreviated numbers (i.e., for quick dialling), 7 for controlling the answerphone recorder, 3 for monitoring and volume control, 3 for the main functionality user interface, 1 for fax detail control, and 3 apparently for phone control (one, [REDIAL] works; one only works with a local switchboard but its functionality is undefined; the manual does not mention the third button and it appears to do nothing).

The handset itself also provides a 'button,' in that the functionality of the fax changes when the handset is lifted.

There is a loudspeaker, which is used for monitoring the handset ear piece, listening to recorded messages, and for making a small selection of beeping noises.

There are 6 LED indicators, inset in buttons [ANSWER], [PLAY], [FINE], [STOP], [START|COPY] and [MONITOR]. Finally there is an LCD display, capable of displaying 12 upper case letters and some punctuation, such as #, as well as space.

To access commands, the user should proceed as follows, which quotes verbatim (including original emphasis) from the user manual's explanatory chart:

> "Press [OPTIONS], and the press the number on the telephone key pad relating to the programming trail you want to follow. At each successive stage of your trail, press [OPTIONS] to scroll through the list in the box, and press [START|COPY] to verify a function choice, or continue to another box. (Where a box contains functions which require your choice—choices are in brackets—you will need to press [START|COPY] to scroll and **not** [OPTIONS].)
>
> An example (changing the dialling type from PULSE to TONE) has been highlighted for you to follow:
>
> Press [OPTIONS], then press [1]. Press [START|COPY]. Press [OPTIONS] three times and then [START|COPY]. Press [START|COPY] again and then [OPTIONS] to change to tone dialling. Press [START|COPY] and the red [STOP] button to end the programming sequence.
>
> Note: Your DF200 can remind you which trail to follow, so that you not need [*sic*] continually refer to this chart. At the start of the programming sequence, additional presses of the

OPTIONS button will reveal the number on the telephone key pad you need to press for each sequence."

We now quote from the manual (page 26) to see how to perform polling:

> "If the machine you are polling from has a password (see page 19), you will need to match the passwords of the two machines. Polling can only occur when both machines have the same password."

Page 19 refers only to "access code," which *may* be the same as "password" here. Page 19 is called, "Using the answering machine from another location," and does not appear to refer faxes, polled or otherwise.

If polling can only occur when both machines have the same password, this appears to mean that your own fax's password has to be changed to be the same as the remote fax. This would seem to compromise your own password security.

> "Press OPTIONS and then 4. The display will show POLLING BASE and (SELECT) alternately."

I cannot find an explanation of the "POLLING BASE" feature anywhere in the manual. (Selecting it results in a request for a three digit 'base number, then a four digit password, then a number, then a time. During this process, it is easy to get the LCD to display `AN/OT UNDEFINED`, which the manual does not define in its list of error codes. In contrast, if OPTIONS then 4 is pressed when there is a document in the fax, the LCD shows `DOC TO BE POLLED`, a feature explained on page 27.)

The manual continues:

> "Now press OPTIONS again, the display will show POLLING and (SELECT) alternately. Then press START/COPY. The DF200 will then ask you to enter the number of the remote unit from which the document is to be polled."

To say it "asks" you is an exaggeration, as the next sentence makes clear.

> "Display shows ENTER NUMBER and goes blank while you key in the number on the telephone key pad."

The text `ENTER NUMBER` is displayed very briefly, for about 1 second. Unusually, the LCD does not alternate between what you are entering and the `ENTER NUMBER` information. (If left to its own devices at this point, the DF200 will shortly reset to standby.) In other words, in the time you have available, a blank LCD panel is "asking"!

> "At this stage you can also enter a memory key (see page 14 for information on memory keys)."

In fact, page 14 immediately refers to page 23 for fax memory keys, which discusses them solely in terms of *transmitting* faxes. Here, we are trying to receive one. Moreover, you cannot 'enter' a memory key at this stage, whether before or after the number. What the manual means is that a predefined memory key can be used, rather than be defined. We allow that this is an ambiguity in the use of the word 'enter'—which could have been resolved by better wording.

## Appendix 2. Specific design flaws of the DF200

Specific design flaws can be categorised under three headings: apparent lack of imagination; inconsistency; and general inadequacies—we do not repeat structural design issues that were covered in the body of the paper. Within these categories (expanded below) the lists are in no particular order. In principle a cost/benefit analysis can be made: comparing the cost to users against the cost to the manufacturer. The list of flaws might then be ordered in decreasing impact. However, in the flaws we list below, the cost to the manufacturer is only in the effort of designers' thought, not in materials: thus the cost would have been amortised very quickly, and might well be considered trivial.

To facilitate reference, the points are numbered.

### *Apparent lack of imagination*

1 There is no [UNDO] button. If the user *notices* a mistake, they cannot do anything about it, other than press [STOP] and start over again. Some examples are given below where the consequences of a mistake are considerable, such as having to re-enter the entire fax set-up and telephone numbers.

2 Unlike many other faxes, and despite being able to print all sorts of technical information (including some related to non-functional features), the DF200 cannot print a command help summary. One would have been very useful. Indeed the manual (p.20) suggests making a copy of part of the manual. Ironically, although the DF200 does have a photocopy feature, the manual is bound in such a way that the DF200 cannot be used to photocopy its own manual.

3 When the fax is being used, the LCD panel often shows the text REPLACE HANDSET. Under these circumstances, the user is holding the handset (likely because they want to make a call) and, evidently, the fax knows the handset needs replacing (e.g., because the last called number has disconnected), so why doesn't it simply perform whatever electrical operation is required itself and save the user the inconvenience? It seems to me as a user that the fax is being terribly condescending, as if it is saying, "Your last call has finished and you now want to start a new call. Put the handset down and pick it up. I could do it for you, but you ought to know that's the proper way it should be done."

4 When my DF200 broke and had a PCB replaced, I acquired someone's (a certain A. H. Bell's) phone number, the phone numbers of their contacts and their Mercury number (which is a charge code giving access to an alternative network). The DF200 was able to print a list of the last 17 faxes transmitted, and most of the numbers of the 13 faxes received. I expect A. H. Bell would be interested to know all this!

I wondered why the DF200 doesn't have an erase function that the repair engineer could have used. Indeed, I had to manually go through the entire set-up and erase each item individually. In fact, there is such a function (as I later discovered) *but it is not documented.*

5 When I rang the BT service to report the DF200's fault, the first thing I was told was to write down some instructions, put the phone down, carry out the instructions, and (in a few moments) they would ring back. Why can't the DF200 print out (or fax) its system parameters while holding a phone conversation? Why are the instructions for printing out these important diagnostic information so complex they need writing down?

6 The DF200 broke again, so I had a chance to ask about the stored data from the previous customer (see above). I was told that I should have pressed [OPTIONS] [#] [9] to reset the fax. I asked why this feature was not in the manual, and I was told so that users did not activate the function *deliberately* (my emphasis).

7 The (undocumented) erase feature does not have a confirmation. It could have said (in the LCD panel): "If you want to loose everything, press START"—or some 16 character equivalent that would fit in the LCD panel. In fact, the DF200 gives you no choice.

8 After the second repair, the DF200 was left in a permanent mode where pressing [#] printed a test pattern, and pressing [*] printed an apparently unending trace (in hexadecimal and code words). These keys had this effect immediately, when the DF200 was in standby, so the features were all too easy to invoke by mistake.

BT's Helpline told me to reconfigure the DF200 to avoid this problem I should press [OPTIONS] [*] △ [#] (it then shows `CONFIGURATION 1`), then press [1] to change the first `0` to a `1`. In fact, as I found out, you should press [0]. The other digits—all zeroes and ones—are flags for automatic tracing, log printing and so forth; overall, there are 72 configurations (9 groups of 8 bits).

9 When the DF200 receives a fax and runs out of paper, it does not terminate the reception or report an error to the sending fax machine. Worse, after manual intervention to stop reception, when a new roll of paper is inserted the DF200 takes the opportunity to report the reception and transmission logs. You can explicitly request these at any time *when* they are wanted; but to take the time to print them just when you are trying to continue receiving a fax is irritating. The reception log reports the manually terminated reception as "Code 06 Printer default" which means "printer incident during *transmission*" according to the manual (p25). My emphasis.

10 When paper is inserted to prepare for transmitting a fax, the LCD changes to `A4 NORM` and it is possible to enter the number to be dialled, which is then displayed in the LCD panel. Yet is it not possible to send the fax, though, because the user should have either lifted the handset or pressed [MONITOR] *first*. Moreover, on pressing [MONITOR] and pressing [REDIAL], the fax will not redial the number just attempted. One has to start over. I make this mistake, being led down a garden path, frequently.

11 When a double-sided sheet is being transmitted, obviously two steps are required. First one side is transmitted, then the other. However, the DF200 terminates the fax connection before the sheet can be taken out and put back into the intray.

12 There are ten buttons that the user can 'program' to generate telephone numbers. The keys also have a paper legend that the user can write on as a reminder of the keys' meanings. (A nice feature is that the number transmitted changes if there is a piece of paper ready to fax; this enables the same button to be used to call someone, choosing voice or fax numbers appropriately.)

Why not permit the programming to extend to other keys than just dialling? If this was done, the user could define one key to activate the alarm, one to change the number of rings before the answerphone activates, one to print out the programmed keys' meaning (which currently requires several commands), and so on. Thus, whatever the user wanted to do frequently could be personalised and made very easy.

13 There are 16 error codes described in the manual (p.25). These codes are numbers and/or letters, and seem completely arbitrary. Why doesn't the DF200 use words for them when the errors occur? (It manages to show words like `FINE` at other times.)

14 The DF200 can define "one touch keys" so that the user need only press a single button to dial a number. It is not possible to dial a number and then have a button defined to dial it, even though there is a [REDIAL] that can redial the number immediately. (In other words, the user cannot check that a number is correct before it is defined.) The [REDIAL] button does not behave the same way as dialling a phone number; it only works when the dialling is direct to an exchange.

In summary, it is possible to store frequently used phone numbers so that they can be recalled by a single button press at a later time. It is *not* possible to store the last number dialled, even though it is displayed on the LCD (and the DF200 clearly 'knows' it). If you wish to store a number, you have to enter it, save it, then check it; it is not possible to save a number, which following a successful phone call, you know is correct.

15 I answer the DF200, and it is my wife asking if there are any messages recorded for her. The answer is, yes, for the [PLAY] LED is flashing. If I press the [PLAY] button, we can both hear the error beep of the DF200, but it is not possible to play the message. Instead, I have to find the DF200 manual, read the "programming the DF200 for remote use" section to her, and leave her to ring again, using the command sequences: *dial number*, press [#] repeatedly until DF200 beeps.

Unfortunately, my DF200 is set up with an access code; the manual does not say how to enter the code! It says, "After you have sent your access code and pressed [#] the DF200 will immediately begin to play any new

messages." (This seems to imply that the access code is entered before the [#] .)

16 The DF200 has two [STOP] keys. Apart from one having a LED indicator, I cannot see what their difference (if any) is. The potential advantage of this design is that each [STOP] key is in a group of keys that activate functions that might require stopping; one [STOP] is close to the answerphone playback keys, the other is next to the [START] key (which starts almost everything).

17 The DF200 has a paper 'tray' so that several sheets of an out-going fax can be held. Unfortunately, if there is any paper in the tray, no in-coming fax will be received.

18 The intray is not as wide as standard US paper. The paper guide only permits sheets of width 20.7 to 21.6cm. With little effort the fax could have handled much narrower paper (such as A5).

19 The DF200 will transmit faxes with headers. Whether a header is sent is an option classified as *transmission param./header transmission(with, without)* but the header itself is classified as *user param./id*—not as a header or header text.

20 The possibility of receiving faxes with and without headers is summarised in the manual but is nowhere explained. (There are several such discrepancies between the summary and the body of the manual.)

21 After pressing the [MONITOR] key, the user can hear through a loudspeaker without picking up the handset. Either dialling or pressing [REDIAL] , and the dialled number is displayed in the LCD panel. This is useful, as it gives feedback to the user what number is being dialled. However, when the handset is lifted, the LCD displays `ON LINE`—as if the user didn't know—and ceases displaying the number dialled.

22 The DF200 can print status and other information on its paper. However it leaves large gaps (around 5.5cm) between separate items. This wastes the expensive fax paper.

23 Although the standby mode continuously displays the date and time in the LCD, the time display is not available when the DF200 is doing anything else. Thus, when using the phone to hold an expensive international conversation, it is not possible to either see the duration of the call or even to see the current time. The LCD simply shows the number dialled or, if the handset is picked up (after using the 'hands free' dialling), the essentially useless text `ON LINE`.

### *Inconsistencies*

24 The DF200 has remote facilities, so that a user may phone it and listen to recorded messages, and perform a few other functions. The user interface for remote facilities has no relationship (so far as I can see) to the normal user interface to the same facilities. Thus the remote user not only has

none of the appropriate button labels to help, no LCD display for confirmatory feedback, but an arbitrary key code to remember.

For example, although the DF200 has a button [#] (which behaves as a command to a remote exchange or as a space in a identifier in a DF200 key definition) the effect of pressing it on a remote phone is the same as [STOP] on the DF200.

The following table compares all remote commands with their equivalent DF200 direct commands, where there are equivalents.

| Remote key | Meaning | Equivalent operation on DF200 |
|---|---|---|
| [1] | Listen to a message again. | Press ◁◁ |
| [2] | Delete a message. | Listening to message deletes it unless [SAVE] is pressed. |
| [3] | Go on to next message | Press ▷▷ |
| [4] | Listen to previous (*sic*) message again. | Press ◁◁ twice. |
| [5] | Listen to all messages, new and old. | [PLAY] |
| | [5] pressed again, "defers the playing"; and pressed again, "allows to go on playing messages" | |
| [6] | Does nothing. | |
| [7] | Answer on. | (complex process) |
| [8] | Record a new announcement. (Note that there is no way to confirm the recorded announcement.) | (complex process) |
| [9] | Answer off. | (complex process) |
| [#] | "Note: Pressing [#] (STOP) allows to stop any operation." | Press [STOP] |

25 When setting up the DF200, the name and number of the FAX can be defined. The number of the FAX can include spaces (by pressing [#] ) and + (by pressing [✳] ) as well as the usual ten digits. It isn't possible to have a standard international number such as "+44 (0) 181 363 6411" because there are no brackets available. In contrast, the name (called the "ID" by the DF200) *can* have a range of letters and punctuation. In the setting-ID mode, [1] now provides punctuation—including a space as previously

provided by [#] —whereas [*] now beeps and does not produce even a + sign!

26 The undocumented [OPTIONS] [#] feature causes ERASE MODE to be displayed in the LCD panel. Yet [OPTIONS] [#] [5] prints a trace of the DF200 (which I cannot interpret, it being mostly hexadecimal numbers). This trace feature has nothing to do with erase mode.

27 It is possible to enter an invalid hour such as 29:00 (which the DF200 will accept and treat as the current time), but it is not possible to set an invalid minute, such as 12:69. Hours and minutes are obviously validated in different ways.

28 There is a button [MONITOR]; if the handset is down and this button is pressed its LED lights. The effect is as if the handset is raised, but the loudspeaker replaces the earphone. The feature supports "hands free dialling." [MONITOR] may be pressed when the handset is actually lifted, but rather than amplifying the earphone to the loudspeaker, one merely achieves clicks and beeps. One of the potential monitoring functions, enabling more than one person to listen to a phone conversation, is therefore not possible.

29 Despite the button [MONITOR], which appears to give the user a choice (monitor or not monitor), *all* incoming calls answered by the answerphone are monitored. It is not possible to set the DF200 so that it silently answers calls. For example, it is not possible to stop anyone in the same room hearing the phone message being recorded.

30 If paper is placed in the in-tray, the LCD displays A4 NORM. If [MONITOR] then [REDIAL] are pressed, the DF200 dials the last number used and sends the fax. However, if [REDIAL] is pressed before [MONITOR], the DF200 enters an undocumented mode called TRANS. PARAM. and then asks for the number of pages, correction mode (whatever that is), transmit speed—and then does nothing.

31 The DF200 has eight digit keys with three to four letters ([2]: ABC; [3]: DEF; [4]: GHI; [5]: JKL; [6]: MNO; [7]: PQRS; [8]: TUV; [9]: WXYZ). Digit [0] generates no special symbols.

The user enters an ID by pressing digits and [#] to move to the next column, though this extra meaning of [#] is not shown on the button.[5] In other modes, [#] behaves like a digit (it is a code that is transmitted to exchanges) and pressing a number automatically causes the position to advance.

Digit [1] is not marked as generating any alternate characters, but in fact it generates 11 symbols (/ - space + ; : , . ' and brackets, ( )); the manual is not specific. Why space when this might have been inserted with [#] (in fact [#] now moves right, which only sometimes has the effect of

5 Pressing [2] [2] [#] [#] [2] [2] [2] [#] [2] [2] [#] [8] [8] enters A BAT.

introducing a space)? Why not put some of the characters in [0], which does nothing (other than insert 0).

32 Star ([*]) moves the cursor left without deleting, and [#] moves right. The symbols don't have any natural association with left or right motion (though the [*] is to the left of the [#] key).

33 To delete a symbol from the LCD (when permitted) the decrease volume button (which is not labelled as supporting this purpose) is used. Decrease volume, [VOL] ▽, deletes the last symbol and moves left, but [VOL] △) does nothing other than beep. *Except* that if there are characters to the right of the entry position (the cursor), then [VOL] ▽ deletes all characters to the right and does not move left!

34 The button [PAUSE] normally introduces a pause in recorded dialling sequences, in fax speed setting mode, however, it is used to protect—in an unspecified way—the fax from echoes; whether a user would be likely to remember this hidden solution to an infrequent problem when it was needed is another matter.

35 The DF200 can transmit faxes at four baud rates but can only receive at three (it can transmit but cannot receive at 7200 baud).

36 When the handset is down, it is possible to key a number (which is displayed in the LCD panel but is *discarded* when the handset is lifted). However in the same situation, the one-key abbreviations beep (i.e., it is an error to press them), when they could—more consistently—have shown the number they would have dialled out had the handset been lifted.

37 Without performing the 'print' command (which wastes paper, and prints *all* one-key abbreviations) it is not possible to determine an abbreviation's full number without actually dialling out.

38 After the answerphone records a message (either from a caller, or from the user) the LED on the [PLAY] button flashes. When [PLAY] is pressed, any recorded messages are heard, and the LED stops flashing when all have been listened to (or skipped). If a new message now arrives, earlier recordings will be erased unless [SAVE] is pressed. When [SAVE] has been pressed subsequent messages are appended to the list of recordings, however *the LED does not flash* in this case. Thus there may be recordings, whether the LED flashes or not.

Imagine recording a message and checking it sounds OK. Doing so stops the LED flashing, and thus whoever is supposed to take the message will not know one has arrived.

39 When the answerphone has received a message and recorded it, the LED in the [PLAY] button flashes. Flashing indicates that there are one or more recorded messages, which will be played back when the [PLAY] button is pressed. When all messages have been played, there is a final tone and the LED stops flashing and pressing [PLAY] has no further effect (unless

[SAVE] is pressed). There is a problem: if [PLAY] is pressed before the tone finishes, the LED continues flashing indefinitely and pressing [PLAY] shows `MESSAGE No 1` in the LCD (so there is apparently a message) but no message is played back. This gives the impression that the user's actions have erased the message (there is a message, but it is cleared); or, if some other user was to use the fax, it gives the impression that someone rang but left no message. Neither of these interpretations are correct. It is simply a DF200 bug.

40 The manual describes what the LCD displays. It is rarely accurate, (perhaps) partly because the LCD itself is short, and often shows abbreviated words.

41 (The inconsistency between the DF200 and its manual is mentioned again for completeness.)

### *General inadequacies*

42 The DF200 and its external power supply overheats; at standby it consumes 6.25W—it gets rather hot for a user to place it on their lap.

43 The DF200 has a Ringer Equivalence Number (REN) of 1.5, when an ordinary telephone (without benefit of a separate power supply) has a REN of 1. A BT line is limited to a total REN load of 4.

44 The paper support is flimsy and doesn't fold flat, so it and its flimsy hinges can break easily.

45 The DF200 clicks despite digital recording technology. The handset microphone distractingly picks up the sound of breathing.

46 We often find the DF200 shows that it has recorded a message, but which in fact is null.

47 The FAX transmits images on paper placed *face down* in the slot. So if the sheet to be sent has the recipient's FAX number on it, it cannot be read! A fax should transmit images face up; and the DF200 should have *some* reminder on it that the image should be sent face down.

48 The DF200 picks up radio interference.

49 All the button labels are in capitals (apart from fast forward and rewind, which use abstract symbols; volume increase and decrease are called [VOL] ▽ and [VOL] △). This may give the fax a 'technical' appearance, but most evidence suggests that lower case lettering is easier to read.

50 The button [RECALL] (a name confusingly like the button [REDIAL]) has no meaning except when connected to a private switchboard.

51 The DF200 provides three commands that have no functionality *at all* ([PRINT], [DELETE] and [PERFORM])—the manual does not define these commands. BT's phone helpline said they had never before been asked by

a user what they do. BT said they are “software built in the machine which it doesn’t provide” [*sic*]. I think they may be a relic from the related but more sophisticated VF800 fax.

52 There are two buttons, [REDIAL] and [RECALL]. The manual does not specify [RECALL] beyond saying it activates unspecified functions from a local switchboard; thus the button [RECALL] does nothing on my DF200, which is connected to a public exchange.

53 To protect the fax against echoes, press [PAUSE] when setting the transmission speed. The key has no legend to suggest this alternate purpose, and the fax’s display does not show even the possibility of the option. Given that protecting the fax against echoes is likely to be an infrequent action—but one that is occasionally necessary—we can assume the user does not remember how to do it without prompting; the DF200 can only be said, then, to conceal this feature.

54 The volume can be increased beyond the loudspeaker’s power rating, as can be verified by increasing so that ‘bleep’ tone distorts.

55 When paper is placed in the FAX slot, the LCD shows NORM. This does not mean “ready for normal FAX transmit,” but “normal resolution” (as opposed to GRAY or FINE). When the FAX is being transmitted, the DF200 says NORM; NORM NORM; or TRANSMIS. (including the dot) alternately. Is there any difference between NORM and NORM NORM?

After one occasion my DF200 was repaired these LCD displays changed to A4 NORM etc.—even when the paper being transmitted was not A4! Of course, it is possible that my repaired PCB was a recycled older one, and later design revision removed the superfluous and often incorrect ‘A4’ part of the message. Whatever the reason, there are still peculiar design choices that seem inexplicable—the more so, given that the differences between my original and repaired DF200 *prove* that the manufacturers can modify the design—yet whichever is the more recent design can’t be said to be better. Why doesn’t it say TRANSMITTING, as the LCD is long enough?

56 An option allows the user to switch off FAX transmission status reports. However, despite switching it off, the DF200 will still provide summaries are a curious mixture of ambiguity, jargon, and unnecessary abbreviations. (It also provided the report after 7 transmissions, which seems an odd number to me!)

57 Although the DF200 can clearly print “Correct” and “Manual Call” and so on, the status column occasionally says “Code 03” or “Code 07.” Why can’t it put these codes into words too?

58 When the report says “subscriber has stopped”—does that mean us or them?

59 What does ‘Disconnected 210E000000’ mean—a code that has occurred during normal use of the DF200?

60 The sequence [OPTIONS] [✳] [#] accesses a list of editable configuration numbers. Neither the command sequence nor the meaning of the configuration numbers is defined.