DESIGNING SOFTWARE TO HIT THE MARK: PROBLEMS AND PROSPECTS, TOOLS AND TECHNIQUES

I.A. Clark

Rapporteur: Mr. M.J. Elphick

Abstract:

Designing software for a mass market is not like bespoke tailoring. The designer has little control over who uses his product and for what. But if the software has no impact on the user's job, it is doing nothing for him. Conversely, software which helps someone do his job better may change the job out of all recognition. Ideal task support is a moving target.

It has been said that computers make hard things easy, and easy things hard. The substance behind this morsel of modern folk-wisdom is that software which does not support all the tasks the user wishes to do, or does so in an unfamiliar or confusing manner, irritates, thwarts or otherwise distresses the user, which isn't good for business. In any event, no designer sets out to write software which is hard to use. Most have a sincere desire to give satisfaction, and a painful awareness of how hard this is to achieve in practice.

Research into the human factors of the software interface has furnished the designer with a wide variety of actual and potential tools to judge before it is too late whether he is hitting the mark. Some of these will be reviewed, along with the prospects for their further development.

THE DESIGN PROCESS

Cognitive Engineers do not exist as a recognised profession yet. I believe the computer industry needs them, just as much as it needs lawyers. Perhaps the theoretical foundations for the necessary training will be laid in a series of seminars like this one. The theoretical side will be your business, to which I fear I can contribute very little that is original.

However, at least I can try to map out the domain of their professional competence, as it exists in the world that I know best: the development of innovative software within IBM.

Firstly, a glossary. There is only one word in it that matters, a word that flummoxes unregenerate computer professionals whenever it is flung at them (which is surely a case of the biter bit!):

COGNITIVE

Webster's dictionary defines this as meaning:

"...capable of being reduced to empirical factual knowledge."

In other words, to do with 'knowing'. People who use the word in a computer context use it to mean anything to do with human information processing. If, for example, I am using a new software system on a familiar terminal, I have to do a lot of information processing in my head to reduce what I see on the screen to the everyday things I know.

Sometimes I have to do more information processing than I feel I ought, because the designer has chosen his keywords badly, or tempted me to draw the wrong conclusions about his structures. That's the heart of the matter.

So I blame the salesman for selling me a load of junk. The salesman blames the designer for clowning around with the interface. The designer blames the psychologist for not telling him in appropriate terms how people's minds work. All of us blame the Cognitive Engineer for not existing.

Perhaps we should examine the design process itself, to see how we might use one of these paragons if we had one.

Figure 1 is perhaps an idealised view of how software should be designed. However, some of IBM's best-selling software started life this way, notably IMS and CICS.

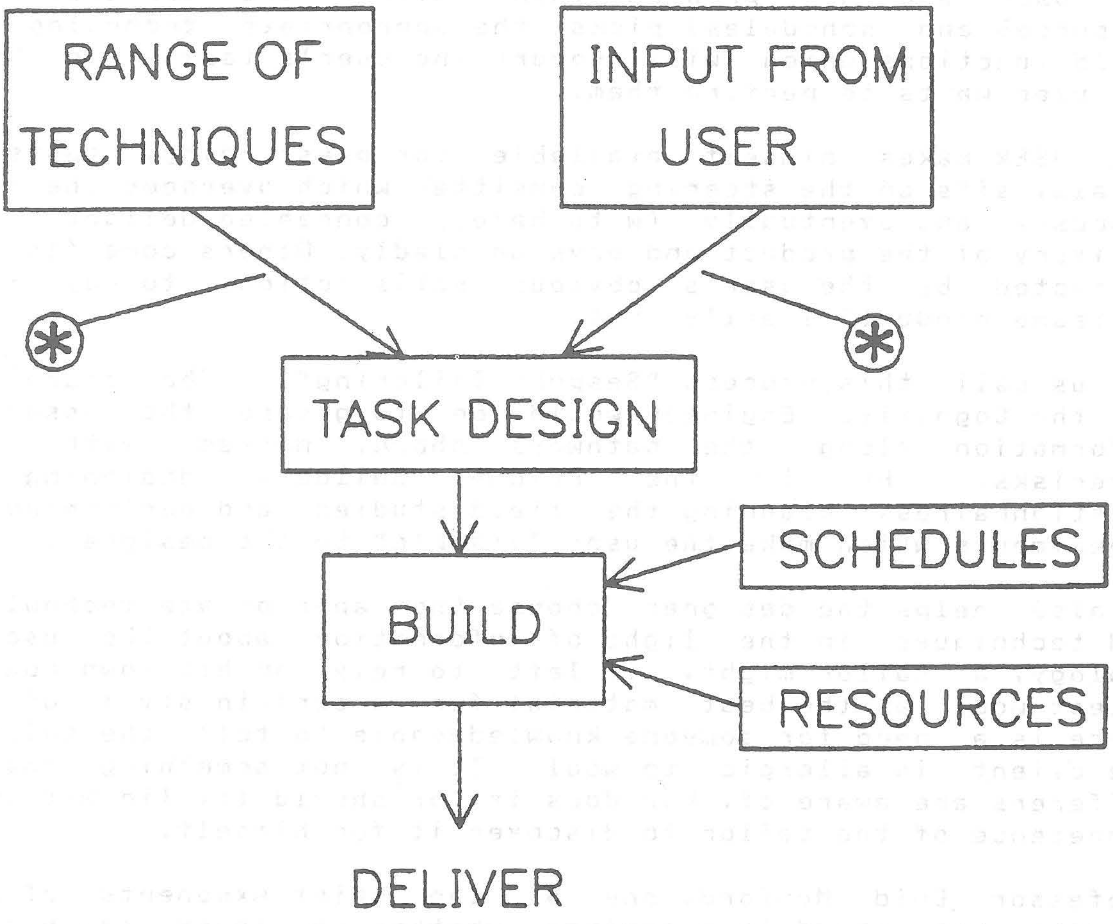

Fig. 1. The "Bespoke Tailoring" process of Software Design

An individual or composite person (masculine, by default), called the USER, furnishes input to another such person, called the DESIGNER, who has at his disposal a repertory of techniques and technologies. In a company like IBM this repertory is both large and accessible, but it still requires breadth of knowledge and open-mindedness on the part of the designer to wield all this power effectively.

The DESIGNER endeavours to achieve a thorough understanding of the user and his problem, then (within the constraints of resources and schedules) picks the appropriate technologies to build functions which will support the user's tasks, in the way the user wants to perform them.

The USER makes himself available for questioning, tests and trials, sits on the steering committee which oversees the design process, and eventually (with barely concealed delight) takes delivery of the product and pays up gladly. Others come flocking, attracted by the user's obvious satisfaction, to buy - the selfsame product? Ideally, not.

Let us call this process "Bespoke Tailoring". The natural role of the Cognitive Engineer would be to grease the passage of information along the pathways shown, marked with ringed asterisks. He is the Bridge Builder, designing the questionnaires, planning the field studies and performing the experiments which make the user "visible" to the designer.

He also helps the designer choose the appropriate technologies and techniques in the light of information about the user. By analogy, a tailor might, if left to rely on his own counsel, select wool as the best material for a certain style of suit. There is a need for someone knowledgeable to tell the tailor if the client is allergic to wool. It is not something that all sufferers are aware of. Nor does it, or should it, lie within the competence of the tailor to discover it for himself.

Professor Enid Mumford, one of the chief exponents of this process, was asked in a seminar whether she found the hardware and software vendors particularly helpful. She replied to the effect that vendors were far too eager to sell her clients their own pet solutions, rather than expend the effort to understand her clients' needs.

I have heard similar things said about drug companies, especially the so-called "non-ethical" pharmaceutical firms.

So it seems that the real state of affairs is as illustrated in Figure 2. In the light of the above, perhaps we ought to call it the "Patent Medicine" process. However, pursuing the tailoring analogy, we shall call it the "Off-the-Peg" process instead.

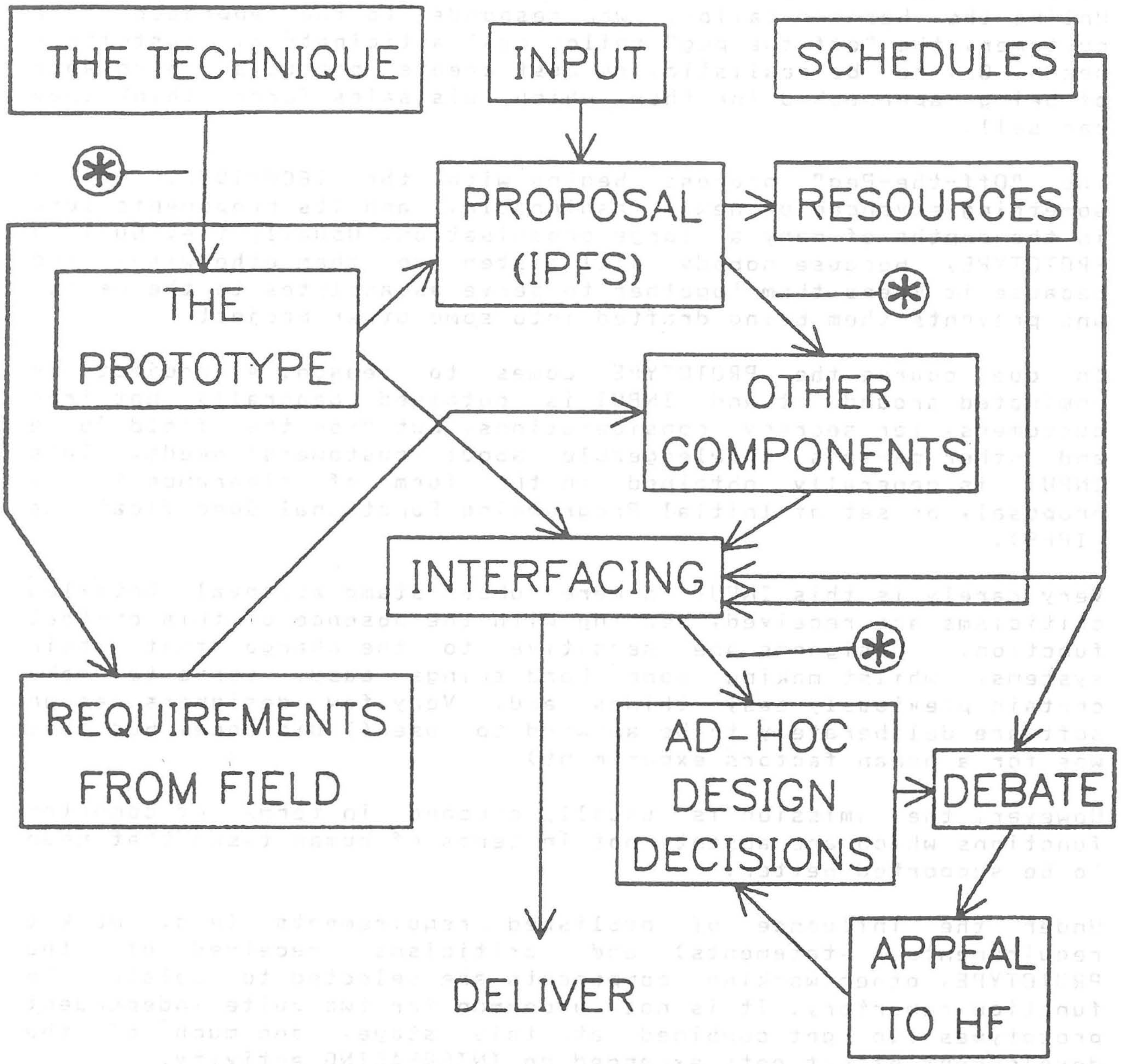

Fig. 2. The "Off-the-Peg" process of Software Design

Before we condemn it, we should note that it is perhaps the only way in which an organisation can create new products for a mass market, when it is expected to do so on its own initiative. Unlike the bespoke tailor, who responds to the approach of a customer, the "off the peg" tailor must anticipate his customer's needs. Or, to be realistic, he must create products, in advance of being approached for them, which his sales force think they can sell.

The "Off-the-Peg" process begins with the TECHNIQUE. It is something advanced or newly fashionable, and its proponents lurk in the depths of many a large organisation. Usually they build a PROTOTYPE, because nobody will listen to them otherwise, and because it keeps them together to serve as acolytes to the beast, and prevents them being drafted into some other project.

In due course the PROTOTYPE comes to season. A project is nominated around it and INPUT is obtained, generally not from customers, for secrecy considerations, but from the field force and other persons knowledgeable about customers' needs. This INPUT is generally obtained in the form of clearance for a proposal, or set of Initial Programming Functional Specifications (IPFS).

Very rarely is this INPUT a mere rubber-stamp approval. Detailed criticisms are received, dealing with the absence of this-or-that function. Designers are sensitive to the charge that their systems, whilst making some hard things easy, serve to make certain previously easy things hard. Very few designers design software deliberately to be awkward to use (I did once, but that was for a human factors experiment).

However, the omission is usually couched in terms of computer functions which are absent, not in terms of human tasks that need to be supported better.

Under the influence of published requirements (e.g. market requirements statements) and criticisms received of the PROTOTYPE, other working components are selected to bolster the function repertory. It is not uncommon for two quite independent prototypes to get combined at this stage, and much of the development effort gets expended on INTERFACING activity.

Internal interfaces take priority over the human interfaces, since it is of utmost important to the developers to deliver code that works correctly, even if this means that a few rough edges remain in the human interface. Later releases of the product can always smooth these over.

Such is the practice, if not the theory, by which much bread-and-butter software gets designed. Who is without sin enough to cast the first stone?

Granted that this is so, and that a company may have the best of reasons for operating in this way, what is the scope for a Cognitive Engineer? Probably none. This system has arisen in a milieu which until recently has lacked the services of human factors professionals to grease the pathways of Figure 2.

There is one notable exception. In the course of the INTERFACING activity, ad-hoc design decisions may be made. Sometimes there is debate, which the developers do not feel themselves competent to resolve, since it means trying to predict what they see as pure customer preferences. In that case, appeal is made to the Human Factors Department, if they have one, or to some outside ergonomic consultant.

A glance through the published reports of most Human Factors Laboratories, inside IBM and out, show that this forms the staple diet of their work, namely to resolve arguments among the developers.

This is not the sort of role I have in mind for the up-and-coming Cognitive Engineer. His is to be a more central role in the design process. However, were he to assume this role, it would very soon alter the design process as I've described it. For the better, I would hope.

From the evidence of many projects the author has known, the following pathways in Figure 2 would seem to benefit from his attention.

- from TECHNIQUE to PROTOTYPE. Many working prototypes suffer from bad human factors, because all that sort of thing fell outside the interest of its designers at the time. Or worse, the designers spent all their effort on getting a particular function to work, without first finding out whether it was the RIGHT function to be implementing, in the context of its likely usage. I have seen great efforts expended on functions to move rectangular areas over a screen, when all the time the intended users thought in terms of circles. A clear case of the wrong functional "peg" being fitted in the task "hole".

- from PROPOSAL to OTHER WORKING COMPONENTS. Sometimes the other components of the product offering are chosen on the basis of the functions they contain, not the compatibility of their interfaces, or the preferred user view of their data structures. Think, for instance, of the problems which can arise if a subsystem written in PL/I is tightly interfaced with one written in APL. (Suppose the user has to move an array of variable length character strings from one subsystem to the other).

from INTERFACING to AD-HOC DESIGN DECISIONS. The Cognitive Engineer has a part to play here, even (and especially) if the developers do not disagree among themselves. Such decisions are invariably made in a hurry, yet sometimes have a human factors impact out of all proportion to the effort spent on making them.

INVESTIGATION V. DESIGN

We turn now to the tools and techniques we expect our Cognitive Engineer to employ.

When one thinks of Human Factors, one thinks of Behavioural Science, and the sort of meticulous laboratory experiments its practitioners perform. The evidence obtained from such experiments is statistical: a hypothesised relationship between two or more measurables is either established or discounted. Many subjects are employed, and any psychological variable which might conceivably be thought to introduce stray variance is either fixed, controlled by statistical design, or excluded by means of collateral experiments. If you're lucky, the researcher reports back in six months.

When designing a product, are such experiments worth the effort? No. Why, then, should anyone be interested in them in this regard?

For a start, they promise to quantify such variables as productivity, which in principle can be translated into money saved by the customer as a result of buying your product. It would be worth any amount of academic research to establish the cash nexus in such cases, like the value of colour, the value of interactive computing, or the value of installing terminals conforming to the German DIN standards.

Unfortunately, statistical evidence for productivity gains requires an enormous number of subjects, if what is being measured is time to complete a given task. Sackman estimated a 30:1 ratio between the ability of the best and the worst in the working population, when using interactive computers. Moreover, whereas this sort of experiment can establish connections, it does not uncover the mechanisms, unless they can be guessed in advance and tested for.

This is no good to the designer, who needs to know why something is going to go wrong, not merely whether it is or not.

Moreover, the quality of information which a reductionist experiment delivers is all wrong for consumption by a designer. It is abstracted from the real world situation, a necessary thing to control variance, but to the non-psychologist this looks like a divorce from reality.

Moreover statistics is, by its very nature, a forensic tool. It aids debate. Scientific debate, of course, but designers are not scientists, and their schedules generally do not permit the luxury of debate. They thank their advisers for a quick and tidy answer.

This is not to decry reductionist experiments - in their place. One of their tangible benefits is the fall-out in tools and techniques from the creative discipline of designing a tightly controlled experiment. I know of a tutorial technique invented in an attempt to control variance in the pre-test training of experimental subjects, which subsequently proved to be a superior training aid in its own right.

Far more promising as a design aid is detailed observation of just a few subjects. But even though designers respond far more positively to such "anecdotal" evidence, how can a serious investigator justify doing this?

Let us say right from the start that timing measurements on a handful of subjects are unreliable estimators of productivity gains to be achieved by the whole user population. So many things can happen in the course of an experiment to prevent a subject completing a task, or to lead him down the garden path. As Sackman says, people vary so much in their performance.

However (and this is an empirical observation, yet to be established scientifically), people do tend to fall into the same traps. Novices spend a long time climbing out. Experts barely falter, but falter they do. They falter over much the same things as novices, although it takes quite powerful measuring equipment, video cameras and so forth, to catch them at it. Moreover they often recover unconsciously, and are actually unaware of the error they made. Much expertise seems to consist in fast, unconscious recovery from error rather than avoiding it in the first place.

Very few trials are necessary to establish the existence of a strong phenomenon, whereas a large number are necessary to establish the existence, or non-existence, of a weak one.

There is an analogy here with telescopes. You do not need a very powerful telescope to see Sirius, indeed the naked eye will do. But to establish whether or not it is a binary star needs an extremely wide telescope, to gather much more starlight than does a simple pair of binoculars. What you achieve is not magnification, but discrimination.

It appears then, that the most fruitful ways of designing more usable software is not to look for productivity gains directly, but to observe the errors and difficulties which users experience with a given offering, since these represent stumbling blocks to ideal usage.

DETAILED OBSERVATION

THE "FULL-FRONTAL" APPROACH

In IBM's Atlanta Documentation Test Center, and Boca Raton Human Factors Laboratory, they pursue what I shall call the "Full Frontal" approach - total activity analysis. The aim is to detect, record and subsequently analyse "critical incidents", by which I mean errors and difficulties encountered by the subject in attempting to use a given product.

The method works best when a working prototype of the product is available, for example a new desk-top computer for use in small businesses by non-DP trained professionals. Such a product usually includes a training package for first-time users, otherwise a task sheet would be provided. The experiment entails closely watching a novice user as he or she works through the instructions, trying to use the machine.

The test cell resembles nothing so much as a television studio, with closed-circuit TV cameras dotted around the ceiling. Observers taking written notes sit behind one-way glass panels so as to relieve the subjects of the feeling of someone breathing down their necks.

This process results in a video tape consisting of a side view of the subject, an inset showing the time-of-day, and, superimposed over all, an image of what is currently on the subject's screen. The sound track contains not only all the subject's utterances, groans and clucks of frustration, but the content of any calls for help.

IBM users in the USA are given a toll-free number (a so-called "800" number) to phone if they need help using the system. This is usually simulated in the experiment by placing a telephone beside the subject, which connects directly through to the experimenters behind the one-way mirrors.

Such videos are intriguing, sometimes hilarious to watch, and convey their message to the developers with impact. However the process is expensive in terms of equipment and in people's time. Subjects are cheap enough, thanks to secretarial agencies, but trained ergonomists are needed as observers.

THE DUDS APPROACH.

At the other end of the scale of expense is what I shall call the DUDS approach, after the "Documenting Users' Difficulties with Systems" (DUDS) experiment of Hammond et al. [5]. This has since been adapted in various forms to test IBM products undergoing development.

Like the foregoing, this uses a working prototype of the system being studied. Video recording is not used, so the experiment can be conducted at a customer's premises upon an installed system, as indeed [5] was.

One subject is observed face-to-face by two experimenters, as he or she works through a task sheet. The subject is prompted from time to time by the experimenters to articulate his or her thought processes, and a voice recording of the conversation is made and subsequently typed up.

The transcript is interspersed with the text of the computer transactions, or by screen images as appropriate, these being reconstituted from a log produced by the computer. An error classification and analysis is added in the left-hand column, and the resulting detailed protocol serves as a vehicle for recommending design changes.

The data is cheap to collect, but without specialised aids the protocol can be expensive and time-consuming to produce.

The problems arise with synchronising the voice log with the machine log, and portraying the subject's input and the computer response in a form which enables the subject's difficulties to be appreciated. This is something which the "Full-Frontal" video recording technique handles naturally, but a protocol in book form does have its advantages.

THE CONTROLLED APPROACH

Earlier I said that a tightly-controlled laboratory experiment in the best reductionist tradition had little to commend it in the timescale of a development project. However, such experiments, [1, 4], do suggest useful techniques for more pragmatic trials.

In contrast to the wider-ranging activity described earlier, the activity of subjects in a controlled experiment is best confined to the subtasks of special interest. This implies that there have been wider-ranging experiments already conducted, which have highlighted certain areas as meriting deeper investigation.

Rather than giving the subject a task sheet to work through, it is easier to control the experiment if the computer dialogue itself drives the subject, prompting him or her what to do next.

One of the casualties of this process is realism - or is it? Psychologists term abstracted tasks of this nature "paradigms". If the paradigm is studied for its own sake, then yes. This point is made strongly by no less a figure than Alphonse Chapanis, [9], who challenges the relevance of much impeccable psychological experimentation to anything in the real world at all.

However if the paradigm is kept in context, which is warranted by its having been observed as a live issue in realistic situations, then realism need not be a casualty. The controlled experiment takes its place as just one of a battery of coordinated techniques, having the special power of paring away contaminating influences from a single issue, enabling its reality to be observed directly. This view is implicit in the work of Barnard, Hammond and others, [3, 7].

One of the features of a controlled experiment is that a simulation, or mock-up, of the system being investigated is more appropriate than a working prototype. Perhaps it is no coincidence that the experimental task comes more and more to resemble a rat-maze as one attempts to bring the subject's activity under control. Eventually, for any given screen image that can be shown, the subject has a limited choice of options open to him or her. It becomes possible (indeed it becomes strongly advisable) to draw a state-diagram, where each node represents a state of the system, and each edge joining the nodes represents a particular choice of subject's response. The maze-like nature of the diagram becomes immediately apparent.

Indeed, with a working prototype, it is hard to stop the subject driving the system into ill-defined states and perhaps crashing the system. With a simulation it is easy. You simply do not define any states except those you are interested in. All responses which the subject makes which are un-catered for, simply lead to a notice saying "you weren't meant to do that. Press any key to go back to where you were".

Some experiments I have been involved with have ended up almost as a linear sequence of screen images, like a film strip. Yet, strange to say, the subjects never reported any feeling of being redundant. Rather they retained an impression of being in control, at least of the machine, if not of the experiment.

As a bonus, mock-ups can be produced and altered quickly, and so are available for experiments much sooner than would be a working prototype. They can be altered from one experiment to the next, and have a habit of behaving themselves. Prototypes tend to be rather ramshackle, a "hazy maze" one might say, and are prone to misbehave badly if the slightest change is made.

To take [1] as an example of a controlled experiment, this employed a solitary subject per session, with minimal intervention by the experimenter. No voice recording was made. The following data was collected:

- all HELP invocation. More than one level of HELP was provided, which gave some indication of what use was being made of the facility, viz. to remind oneself of a command name, or to discover precisely how to use it.

- errors made in invoking a command,

- time intervals, viz.

  - from screen image appearing to the commencement of keying

  - the duration of keying

  - from the termination of keying, viz. by pressing a key marked ENTER or RETURN, which signalled completion of the invocation, to the next screen image appearing.

The controlled approach is never so well controlled that no unexplained variance creeps in. Certain things are hard to control: in fact they would serve as splendid research topics for postgraduate trainee Cognitive Engineers.

Firstly there is the problem of training the user reliably to perform the experimental task. This may range from formal face-to-face lessons in a class, through miniature computer-aided learning courses, to merely designating the first few experimental trials as the "training phase".

The tendency of subjects to ask different questions, the desire of the experimenter to conceal from the subject what is being measured, and the strong influence of a stray word, tends to make face-to-face instruction an ill-controlled process. Experimenters are inclined to prefer the clinical administration of fact-sheets in complete silence. To some people, this is a wholly unfamiliar mode of learning. Perhaps they never discover anything new but someone has to tell them about it verbally.

Even when subjects can be persuaded to use documents to learn about a system, it is difficult to monitor the use of the document. How do they scan the pages? Do they attempt to read the whole thing from start to finish? Do they use the index? - the table of contents? Do they skim through looking for words suggestive of the problem they are trying to solve? Do they merely use a manual to confirm what they have guessed already?

Then there is the statistical nature of the study and its findings, and how best to report these. The scientific press is familiar with this sort of material, but few designers can make direct use of it. More likely they will summon the experimenter and beg him to tell them in words of one syllable what it all really means. Instantly the experimenter forsakes the world of statistical truth and enters that of subjective impression.

THE BARELY-CONTROLLED APPROACH

Experiments like [1] have nonetheless furnished us with useful tools for quick-and-dirty investigations. Here the measurement tools are used the same way thermometers and voltmeters are employed in engineering labs: not to quantify scientific truth, but to provide an objective reference point to support a sensory impression. The matter is discussed in [2]. What happens is as follows. The investigator says to himself something like "I think that's done the trick". Then, if he is cautious, he follows that by "Let's see what device XYZ has to say about it. Does it support my good impression or doesn't it?"

This is non-scientific activity, nay, it is magical. Rational science has a part to play, however, in investigating what faith we may place in device XYZ, used in this way. Remember all the time that we are trying to forecast the problems that are likely to arise with a product, once it hits the marketplace. The proof of the pudding is in the eating.

The process, as described in [2], starts with a simulation of the system under test. This is actually altered from trial to trial, in an attempt to correct usability defects which become apparent as subjects try to pursue a set task with the simulation. The "t" statistic can be used as a crude thermometer (amid howls of protest from statisticians, who demand Gaussian behaviour in the underlying variable) to check whether the corrective action is doing any good or not.

Rather than one solitary subject, two subjects can work wonders. One sits down at the terminal, the other does back-seat driving. The resulting flow of dialogue is recorded, in synchrony with the terminal keystrokes. With suitable apparatus this can be played back in real time. It is perhaps not as compelling as the video produced by the "Full-Frontal" method, but it is compelling nonetheless. Sitting in the subject's very seat, watching the cursor move across the screen, is perhaps the nearest one can get to seeing the task through the subject's eyes. If the experimenter can contrive to leave the room, too, the voice tape becomes even richer in diagnostic information, though usually far from flattering to the designer's ears.

FORMAL TOOLS

Human Factors experiments are expensive in terms of time and equipment. However they don't entail any great intellectual challenge, unless one or the other is short.

Often in the course of investigations, experimenters draw up flow diagrams to represent the man-computer interaction. In studying these at leisure, they often see counterparts to the areas of subjects' difficulties. Somehow it seems that complexity in the diagram can be related to complexity in the user's task. This suggests that analytic techniques are possible to predict the empirical findings of experiment, possibly allowing us to dispense with much expensive practical investigation.

Now of course a flow diagram of the interplay between man and computer is only a finite-state machine description, which can be considered to parse the "language" of the transactions which pass between them. The fact that there are two emitters of messages, not one, does not matter. They emit alternately, and together can be treated as a single message-uttering system, whose utterances consist of well-formed formulae of the basic type:

[human] [computer] [human] [computer]...

So in principle its syntax is amenable to treatment by the theory of formal grammars, which are more familiar in the sphere of programming languages. Various investigators are trying this approach, notably Moran, with his Command Language Grammar (CLG) and Reisner, with her Bachus-Naur Form (BNF)-based Action Language. Both claim to be able to handle not only physical actions by the user, but also cognitive actions, and to predict not only performance, e.g. time to complete a task in the absence of errors, but also detect the possibility of errors and predict what form these are likely to take.

The man-machine function diagrams of Singleton [10], and other similar flow schematics, are also grammars in a sense, being the graphic counterpart of some linear grammar. Timing estimates and the possibility of error can also be guessed from them to some extent, in ways yet to be fully formalised.

Students of programming languages distinguish between the "syntax" and the "semantics" (meaning) of a language, although more powerful tools such as attribute grammars are being claimed to lay bare the deeper structure of sentences.

A simple grammar, such as BNF as it is usually employed, is "superficial", it allows one to generate well-formed sentences, but does not promise to say that much about why a user falters over a given construct. This problem has been addressed by Morton et al. [6], using the notion of interaction between different domains of knowledge. Thus the domain of natural language may interfere with the proper usage of an interactive command language if the latter's terms are badly chosen.

Subsequent work by the same research group has come up with the notion of "knowledge fragments". A novice user, it appears, learns how to do certain things as a growing collection of largely disconnected recipes and caveats, which they bring into play in given circumstances. Even if a knowledge fragment is "wrong", however, it is rarely discarded, but overlaid with more refined fragments to deal with the awkward cases.

To take an example from a child learning to spell English, he or she may be given a rule:

"'i' before 'e'"

and because this is not universally so, immediately afterwards comes:

"except after 'c'"

This will still fail in some circumstances, so a further rule may be added:

"Don't forget, please, the little word 'seize'"

This is still not universally true, but it suffices for most purposes. According to Barnard and Hammond, [8], many errors arise because of competing knowledge fragments, so that the wrong fragment is likely to present itself when called upon.

It remains to combine this theory, if true, with formal syntax analysis.

It seems, then, that formal grammars offer a challenging theoretical foundation on which to advance towards a sound discipline of Cognitive Engineering. The short-term aim is to limit recourse to costly behavioural experiments, but this is often achieved today by simply not doing the experiments and hoping for the best. On the other hand, patching up the human factors of a product by issuing further releases can be viewed as performing only the most costly and inefficient sort of experiments.

If formal methods are to prove a viable substitute for experiment, then they must be able to predict those serious usability defects, the "show stoppers", which experiments are capable of catching. As we have said, the object of the exercise is not to achieve some comfortable academic vantage point from which to admire the view, but to eliminate nasty surprises in the field. Formal analytic methods are only cheaper if they can be relied on to do this.

Nevertheless, it is hard to see how some of the facts obtained by experiment could have possibly been achieved analytically, as viewing some of the video sequences produced by the "Full-Frontal" approach shows. Perhaps some experimentation will always be needed.

PROSPECTS.

In recent years IBM developers have witnessed a greater concentration on experiment, the more rigorous the better, to assure the quality of their products as regard to human factors. Much expenditure has taken place in recent years, the goal being to achieve excellent resources to detect and solve usability problems empirically. Total Activity studios have appeared in several locations in the USA and in Europe, and we can look forward to the time when the testing of documents which form an important part of software products is routinely as thorough as machine-code testing.

Studies of developers themselves, and the development process, have been conducted to determine how best to proffer usability data, so that its lessons are incorporated earlier rather than later into the design of a new product.

I anticipate that much the same sort of movement is taking place outside IBM, as the appearance on the market of increasingly easy-to-use products seems to indicate. However there is already a serious shortage of people skilled in this kind of work. I foresee the rise of "usability consultants", some of whom will deign to undertake experimental work, but more and more may turn to giving advice in the nature of a housing "structural survey", largely unsupported by experiment. In the present state of things that may be dangerous, and discredit this sort of consultant.

However, in the long run, I feel these "usability consultants" will turn out to be the forerunners of the soundly grounded Cognitive Engineers the industry so badly needs.

REFERENCES

[1] BARNARD, P.J., HAMMOND, N.V., MORTON, J., LONG, J.B., CLARK, I.A., (1981), Consistency and compatibility in human-computer dialogue,
Internat. J. of Man-Machine Studies, 15, 87-134.

[2] CLARK, I.A., (1981), Software simulation as a tool for usable product design,
IBM Systems J., 20, 3, 272-293.

[3] BARNARD,P.J., HAMMOND,N.V. (1982) Usability and its multiple determination for the occasional user of interactive systems.
In M.B.Williams (ed.), Pathways to the Information Society, North-Holland: Oxford, pp 543-548.
Also Hursley Human Factors Report: HF059. IBM UK Laboratories ltd., Hursley, Winchester, Hants., UK.

[4] HAMMOND,N.V., BARNARD,P.J., CLARK,I.A., MORTON,J., LONG,J.B. (1980) Structure and content in interactive dialogue.
Paper presented at: 88th Annual Convention of the American Psychological Association, Montreal.
Also Hursley Human Factors Report HF034. IBM UK Laboratories ltd., Hursley, Winchester, Hants., UK.

[5] HAMMOND,N.V., LONG,J.B., MORTON,J., BARNARD,P.J., CLARK,I.A. (1980) Documenting user difficulties in interactive systems: annotated performance protocols.
Hursley Human Factors Report HF024. IBM UK Laboratories ltd., Hursley, Winchester, Hants., UK.

[6] MORTON,J., BARNARD,P.B., HAMMOND,N.V., LONG,J.B (1979) Interacting with the computer: A framework.
In E.J.Boutmy and A.Danthine (eds.), Teleinformatics '79. North-Holland: Amsterdam, pp 201-208.

[7] BARNARD,P., HAMMOND,N., (1982) Cognitive Contexts and Interactive Communication.
Electronic Journal, in press.
Also Hursley Human Factors Report HF070. IBM UK Laboratories ltd., Hursley, Winchester, Hants., UK.

[8] HAMMOND,N., MORTON,J., MACLEAN,A., BARNARD,P. (1983)
Fragments and signposts: Users' models of the system.
Proceedings of 10th International Symposium on Human Factors in Telecommunication, Helsinki, June.
A slightly expanded version of this paper is available as: Knowledge Fragments and Users' Models of Systems. Hursley Human Factors Report HF071. IBM UK Laboratories ltd., Hursley, Winchester, Hants., UK.

[9] CHAPANIS A, (1967), The relevance of laboratory studies to practical situations.
Ergonomics 10, 557-577.

[10] SINGLETON W T, (1967), The systems prototype and his design problems.
Ergonomics 10, 120-124.

DISCUSSION

Dr. Card suggested that there was a need to distinguish between research and design activities: in the case of research, more experimental techniques can be investigated to generate models, etc. rather than simplify design work. The speaker agreed - it takes a long time to design a system from the ground up. There was a future for ´packaged experiments´, which could be used as a tool in design, but also a danger in simply accepting such a test, without appreciating all its assumptions.

Mr. Howard asked whether experts´ performance could be related to novices´ reactions; he was under the impression that Cognitive Science had shown otherwise, and that expert behaviour was different in kind. Dr. Clark said that he was not a psychologist, but agreed that ´plateau effects´ were possible, and that one could not predict expert behaviour from that of novices. The converse was not necessarily true - where the expert stumbles, the novice will almost certainly stick.

Dr. Barnard stated that with experts, error recovery is automatic and there apppears to be a smooth flow of actions, but novices can´t maintain this continuous sequence. The speaker referred to some work on keyboard memorisation, which showed that even experts looked down at times!

In response to a comment on some studies of the interpretation by radiologists of X-ray films, where it had been noted that the novice scanning the whole picture could in some cases find more than the expert, Dr. Clark suggested that the novice programmer often adopts a strategy designed to minimise pitfalls - e.g. in the case of PL/1, not using procedures!

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]