Design for a changing environment

J. N. G. Brittan

Military Vehicles and Engineering Establishment, Cobham Lane, Chertsey, Surrey KT16 OEE

1. Introduction

The theme of this conference is 'New approaches in systems analysis and design'. In one sense, there is nothing very new and certainly nothing clever in what I am going to say but I make no particular apology for this. In the computer and information systems field we have perhaps spent too much effort on inventing new technologies, new operating systems, new languages, new applications of software and new techniques generally. Too often we find (after the event) that we are inventing a rather old wheel with different coloured spokes. We have not paid enough attention to learning by the experience of others, discussing, consolidating, and codifying what we do know, and above all trying to get what is well known in theory to work in practice.

In this paper, the word 'design' is used in a rather wider sense than is normal in this sort of forum. In most information systems projects we start with an idea or a problem and we have to produce a system which implements the idea or solves the problem, subject to a number of constraints. This is a process known in engineering as development and a key activity early in the project is the design of what I call the development strategy. In this paper the word 'design' includes this meaning.

An information system as implemented represents a synthesis between what the users want, or think they want, or state that they want and the designer's appreciation of the user's wants and needs, together with the constraints of time, cost, human capability and technology. In a large project, all these factors are likely to change as development proceeds so that this process of synthesis can be very complicated. The lack of a mechanism for ensuring that this synthesis leads to an acceptable solution is a possible recipe for disaster as we saw in the many failures in information systems projects in the 1960s. This lesson has been learned and nearly all organisations have sets of procedures for information systems analysis, design and development which attempt to provide a framework for taking a project from initial conception to successful implementation and operation. In this paper, however, it is suggested that nearly all procedures are based on one particular development strategy and take no account of possible alternatives. One of these alternatives will be described and its advantages and disadvantages discussed.

Most procedures as set out in company or departmental standards manuals regard an information systems project as a set of activities which are carried out serially. The project development cycle as given by the Central Computer Agency (1976) is given in **Fig. 1.**

The terms used to describe the different stages and the degrees of emphasis placed on each vary from organisation to organisation; for example, Stage 2 could be described by such terms as Initial Study or Feasibility Study. Decision points are usually included so that effort, conclusions and recommendations can be justified to management and approval sought to proceed to the next stage.

2. The linear strategy

This serial definition of the project development cycle, known as the linear strategy, embodies one fundamental concept: that one activity follows logically from its predecessor so that each stage is complete before the next begins. To quote from the CCA Management Note (1976) on computer system development, 'Project development will proceed in stages, with no authority to the Project Manager to commit resources beyond the stage currently authorised'. Some allowance is often made for looping back when detailed investigations and analysis reveal problems or when questions indicate that a change in requirement may be necessary, but any extensive looping back is regarded as both unusual and unsatisfactory since it can be held to imply deficiencies in earlier work.

An important feature of the linear strategy is that it must be possible to take all important decisions at the appropriate time in the project life cycle. In particular, it must be possible to establish fairly firm requirements during Stages 3 and 4; significant modifications during later stages, though they may frequently occur in practice, are contrary to the spirit of this strategy and need to be very tightly controlled. Although provision for modification after implementation is normally allowed, such modifications are required to be minor so that no fundamental redesign is necessary. Thus, the linear strategy places great reliance on the studies in the early stages of the project development cycle.

3. Shortcomings of the linear strategy

Let it be said at the outset that in a majority of cases, particularly when the organisation responsible for designing and implementing the system has experience of similar systems and when the users are clear about what they want, the linear strategy is perfectly satisfactory and produces good results. When these conditions are not fulfilled, success is rather more patchy. Too often, a project starts on the linear strategy but the initial requirement is vague, over-ambitious or fails to meet the real need: in fact the requirement is still fluid. The project then proceeds in a series of short term loops as the requirement solidifies:

1. Analysis is started on the basis of the requirement, which then changes requiring further analysis.
2. Part of the system is then designed to the new requirement, which changes again.
3. Further analysis and design is then required.
4. When implementation starts, a feature of the new design is found to be impracticable; it is changed.

1 PROJECT PROPOSAL STAGE

2 PRELIMINARY STUDY STAGE

3 FULL STUDY STAGE

4 SYSTEM SPECIFICATION STAGE

5 SYSTEM CONSTRUCTION STAGE

6 IMPLEMENTATION IMPACT STAGE

7 FINAL IMPLEMENTATION STAGE

8 PROJECT CLOSURE STAGE

Fig. 1 Stages in project development (from CCA Management Note No. 4)

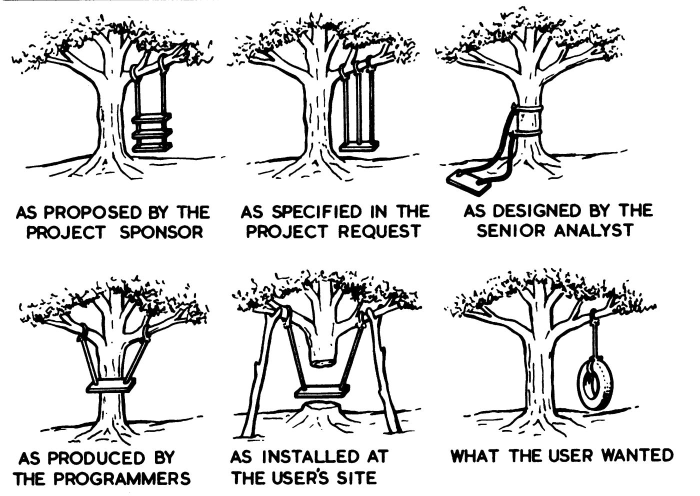
AS PROPOSED BY THE PROJECT SPONSOR
AS SPECIFIED IN THE PROJECT REQUEST
AS DESIGNED BY THE SENIOR ANALYST
AS PRODUCED BY THE PROGRAMMERS
AS INSTALLED AT THE USER'S SITE
WHAT THE USER WANTED

Fig. 2 Design for a changing environment—1: The user's view of what happened

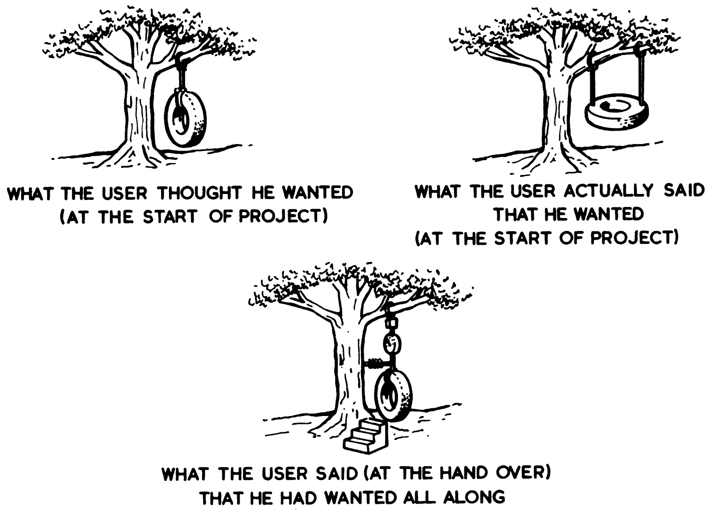
WHAT THE USER THOUGHT HE WANTED (AT THE START OF PROJECT)
WHAT THE USER ACTUALLY SAID THAT HE WANTED (AT THE START OF PROJECT)
WHAT THE USER SAID (AT THE HAND OVER) THAT HE HAD WANTED ALL ALONG

Fig. 3 Design for a changing environment—2: Another view of the user's part in the project

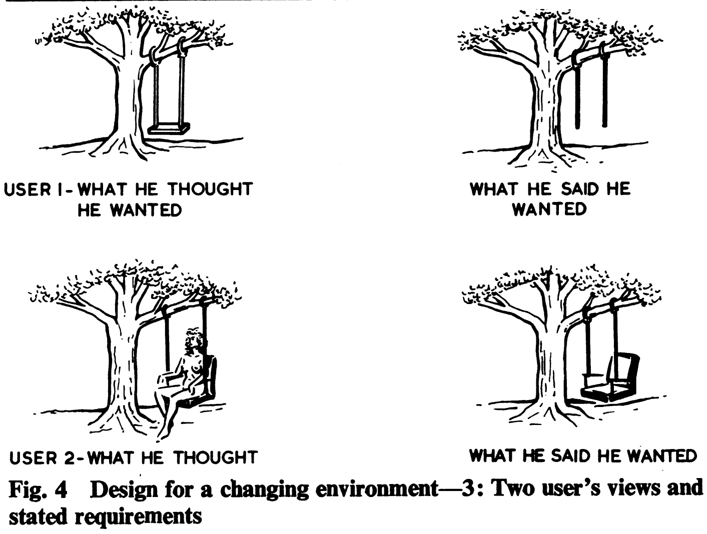

Fig. 4 Design for a changing environment—3: Two user's views and stated requirements

5. Before the project is complete, the requirement is changed again, causing a loop back to design.
6. and so on.

4. The loopy linear strategy

We shall call this method of proceeding in a series of somewhat haphazard and short term loops, the 'loopy linear' strategy. Some loops are inevitable in any difficult project but too many will be disastrous. One of the symptoms of excessive loopiness is a feeling of antipathy between the different groups associated with the project, particularly if they are geographically dispersed, (the 'we/they' syndrome); the system designers will grumble about users never knowing what they want and users will be annoyed by the apparent lack of good project management as the system overruns its budget in both time and cost.

A few years ago, it was quite common to show something like **Fig. 2** in talks on systems analysis and design. It is quite a good illustration of problems in communication downstream from the user, but the phrase 'what the user wanted' begs the very difficult question of getting the user to decide what he wants, to specify it clearly and live with what he has said; Fig. 1 is the user's view of the way the project went. In fact, his/her role was not quite as lily-white as Fig. 2 implies. **Fig. 3** shows that, although at the start of the project he wanted a tyre hanging vertically on a single rope, what he actually said to the designers was something different; and at handover what he said he had wanted all along was different again. However, even this is an oversimplification because of course the user is not one person, and **Fig. 4** gives a picture of different thoughts on the requirement and different statements of the requirement. Of course I have exaggerated to make a point but these kinds of things can and do occur and when they do, excessive loopiness in the project development is likely to follow. One of the great problems with the linear strategy is that the design can only be based on studies, investigations, calculations, estimations, simulation or experience of similar systems. Unfortunately, most end users who ultimately determine the degree of system success are not usually adept at such conjecture and extrapolation, especially when they have no experience in the use of computers or automated information systems. They can, of course, describe the operation of the existing system (if there is one), but to ask them to extrapolate too far beyond their experience is dangerous. Once a user can get to grips with the new system, the comments, criticisms and suggestions come in thick and fast. Yet the truly linear strategy, by definition, cannot allow for such valuable user feedback in the project development. New user requirements and minor modifications can usually be put into effect after what is described as a 'final implementation', but meatier modifications or fundamental redesign pose serious problems for the linear strategy. The rest of the paper is devoted to a discussion of an alternative to the linear strategy, known as the 'Prototype strategy' (Bally, Brittan and Wagner, 1977), which does not suffer from the same shortcomings as the linear strategy. This approach is well established in a number of fields of engineering and the idea has been around for a long time in information processing, but has not really been accepted and I have yet to see it in any Standard Procedure Manual as a firm alternative to the linear strategy.

5. The prototype strategy

In the prototype strategy, an initial and usually simplified prototype version of the system is designed, implemented, tested and brought into operation. Based on the experience gained in the operation of the first prototype, a revised requirement is established, and a second prototype is designed and implemented. The cycle is repeated as often as is necessary to achieve a satisfactory operational system, bearing in mind the escalating costs of each subsequent cycle; it is very likely that only one prototype would be necessary before the final system.

There are wide variations possible in the prototype strategy. It can look, with hindsight, very like a phased version of the linear strategy if the prototype does in fact become part of the final system. If this can be done on the day, it is of course

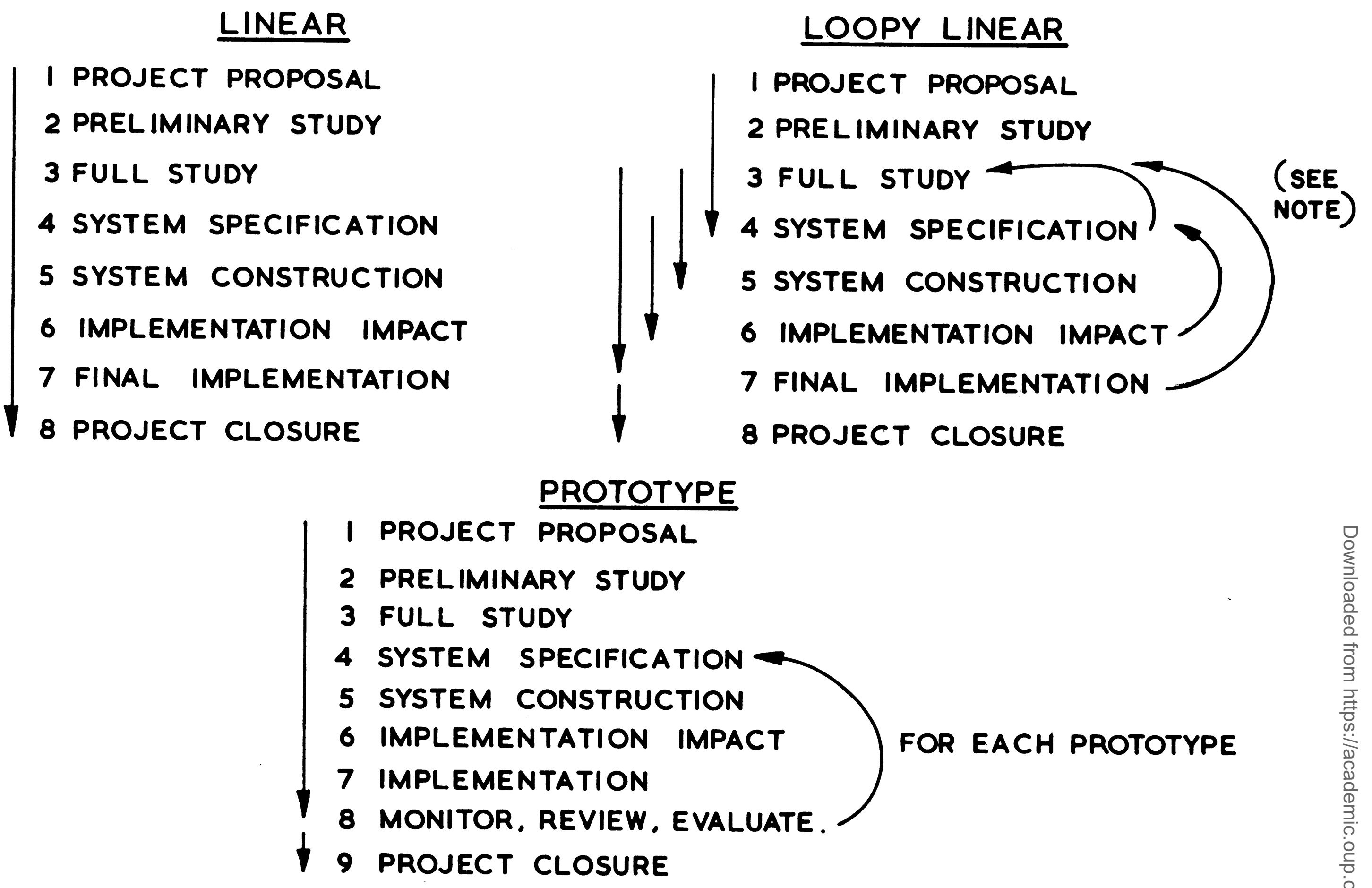

Fig. 5 Development strategies
Note These arrows represent a haphazard and unplanned series of advances and loops back to previous stages

excellent but to plan on the assumption that this will happen is to negate the whole basis which is that the prototype(s) are regarded as test vehicles built to gain information about one or more aspects of the final system, e.g. what are the real requirements, or how do we design and build the system? The final system may well have features of one or more of the prototypes, but the basic philosophy is pragmatic and experimental; the aim of the prototype is to learn, to find out, to discover. It may well be that circumstances, previous experience or other knowledge will enable the project management to define the 'unknowns' in a particular project as belonging to certain areas and to confine the learning element in the strategy to these areas. It may be possible to standardise on a specific hardware range or on a particular documentation system and programming language, but shackles and constraints that are attached to the prototype strategy (particularly if these are built into a standard procedure) must inevitably detract from the heuristic nature of this strategy.

The difference project development cycles under the linear, loopy linear and prototype strategies are summarised in **Fig. 5.**

6. Examples

The first example of the Prototype Strategy is a very small workshop planning system in which the author was involved in the early stages. A survey showed that none of the available planning and production control systems would be suitable since all were too complicated for the job in hand. Thus, a system would have to be developed to meet the requirement, seen initially as simply recording information on the various jobs as they were presented to the shops. This information included items like title, job number, estimates of requirement for resources of different types (pattern shop, foundry, etc.), and priority. There would also be a requirement to sort the information in various ways (such as jobs in priority order and predicted load on various types of resources). Eventually some limited scheduling might be attempted using the system. The staff in the workshops had no experience in using this type of automated system, so that although the management were fairly sure of current needs, they could not say what they might want in the future. The only programming effort available was one student gaining industrial experience during one 6-month period. It was decided to use the prototype strategy with the first prototype essentially a punched card system. Cards representing job transactions were input to a file using the standard facilities of the operating system, and the resulting file (with information on all the jobs) was listed weekly on the line printer in the computer room one half-mile from the workshops. This system required no computer programming and was implemented within one month of the statement of requirement. This first prototype was valuable in familiarising the workshops staff in interacting with a computer. The second prototype involved the development of sorting facilities so that more valuable management information could be produced.

The success and cost-effectiveness of this very simple system convinced management that they should take the further step of purchasing a terminal which would be located in the workshops planning area but linked to the main computer for further development of the system. Job data is now keyed-in directly, thereby eliminating transcription to special forms for transmission by messenger to the Computer Centre and subsequent key punching. Experiments with a limited degree of job scheduling are now under way. It is important to note that each step in this simple development sequence was taken in response to a clearly perceived demand based on practical experience gained in the previous step. Of course, this is a very small scale, almost trivial, example although it has some interesting features, in particular a system brought into operation very quickly by people with little or no previous computer experience.

6.1 *The Driver and Vehicle Licencing System*

The second example is a very large project well known to nearly everyone, on which hindsight indicates that the prototype strategy could have been used with considerable advantage. It is the Driver and Vehicle Licensing system based on the centre at Swansea. As the basic functions of the system are very generally known, a minimum of background information will be given: a significant part of the project development history is summarised in **Fig. 6** which is taken from the report of the Public Accounts Committee for Session 1974-75 (HMSO, 1975).

The following paragraphs are extracted from the PAC Report.

'38. Your Committee were informed by the Department and Central Computer Agency that in 1965 no relevant experience of a project of this kind was available in this country or elsewhere. It was only as the project developed that the demands imposed by its scale and complexity could be fully assessed. The Working Party had consequently underestimated the numbers of staff required and overestimated the work rate in the sections processing the applications. Detailed examination of computer system procedures during the period 1966-69 had elicited the need for additional computer staff, but the clerical work and staff which the system involved had continued to be underestimated. Practical trials in mid-1972 have provided the Department with a more realistic basis on which to assess the clerical tasks and they have increased their staff estimates accordingly. Also, experience of live running of driver licencing on the computer had shown that considerably more effort than had been expected was needed to deal individually with the small percentage of difficult cases, about 600 a day. We were further informed that in the mid-1960s it had been thought that, wherever possible, all processes should be computerised, with the result that insufficient consideration had been given to clerical processes and to the problems of handling a large mass of information received in uncollated form from the public and the bottlenecks this created.

'39. The Department agreed that the project had not been planned as well as it might have been and that decisions had been taken in 1968 on the basis of inadequate estimates, but they explained that the prime consideration at the time had been to proceed as quickly as possible with the new centralised system, since it was thought that the old system would be unable to cope with expected growth in driver and vehicle numbers. If they were embarking on this project now with present resources, they would have followed up the working Party's report with a full feasibility study before proceeding further. In the 1971 Review, when it was recognised that the new system would be more costly than keeping on with the old, the Government decided that the new system should go ahead. Since then, the Department has been in an operational stage.'

The report concludes by saying that, as this major and novel computer project was started without a thorough feasibility study, the Committee were not surprised that the planning and timetables were found to be unrealistic, and the completion delayed by some three years with consequent increase in costs. They were glad to be informed that the department, and indeed the service as a whole, had learnt that there was a need to look at 'off the computer' problems. They trusted that in future computer projects more attention would be given to preliminary studies and the preparation of reliable estimates of cost, even when there is an urgent need to proceed.

We are all, at times, prone to criticise our politicians but there is some horse sense in these paragraphs, rather more horse sense I am bound to say than is to be found in many voluminous papers written by computer professionals. But, nevertheless, have they drawn all the right conclusions? The implication of this report is that the project seriously overran its estimates in terms of development time, cost and staff required to operate the system, but that this would not have occurred if a thorough feasibility study had been done before starting. With all possible respect to this eminent committee, I wonder if this is really true. The report itself makes clear that detailed examination over three years (1966-69) had elicited the need for more computer staff but that the clerical work required had continued to be under-estimated. It was not until practical trials (in mid-1972—after the first planned live date) and live running that the size of the clerical task and the need for large manual effort on exception cases were properly appreciated. Would a thorough feasibility study have brought these problems to light where a 'detailed examination' over three years failed to do so? It is easy for us, with the benefit of hindsight, to say we would have spotted the problem areas, if we were in the position of doing the job at the time, but I somehow doubt if many of us would actually have done so.

Now just supposing that the project had been planned in three phases.

Phase 1 Design and build a prototype Driver and Vehicle Licencing system for processing the licences in, say, one county.

DATE OF ESTIMATE	ESTIMATED STAFF NUMBERS	ESTIMATED COST	VEHICLE LICENSING ESTIMATED DATE TO START	VEHICLE LICENSING ESTIMATED COMPLETION DATE
1966	3950			
1968	5431	£ 146 M	4·72	1·75
1970	7024	£ 233·5 M		
1974	7961	£ 350 M	10·74	12·77

Fig. 6 Driver and Vehicle Licensing project
(information from Report of Committee of Public Accounts Session 1974/75)

Phase 2 Put the prototype system into operation, evaluate the results and, based on this evaluation (of actual operation on a relatively small scale) assess the costs and benefit of applying the system to the whole country.

If considered worthwhile, go to

Phase 3 Design, build and operate a countrywide system.

If this had been done nearly all the lessons about 'off the computer' problems, increased clerical effort and problems with exception cases would have been learned during Phase 2. This phase would also have thrown up the painful truth that (quoting the PAC Report again) '. . . the new system would be more costly than keeping on with the old . . .', and, here is the catch, *before* the Government was committed to going ahead with the complete new system. It also might just have thrown up the idea of doing away with Vehicle Excise Duty altogether.

This is not to criticise those involved. I might well have done the same myself given the various political and other constraints, but if we do not look back we shall never learn. Here was a case where something was being attempted which had never been done before. It involved a large commitment in terms of staff and money and a major upheaval to the existing arrangements for processing transactions involving a large percentage of the population of the UK and where the 'environment' was liable to change. This is one sort of project where an alternative to the linear strategy should be considered very seriously indeed and, as has been pointed out, the prototype strategy would in fact have been very suitable.

7. Discussion

As has already been stated, a major advantage of the prototype strategy is that it is designed to cope with a fluid situation, and a changing environment. We have all been involved with the fuzzy type of requirement such as: 'I have a management problem: I don't seem to be getting the right information, or if I get the right information it is too late. I am sure your computer could help me. Go away and produce a scheme for getting my information on to the computer.' Essentially that was the situation at the start of the small workshops system described above.

In order to use the linear strategy for system development, the analyst would refine the requirement in increasing detail by investigations and studies involving the originator, various subordinate levels of management, and a host of other agencies or relevant activities. Frequently such investigations throw up problems which were not suspected at the outset; for example, the organisation may not be clearly defined or it does not behave in the way that it is defined on paper. Under these circumstances, the analysts can spend weeks, months and even years probing all these highways and byways of the existing system before they can design the final system. I wonder how long we would have taken to get the small workshops system going using this approach.

With the prototype strategy, these investigations and studies need be far less protracted. In devising the prototype system, the analysts can often base their detailed interpretations on their own assumptions. The users, for their part, knowing that these assumptions can be proved or disproved when the prototype is put into operation, can be far bolder about accepting them than when faced with a 'last chance to make up your mind on the detailed requirement' which occurs when using the linear strategy. This shows the greatest advantage of the prototype strategy: the generation of user confidence. Any information processing system must achieve both technical and psychological success. Technical success is the degree to which the actual performance of the system matches its specification, whilst psychological success is the degree to which the end user has confidence in the final system. Of course these components are closely interrelated: technical success is normally required for psychological success, but a psychological failure can negate a technical success. Now, the prototype strategy is designed to allow the user to learn about, and gain experience in, the system at an early stage in the development cycle and allow modification, perhaps radical, to meet the real needs. Thus, having tailored the final system to requirements based on actual experience of the prototype(s) the user is far more likely to have confidence in the final product. Designers of automated information systems tend to underestimate the demands which the linear strategy makes on the inexperienced user. It is rather like asking him to agree to having a new and exotic breakfast food every day, on the basis of paper studies, but without having tasted it. The prototype strategy gives him a chance to taste it first.

Again, because all concerned can be bolder about making and accepting assumptions, the prototype strategy will produce something tangible, something which can actually be used, much more quickly than the linear strategy. It encourages a feeling of 'let's get cracking and get on with it'. This is an attitude of mind which the Victorians had but we seem to have lost in so many fields today.

A number of technological advances in both hardware and software make the prototype strategy rather more attractive than in the past. Hardware has become smaller and cheaper—not only that but the general perception of computers has begun to change. With the advent of computers for sale in the high street the conviction, particularly among noncomputer people, that a computer is necessarily something very expensive has begun to change and therefore the idea of possibly buying a computer, developing something on it and then perhaps throwing it away or doing something quite different becomes less unthinkable. Software developments have enabled systems targeted for one type of hardware to be developed and tried out on another and software prototyping has become a fairly well established technique, although usually still within the overall development framework of the linear strategy.

The prototype strategy, of course, has a number of disadvantages. It is difficult to find the right balance between the aims of keeping the prototype simple and giving it enough resemblance to the system to allow the correct lessons to be learned by system developers and users. The 'model effect' is well known in engineering development where the effects of size, dimensionality, complexity and lack of inclusion of all possible facets, can lead to the wrong conclusions being drawn from a prototype.

However, the greatest disadvantage is, of course, the greater apparent cost at the start of a project of deciding to adopt this strategy. There are many projects which, with hindsight, one can see would have benefited from adopting the prototype strategy as, for example, the Driver and Vehicle Licensing System. The trouble is that hindsight is the one thing that people at the start of a project do not have, and it is often only with hindsight that one can see just how much uncertainty actually existed at the start of a project and how much the environment and the requirement changed during the development period. At its inception, a project is a fairly delicate flower highly vulnerable to cancellation. To the perhaps sceptical laymen with the money and the power to say Yes or No a proposal to build a prototype system which may have to be extensively modified or even thrown away after implementation is not immediately attractive—to put it mildly. That is why the question of codification and standard procedures was emphasised at the start of this paper. If we accept my basic proposition, that the prototype strategy has definite merits in a proportion, perhaps even a small proportion, of cases and more generally that alternatives to the linear strategy should

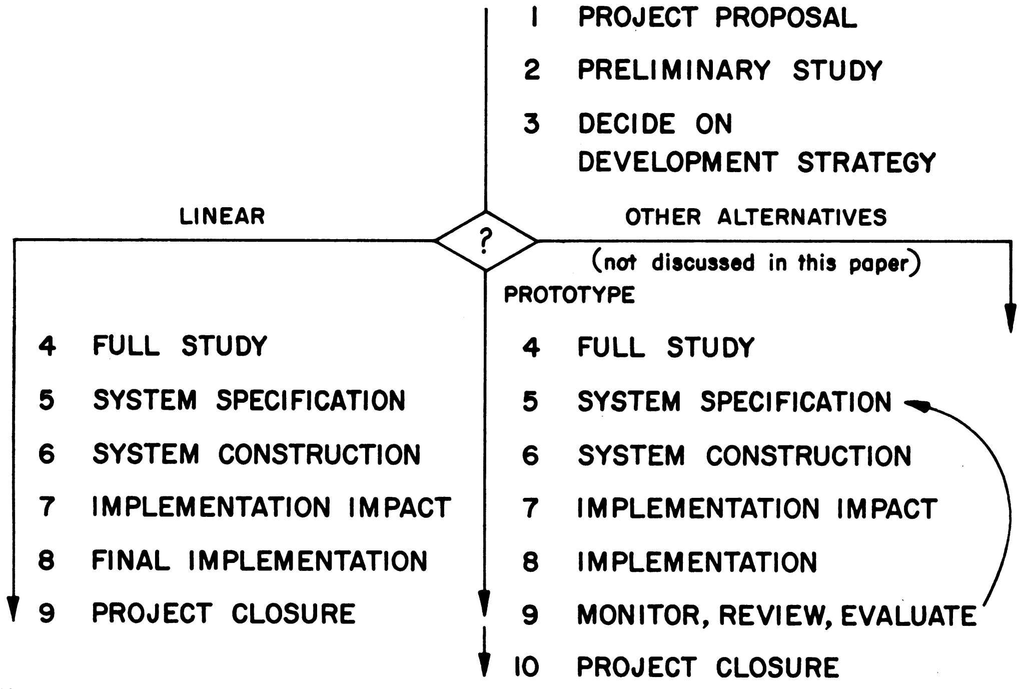

Fig. 7 Stages in project development—proposed

always be considered at the start of a project, then this proposition must find its way into the standard procedures for companies and other organisations, and the appropriate training and education courses. If he does not have this kind of formal support, the individual project manager will have little hope of justifying, say, the prototype strategy at the start of his project when it is at its most vulnerable.

All this boils down to one extra step in the project development cycle: 'decide on development strategy'. This would mean that the outline of a project development cycle according to the Central Computer Agency as given in Fig. 1, would be changed to that given in **Fig. 7.**

8. Comparison with other fields of engineering

There is much to be gained from looking at fields other than information processing. If we could only bring ourselves to accept that information processing and, in particular, the development of information processing systems, is not just like engineering but in fact is a branch of engineering (Brittan, 1978), then we would get a good deal of benefit. In particular, concepts which are well established in older branches of engineering but which are regarded as a little novel in information processing, or as I would prefer to call it, information engineering, would fall naturally into place. In this context one might think of such comparatively recent ideas as decomposition of systems into subsystems and modules, rigorous quality assurance and testing procedures, particularly as applied to software, and the use of prototypes as a well established procedure. There are, of course, important differences between a large management information system which is probably 'one off' and a vehicle or radio set which may have a production run of many thousands, but the basic idea of building something and trying it before going ahead with the final product can be equally applicable to both.

9. Conclusion

This paper has emphasised that there are alternatives to the linear strategy for the development of information processing systems, and that those responsible for projects should understand these alternatives and be able to make a balanced and reasoned judgement between them at the start of the project. The prototype strategy has been emphasised, not because it is the best in any general sense but because it is much less widely accepted than other strategies. In one sense, the prototype strategy is an admission of failure, an admission that there will be circumstances in which, however good our techniques and tools for investigation, analysis and design, we shall not be able to develop the right system at the first attempt, quite possibly because of changing environmental factors beyond our control; but surely this is only realism based on hard experience. Theoretically ideal solutions are often far from satisfactory in a very imperfect world.

When one looks at the different approaches to information system design and development over the very short history of computing, one can discern a swing of the pendulum. In the 1960s, during the computer explosion, people bought computers (which were relatively large and expensive) often without any very clear idea of what they were going to do with them. As a result, a great deal of money was wasted—at least apparently it was wasted. The lesson, as has been said earlier, was well and truly learned by the late 60s and early 70s. By then, everyone (including me—I was the joint author of a pamphlet based on the linear strategy (Brittan and Wade, 1971)) was talking about the need to do proper studies, justification, etc. at every stage before going ahead. It is this experience which has burnt the linear strategy so deeply into all of us.

In essence, this paper is saying that the pendulum did perhaps swing a little too far during that period. If there is one universal human shortcoming it is that we underestimate the value of learning: perhaps not all the money that we said was wasted in the 1960s really was wasted because a great many people got their feet wet, made their mistakes and learned a great deal. There are times when this approach, untidy and inelegant as it may seem, may be the only one that works.

10. Recommendation

Let me end on a practical note. At a software conference a year or two ago, an eminent professor complained that, while the cogniscenti were all talking ALGOL 68 and PASCAL, back at the ranch everyone was actually using COBOL and FORTRAN. How can we get some of the excellent ideas put forward at this conference knocked into a shape that can actually be *used* by a wide spectrum of practising analysts, designers or—as I would prefer to call them—information engineers? We need, as has already been said, to get these ideas built into standard procedures, training courses and so on. A first step would be to produce a suitable book or pamphlet: could not the British Computer Society sponsor the production of such a book by a suitable working party?

References

Bally, L., Brittan, J. N. G. and Wagner, K. (1977). A Prototype approach to information system design and development, *Information and Management* Vol. 1, pp. 21-26.

Brittan, J. N. G. and Wade, R. (1971). *Systems Analysis and Design*, Royal Military College of Science, Tech note 8, April 1971.

Brittan, J. N. G. (1978). Information Engineering, *Computer Bulletin*, March 1978, pp. 8-9.

Central Computer Agency. (1978). *Computer System Development: 2. Project Management*, CCA Management Note No 4, April 1976.

HMSO. (1975). 4th Report from the Committee of Public Accounts, Session 1974/75, July 1975.

The impact of social movements

K. Nygaard*

Norwegian Computing Centre, Forskingsvnn, 1B, Oslo 3, Norway

1. Introduction

I am very honoured to have been asked to speak at a conference which is being held in memory of Eric Mutch. As has been said a number of times, Eric was very concerned about the problems of the user. In the papers which accompany this one the needs of the user is a factor which occurs again and again. So 'user needs' has become a crucial term and concept in the development of data processing systems.

2. The user

It is my opinion that the term 'user' is not sufficiently precise for a discussion of systems design strategies. It can be used as a common term but if we wish to undertake a detailed examination of different strategies then we need a much more carefully thought out definition.

There are two main ways of considering users, both of which are crucial in their own way. The first is to consider 'functional roles'. In sociology the notion of a role is interpreted in different ways and used in different contexts, but when I use the term role I am thinking about functional roles. This is because, when we are developing systems, we have to consider carefully the different functional roles which are involved. Some examples of functional roles are the following:

(*a*) the operator role—the operator of a system is a person who, in everyday work, is engaged in interaction with the system and the system is dependent on his actions in order to function.

(*b*) the customer role—a customer is a person who interacts with the system in order to obtain some service.

(*c*) the ruler role—a ruler is a person who is in control of resources and in a position to be able to define objectives for a system.

The role of the designer, whom I shall call systems specialist, has to encompass all of these, hence one role can be subdivided into other roles. One person may have a number of different functional roles all at one time. Similarly a role may be enacted by a group of people rather than an individual. Functional roles are most important in systems development since most of the material which has to be prepared is frequently related to these functional roles.

The second way of considering users is to look at them as 'interest groups'. It isn't possible to give a deep philosophical or clear definition of an interest group, rather it is better to look around and see the ways in which people join together in different kinds of association through which they work on behalf of their common interest. The most common forms of interest groups in the industrial scene are employers and trade unions. Interest groups are defined in a different way from functional roles. The assumption is that interest groups spread across functional roles; although people might have different jobs they could still have common interests.

The particular user classification and user concept which I am going to discuss in this paper is that which is related to the problems of organised workers, that is workers organised in trade unions. In particular I shall describe how unions react towards systems development and how they participate in it. During the discussion it may be that a number of points will arise which will have relevance to other interest groups.

3. The system

As part of this discussion of systems design strategies it is also necessary to say something about the concept 'system'. I am doing this not only in order to give my pet definition of a system but also to emphasise something which is important to keep in mind when one is considering the situation where a number of different interest groups are involved in problems of systems development.

Following Langefors it is possible to give a number of definitions of a system. One of these is to define a system as a collection of components which have some relation to each other. But a consequence of this definition is that it is not possible to ascertain whether any part of reality is, or is not, a system because everything is a system and has always been a system. There has always been a system and there always will be a system. Hence this definition is not very useful.

The kind of definition of a system which we use in data processing is something like the following. A system is a part of the world which we regard as the whole for some period of time under consideration and which we separate from the rest of the world. We then define an 'entity', which we define as consisting of a set of components, each being characterised by a set of

*This paper is an edited transcript of the actual talk given by Kristen Nygaard at the conference. The Journal is confident that no loss of salient points has occurred through this method of presentation.