Software Prototyping: Implications for the People Involved in Systems Development

Pam Mayhew

School of Information Systems
University of East Anglia
Norwich, NR4 7TJ, England

Abstract

This paper explores the adoption of systems prototyping and, in particular, concentrates on the various ways in which this may affect those people associated with systems development.

One widely accepted prerequisite for successful prototyping is that it requires a fundamental change of attitudes for developers, users and managers alike. This all too frequently ignored requirement is discussed and illustrated.

Adopting a prototyping approach to systems development can have a significant effect on the roles played by both users and developers. These alterations in roles and responsibilities are examined. Particular attention is given to the role of the prototyper, as well as to the issues to be considered when choosing who should play this crucial part.

Examples are drawn from experience of both teaching prototyping within a University and its adoption in a commercial environment.

1. Introduction

During the 1980's systems prototyping has fast been transformed from a relatively unknown research topic to a widely discussed approach to developing computer based information systems. Consequently, there now exists a considerable amount of literature on the subject. The majority of these publications deal with one or more of the following areas:

- a general discussion of prototyping often concentrating on advantages and problem areas;
- a description of the execution of prototyping, often in an ad-hoc and retrospective fashion;

- a detailed presentation of a particular product, followed by an argument as to its suitability for use as a prototyping tool.

Unfortunately, a significant number of the available articles give the misleading impression that prototyping is a simple, even trivial process. This occurs in two main ways:

- the first implication is that with the currently available range of sophisticated development tools, prototyping is simply a matter of choosing and subsequently using an appropriate tool;

- the second misleading impression is founded in the notion of a "stereotype prototyping session" [Boar 1984]. This involves one user and one developer, sitting at one terminal making immediate changes to a simple problem system.

Some prototyping may be as straightforward as this, but the majority will not. A prototype must be developed just as any other piece of software. Its production will involve all the investigation, analysis, design, implementation and evaluation activities found in the more conventional approaches to software development. In addition to the construction and evaluation of useful prototypes, successful prototyping requires careful preparation, detailed organisation and effective control.

Despite the widely held debate on the use of prototyping, there are only a small number of documented cases of its actual adoption within commercial organisations. Furthermore, a disappointingly low percentage of these articles actually discuss and assess details of the manner in which the prototyping was organised and controlled.

In practice there are a wide range of issues to be taken into account when preparing to adopt a prototyping approach to development. These necessarily include many decisions that will affect the people who are to be involved in the prototyping, for instance: who is to participate?; how are participants to be selected?; what are their roles and responsibilities?; how are they to be prepared?; how is prototyping to be organised?

Careful attention to issues relating to the preparation and organisation of prototyping maximises the chances of it being successful. Since prototyping is critically dependant on those people who are participating, it is clear that decisions concerning people are central to the success of the entire prototyping process. This paper highlights some of these important issues.

Prototyping provides a hitherto unparalleled opportunity for the meaningful, active participation of users within the systems development process. Such a high level of participation must be expected to have implications for all the people involved. Hence this paper also explores the adoption of systems prototyping, with particular emphasis on the various ways in which this may affect those people associated with systems development.

2. Towards Successful Prototyping

An understanding of what is meant by "successful prototyping" can be gained by considering a definition of systems prototyping. Prototyping is defined as:

"the process of constructing and evaluating working models of a system in order to LEARN about certain aspects of the required system and/or its potential solution."

The crucial component of this definition is the word "learn". If nothing has been learned from the prototyping process then it has been a waste of time and effort.

There are many issues which can be clarified by the use of system prototyping, such as establishing requirements; assessing alternative designs; and investigating the likely affect upon the organisation [Floyd 1984, Mayhew 1987a]. Assuming that the objectives of prototyping have been carefully planned and recorded in advance, one definition of successful prototyping is as follows:

"prototyping which enables LEARNING about those aspects of the required system that were identified as crucial at the outset."

Experience suggests that this might be a somewhat narrow definition, as prototyping often enables those involved to learn far more than just those aspects initially declared. In addition, such a high level of user participation inevitably affects the existing working environment, influencing such things as personal attitudes, expectations, relationships and responsibilities. It is important that the full implications of prototyping are taken into account during the early preparatory stages. In this way the chances of obtaining the maximum potential benefit from prototyping are enhanced, thus enabling "more successful" prototyping.

Three central factors have been identified as crucial prerequisites for successful prototyping [Law 1985], as follows:

- suitable tools;
- changes in attitudes;
- a methodology.

Appropriate tools must be available that enable both the speedy construction and the speedy modification of software prototypes. Almost every day sees the introduction of some new high-productivity system building tool onto the market. While few, if in fact any of these have been designed specifically as prototyping tools, many of them could be used in this way. Experiments have indicated that even the simplest tools enable much to be learned through prototyping [Mayhew 1987b]. For these reasons, it is unlikely that the adoption of prototyping is being hampered, to any large degree, by a lack of suitable tools.

A change in peoples' attitudes is given as the second requirement for successful prototyping. The use of a more experimental systems development method challenges many of the underlying principles of the now widely accepted structured approaches. Given the amount of effort which has often accompanied the introduction of these more formal development methods, it is hardly surprising that systems development managers are somewhat less than enthusiastic when they first encounter prototyping. A proposal to construct a system (or more) with the intention of disposing of it soon afterwards, seems the very anti-thesis of the prevailing doctrine which these same managers may have worked hard to establish. Since the management views on prototyping are highly likely to affect the decision of whether or

not to pursue this approach to development, an adjustment in their attitudes is clearly necessary in order to enable successful prototyping.

Having decided to prototype part(s) of a required system, it must be appreciated that this will require both users and developers to work in a manner which is likely to be unfamiliar. Developers must be prepared not only to accept criticism but to elicit it effectively. Users must not expect perfection from early exploratory prototypes and must be encouraged to participate actively in the development of their system. Section 3, below, examines this "changes in attitude" prerequisite in further detail.

The third requirement is that of a methodology to support prototyping. Little has been published in this area, and there appears to be few guidelines as to how to put prototyping into practice [Riddle 1984]. In particular, questions such as: How do I introduce prototyping? How do I organise prototyping? and How do I control development that uses prototyping?, have largely remained unanswered [Harker 1988, Mayhew 1989, Vonk 1990, Mayhew 1990]. These methodological issues may be responsible for the reluctance of management to introduce prototyping into their organisations. The selection of prototyping participants, and the explanation of their roles and responsibilities, are two important topics to be taken into account when preparing to utilise a prototyping approach. These are discussed in detail below in Sections 4 and 5 respectively.

3. Prevailing Attitudes and Misunderstandings

The introduction and subsequent use of prototyping to assist in systems development constitutes a change in working practices. Changing working practices is no simple affair, necessitating a re-examination of both attitudes and traditions [Andersen 1990]. Hence, an organisation adopting a prototyping approach for the first time may encounter problems, e.g. the users take one glance at the first prototype and immediately begin to broadcast the incompetency of the developers; an egotistic developer refuses to acknowledge criticisms that indicate the redundancy of a particularly ingenious and intricate piece of code.

The majority of potential problems are founded in a lack of understanding and appreciation on behalf of the people involved, including managers, developers and users. Each participant needs to be familiar with the concepts of prototyping, his/her role within the process and what can be expected of other participants. Sections 3.1, 3.2 and 3.3 below, discuss some of the misapprehensions among participants, their implications, and suggestions for avoidance.

3.1 Management Attitudes

It is the management's attitude to prototyping that affects the decision of whether or not to pursue a prototyping approach to development. They have to be able and willing to expect, encourage, and accept re-work and iteration as a natural consequence of prototyping.

Systems development managers need to be sympathetic and supportive of the application of alternative tools and techniques. It is advantageous to establish a development environment in which creativity is not stifled [Amabile 1989]. This appears to contradict various well-known and widely used structured development methods, which aim to divide development into a large number of well-defined tasks [Cutts 1987]. Being forced to carry out a predetermined sequence of these tasks is intended to leave little opportunity for individual creativity. Prototyping is unlikely to perform at its best within this type of rigid regime.

Individual project mangers need to be willing to search for and establish enhanced methods for project management. The main areas of concern include: the justification of adopting a prototyping approach; project estimation; and the actual management of prototyping.

User managers need to be aware of the differences between a traditional pre-specification type development approach and an experimental approach involving prototyping. In particular, it is imperative that user managers are aware of the users' central role in the actual prototyping process. It is important that the significant increase in user time for participation is recognised at the outset. The managers must be able to reassure participating users that this is going to be "time well spent" and that they will not be penalised for having completed less of their "real" work.

3.2 Developer Attitudes

Developers require a thorough appreciation of any new tools and/or techniques which they may be applying for the first time. They are also likely to be more successful if they approach pilot developments of this type in as open-minded a fashion as is possible. This is as true for the introduction of prototyping as it is for other unfamiliar methods.

In fact this problem may be exacerbated for the introduction of prototyping. The false image, as discussed above, of prototyping as the simple application of high-productivity tools, is doubtless partially responsible for this concern. Developers may view prototyping as completely different from the approaches with which they are familiar. Prototyping may be seen as a threat to their hard-earned position and status, by rendering their current knowledge and experience worthless. These worries spring from a lack of understanding of the concepts of prototyping. Developers must be made aware that prototyping is an additional approach at their disposal, not a replacement methodology.

The proposal that prototyping needs to be seen as a useful approach to combine with existing approaches, should not, however, disguise the fact that to perform prototyping can require profound changes in attitudes for some developers.

Since prototyping, if it is to enable learning, requires the development of working models prior to their being satisfactorily specified, developers are constantly in a position of demonstrating that their work is not yet "correct". Some developers may not take kindly to "making a fool of themselves" in this way. The tendency to want to show the parts of the development that are working properly, rather than to expose those parts with serious shortcomings is only human nature. However, this can seriously jeopardise the success of prototyping. Experience, as described below, indicates that this may be a difficult problem to overcome.

A group of some thirty, third year Computer Science undergraduate students, were given an assessed coursework exercise that involved prototyping. Each of these students had attended several courses dealing with the implementation of computer systems. They had also taken, in the previous year, a first course in systems analysis and design, which had exposed them to a wide range of tools and techniques, mainly from the Structured Analysis family of development methods. Prior to the coursework being distributed, the class had been taught the concepts and practices of prototyping, and had been warned of the possible need for the adoption of unfamiliar attitudes. In particular, the need for the prototyper to be prepared not only to accept criticism, but to actually elicit it effectively, was discussed.

Each student was provided with a reasonably detailed account of the operation of a small wholesaler of domestic electrical appliances. (S)he was then required to produce a prototype and utilise this during a session with a representative of the firm, who was in practice one of the two course tutors. The exact orientation of the prototype was to be decided by each student, although the scenario provided did give pointers towards those parts of the system which were important and/or poorly defined. Students were allowed approximately three weeks to prepare for their prototyping sessions, and were instructed to use dBASE III on an IBM PC AT. Given that this course represented only a sixth of each student's workload, it was clear that the prototypes could at best automate only part of the original system description.

Despite the fact that this set of student developers knew about prototyping and its potential pitfalls, the outcome of this exercise was disappointing. Apart from a small number of students, it appeared very difficult for them to operate in this unfamiliar manner. This manifested itself in several ways, including:

- the choice of functions to prototype was often based upon the ones which were described in most detail and hence offered the greatest opportunity for getting it right, rather than the greatest opportunity for learning;
- the emphasis during the actual prototyping sessions was on showing what had been developed rather than on learning;
- the firm's representative was frequently discouraged from being anything more than an observer;
- when the representative managed to ask a question about a certain function, (s)he was all too often faced with a response such as, "Oh, I haven't implemented that yet, but let me show you what this function does";
- the user representative was rarely made to feel that (s)he was a crucial component of the prototyping activity.

Many students were unable, because of the way in which they handled the sessions, to learn any more than what the user thought of the small part of the system that they had already managed to construct. The potential to learn via prototyping had been wasted as a direct consequence of the self-protective attitude displayed by the developers.

Clearly, development in a University student environment is different from that in a commercial organisation. Students were being assessed on this exercise. Their desire to show their development talents, unfortunately, took precedence over what they had been

taught about prototyping. It appears that old habits do indeed die hard, even in a University environment. The worry is that there may well be a similar reluctance on behalf of commercial developers to expose themselves during prototyping. After all, the commercial developer, like the student, is rewarded by being seen to get things "right". It is extremely important that developers are prepared to admit that their knowledge is incomplete, and hence be in a position to learn via prototyping.

Prototyping often requires that prototypes are produced to short deadlines [McCracken 1981, Donovan 1977]. These deadlines are largely determined by the need to maintain user interest and enthusiasm. This can sometimes mean that models are not as complete as developers may have wished. This practice may be difficult for developers who have perfectionist tendencies to come to terms with. People with these attitudes, (which are highly commendable under different circumstances), must be reassured that even simple, incomplete prototypes enable much to be learned.

Developer participants in prototyping need to have a user-oriented attitude. They must be prepared to work in partnership with users. They must not just listen to the users' suggestions then automatically persuade them that the developer's approach is the only approach, but they must be willing to try these suggestions in practice. As above, this attitude may not be as easy to actually adopt as it is to appreciate in principle.

This problem was clearly illustrated during a prototyping session that was attended by the author. The session was progressing well until one user suggested some changes which challenged a part of the logical design. The designer responsible was also present at the session. Unfortunately, this particular part of the system was both one that had taken much effort to analyse, and one of which the designer was most proud. Despite the designer supposedly attending in a passive capacity, he was unable to contain himself. He quickly and forcibly did his utmost to quell the user's suggestions, and in so doing put at risk the entire prototyping process. This designer portrayed an attitude towards users that was potentially devastating to the prototyping situation.

A further examination of the selection of developer participants for prototyping can be found in Sections 4.1 and 4.2 below.

3.3 User Attitudes

The users who are to participate in prototyping need to be ready and willing to get meaningfully involved in the development of their systems. Hence, they must be prepared to dedicate the required amount of time to work in partnership with the developers. It is important that user participants are aware from the start of the significant amount of their time which will be needed for evaluation of the system prototypes. This also implies that users will play a greater role in development and consequently have an increased responsibility for its progress and eventual outcome.

The users must appreciate that the aim of prototyping is to enable the participants to learn about some aspect(s) of the required system. They must not expect prototypes, especially early exploratory ones, to be perfect operational systems. Conversely they must understand

that what appears to be a highly suitable system, may in reality be a rather fragile, incomplete prototype under its professional exterior.

Most of these problems are unlikely to occur if the users are familiar with the theory and practice of prototyping. It is particularly important that users appreciate that their input is crucial to the process. Without their evaluation and feedback prototyping is a waste of time and effort.

3.4 Attitudes Summary

The attitudes of all people involved in prototyping, managers, developers and users alike, contribute to the success or otherwise of the prototyping process. Sympathetic attitudes can be encouraged through a carefully designed and executed education and training programme. It is important that the participants are happy to work in partnership for the benefit of the organisation rather than for their own personal kudos. Prototyping which takes place within such a supportive environment, where creativity is not stifled, has an improved opportunity to succeed.

4. Selection of Participants

This involves selecting the number and make up of both developers and user representatives who are to participate in prototyping. Perhaps the most crucial decision to be made at this point is concerned with who is to play the role of the prototyper.

4.1 Selection of Prototyper

Those who have been involved in the early analysis and/or logical design are in a position of knowing a great deal about the system, as well as already having been involved with the users. An analyst/logical designer appears to be the obvious choice for the prototyper position. However, these people may have expended a great amount of time and effort in preparing the logical design and consequently they may have considerable "ego-involvement". It would be difficult for one of them to act as an impartial prototyper. There is a danger that the users could be persuaded that the logical design is as it should be.

Those developers who will be responsible for the physical design must also have a complete understanding of the system. However, it might be difficult for these people to act impartially as they may well be preoccupied with the implementation of the system. The danger here is that the users could be steered towards a particular conclusion because this was the simplest to implement.

Another possibility is to employ the person who is the most proficient with the chosen construction tool as the prototyper, in which case suggested alterations could be easily

incorporated. The main concern in this case is that this person may be a programmer who might lack the valuable experience of dealing with the user population.

The personal qualities of the prototyper are also very important. Ideally, this person must possess the qualities traditionally required by a systems analyst, including patience, diplomacy, perception, acceptability and objectivity. Perhaps the most important attribute is that the prototyper must not be regarded as intimidating by the participating users. Should this be the case, they may be dissuaded from making suggestions and criticisms, thus adversely affecting the feedback so crucial for productive prototyping.

4.2 Selection of Additional Developers

It may also be advisable to involve additional developers in the prototyping sessions, in order to provide support for the prototyper. A logical designer could be made available to provide clarification at the request of either the prototyper or the users. A physical designer could be made available for consultation in order to advise the prototyper of the feasibility of proposed changes. However, care must be taken that these developers do not defend or propose solutions so enthusiastically that it is to the detriment of the prototyping process. In practice it appears to be extremely difficult to attend a prototyping session in a purely passive capacity.

A further developer may be required to attend the prototyping sessions. This person would be responsible for keeping a written account of the session for control purposes, in particular to record all incorporated changes and suggested alterations. This should allow the prototyper to give his/her full attention to the user participants. The person performing this secretarial role need not necessarily be the same for every prototyping session. The use of video could negate the need for this additional developer participant.

4.3 Selection of Users

One concern during the selection of user participants for prototyping involves choosing between the management, those people who will be responsible for the system under construction, or choosing the end users, those people who will operate it, or in fact choosing any intermediate level of employee. It is, of course, possible to choose participants from more than one group of employees. However if this is the case, and the intention is that they should all attend the same prototyping session, there is a possibility that this may lead to a loss of objectivity among those involved. Selecting individual user participants is usually the responsibility of user management and will depend upon such criteria as status, experience, attitudes and enthusiasm.

There are at least two alternative ideas concerning the selection of user participants based upon their attitudes and levels of enthusiasm. Selection of appropriate users who are already interested in the system and who are keen to participate is hardly likely to fail. However, the selection of an unsympathetic, potentially antagonistic user, while being a little more risky, offers an excellent opportunity for improving relationships within the organisation.

The author has witnessed this phenomena in action. A series of prototyping sessions were organised and the participating developers chosen. User management were responsible for the selection of the user participants. Two of the four users selected, openly declared that they had no wish to be present, but that they had been "directed" to attend by their senior management. The prototyping began in an unfriendly and suspicious atmosphere. However, the attitude of these users gradually softened as they began to appreciate that what they had to say really mattered. Suggestions they made were incorporated into the prototype system. Their support and responsibility for the task they had been assigned grew rapidly, and at the end of the second prototyping session, one was heard to make a comment concerning "personalising our own system". The speed at which this complete change of attitudes had occurred was staggering. Much later, the same two users willingly adopted the roles of local "champions" for the new system. They were also vocal with their opinion that they would insist upon being involved, via prototyping, in any future developments in their area.

This may, in fact, indicate a further complication in the selection of user participants. Despite some users being reluctant to take part in early prototyping ventures, the situation could quickly change with many people wanting to participate. If participation in prototyping begins to be linked with status, or perhaps rewards, this could pose further selection decisions.

An alternative method of selection could be founded on lines similar to those used for the consensus approach to participative development [Mumford 1979], in a more democratic manner.

The number of participating users must also be established. The usual assumption is that prototyping takes place with one developer and one user, however, this may have disadvantages, as follows:

- it does not enable user debate;
- it does not encourage general system ownership;
- it could be intimidating for the one user;
- it relies upon the knowledge and abilities of an individual.

Conversely, the more users that participate in the prototyping sessions, the greater the practical problems, e.g. many people attempting to view the same VDU. It has also been suggested that having multiple user participants slows down each of the prototyping activities, hence it is preferable to develop a prototype with one typical user, and then to use it as a pilot or initial prototype with all other users [Jenkins 1983].

Another consideration is whether the same user participants should attend all the prototyping sessions, or whether to involve different people. Having one set of user representatives throughout prototyping has the obvious advantage of continuity, thus reducing the familiarisation effort for each session, and enabling the users to become quite adept at their prototyping role. Introducing a number of different users has the advantage of involving more people with the project development and so providing a greater wealth of knowledge, experience and ideas. This also provides the opportunity for a wider

understanding and appreciation of the system and consequently a wider sense of involvement, responsibility and ownership.

5. Roles Adopted

This section considers the roles that users and developers must play during systems development. It suggests that the users, in particular, are required to play a very different role when development incorporates prototyping.

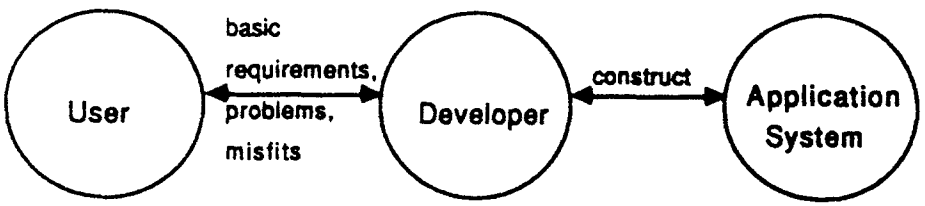

Figure 1 : Traditional Development Roles

Figure 1 shows a somewhat simplified view of the roles adopted during a traditional, prespecification approach to systems development. The provision of the application system is almost entirely the responsibility of the developers. Users are involved in a passive way, usually during the fact finding activities, and much later for acceptance testing. They participate when requested to do so by the developers. For the remainder of the time they can do little more than wait and pray. Meanwhile the developers are attempting to design, implement and test the system, to meet their own understanding of its requirements. More up-to-date structured methodologies have introduced numerous checkpoints at which the user is consulted. However, the participation remains at a relatively passive level, hence the user's role is little changed from that previously described.

The introduction of prototyping into systems development has a significant effect on the roles of both user and developer. The extent of the change depends upon the way in which the organisation adopts the approach. The roles shown in Figure 2 have been suggested as appropriate for prototyping [Naumann 1982]. The titles of both users and developers have been altered to more accurately reflect their new responsibilities. The user/designer is the active participant in this model having full responsibility for the design and evaluation of the prototype. The developer, who in this model has been reduced to a system builder, is now a relatively passive participant.

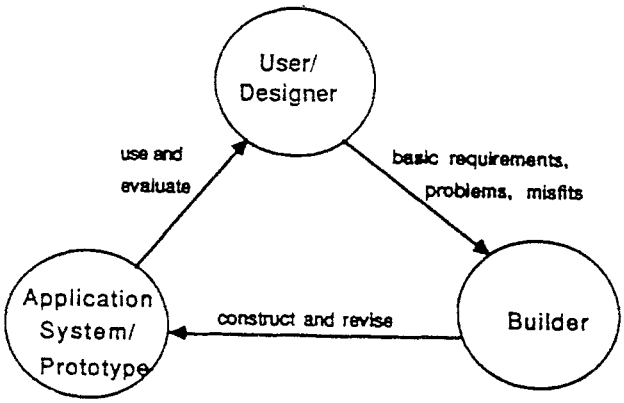

Figure 2 : Prototyping Rules - One Approach

The two previous models, although quite the opposite of each other, have an identical deficiency. They both fail to make full use of all those people involved. Figure 3 illustrates an approach to prototyping that draws upon the combined knowledge and experiences of both user and developer, here referred to as the prototyper. Users and developers form a partnership in order to develop prototypes, and hence learn about the required system. Users are required to play a far greater role in development. They are actively involved in the use and evaluation of prototypes. They may also build parts of them depending on the available tools and the type of prototypes being constructed, e.g. using report generators. In association with this greater role is an increased user responsibility, both for the progress of development and for the quality of the final product. Users are no longer merely the developers' customers.

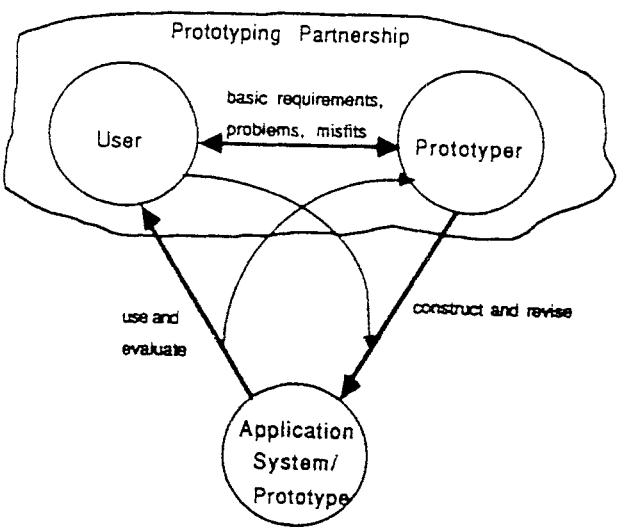

Figure 3 : Prototyping Roles - "Partnership Approach"

The prototyper takes responsibility for the majority of the prototype construction and modification effort. (S)he may also be involved in the evaluation process depending upon the types of prototyping being used, e.g. performance prototyping.

Co-operation in this prototyping partnership, emphasises the fact that both users and developers are important for the development of computer based systems. An effective partnership implies a shared responsibility as well as a mutual desire to make a success of both prototyping and the subsequent systems development.

6. Post-Prototyping - Implications

The intention of this section is not to reiterate discussions on the overall advantages and problems associated with prototyping, which have been covered in detail elsewhere, (see for example [Brittan 1980, Dearnley 1983]), but rather to highlight some of its implications for those people who have participated in the process.

As was mentioned previously, the majority of publications about prototyping are relatively theoretical. Those that do focus upon its practical application, generally concentrate on its affect on users, and comment less on the implications for managers and developers. The few empirical studies published in the area, also tend to concentrate on users' perceptions rather than to those of the other participants [Alavi 1984].

The experience related in Section 4 above, supports the widely held belief that prototyping appears to have a positive influence on the attitude of end-users towards both the development process and the information systems department. The example described two users who were initially unwilling to participate, yet who quickly became avid supporters of both prototyping and their system under development. Other authors who have published their own practical experiences of prototyping, agree that such positive influences really do occur, (see, for example, [Earl 1978]).

There is far less information available on the effects of prototyping upon developers and managers. There are few, if any, objective studies in this area, hence the points made below are based on experience and intuition.

The author attended a series of prototyping sessions at which the prototyper was a young, relatively inexperienced graduate. He had no previous experience of prototyping and at the outset had a limited knowledge of its concepts. Despite this, he agreed to take on the role. Since there was no possibility of him being overly influenced by years of experience using a more formal development method, he approached the task with a relatively open mind. Both his outward going personality and his friendly disposition proved invaluable during the prototyping sessions that followed.

When questioned about his experience after the prototyping was completed, the prototyper admitted that he had secretly been daunted by the prospect of demonstrating a system to multiple user representatives. However, once the sessions were underway he conceded that for most of the time he enjoyed working closely with the users. He had necessarily become far more involved with the detailed requirements of the system, than with any of his previous programming work. While he considered this increased level of involvement as

more taxing, he readily admitted that a greater appreciation of the whole problem increased the enjoyment of his work. However, he also made the remark that he looked forward to getting back to some simple programming.

The main problem highlighted by this prototyper was that at times he felt rather vulnerable. He was referring to what he saw as a lack of a comprehensive support environment; there were few methodological guidelines to which he could turn for direction. The project manager, who like the prototyper had no previous experience of prototyping, also commented on the relative lack of guidelines to support his own role in the process. Perhaps this is an inevitable problem for an emerging approach, however, it is one that needs addressing in the near future.

Hence from the experience outlined above, it appears that the reactions of the developers, in particular those of the prototyper, are likely to be significantly influenced by the level and quality of available support for prototyping. Implications are likely to vary substantially depending on the personality of the prototyper.

Observations concerning the implications for management can also be taken from this same prototyping application. In the beginning, user management somewhat reluctantly agreed to the software developer's proposal to utilise prototyping during the early stages of development. However, when they began hearing the user participants enthusiastic reports of the prototyping, they became more interested in the process and made arrangements to attend the next session. They were delighted by the way in which their users were actively involved in the process, particularly as they had originally anticipated possible opposition to the new system. However, they were more guarded than their subordinates in their comments concerning the use of prototyping in the future.

If prototyping has been successful, (i.e. having developers working with users, in such a way as to enable their active, meaningful and satisfying participation, in the learning process), it encourages an improved organisation-wide attitude towards systems development. Future systems are likely to be designed and introduced with high levels of competence because the users are now better informed and educated concerning the application of technology within their organisation. At the same time the developers have improved their understanding of the activities of the user departments [Land 1983].

7. Summary

Prototyping involves far more than choosing, and subsequently using a tool to build working models of a system. Successful prototyping requires substantial preparation, as well as careful organisation and management. Many important decisions must be taken concerning the people who are to participate in the prototyping. The effectiveness of prototyping is largely dependent on the individuals involved, in particular the prototyper and the user representatives. Hence it is crucial that proper care and attention is given to those issues concerning the prototyping participants.

A large number of articles have examined the potential advantages of prototyping, e.g. reduced development effort, reduced testing time, more effective solutions, fewer maintenance requests, greater user satisfaction. There are additional benefits, however,

which despite being difficult, if not impossible, to quantify, are nevertheless extremely important. These benefits stem from the positive effect that prototyping so often has on its participants. Working closely, with a single objective to learn more about the required system, both users and developers can expand their knowledge and improve their mutual understanding. This fosters an atmosphere of co-operation between the systems developers and the user community, which can only improve the chances of successful developments in the future.

It is essential that organisations are aware of the possible implications of prototyping on those people involved. Such an appreciation enables the organisation to plan for prototyping that will take full advantage of all the extremely important potential benefits described above.

8. References

[Alavi 1984]
Alavi, M., "An Assessment of the Prototyping Approach to Information Systems Development", Comm. ACM, Vol.27, No.6, June 1984.

[Amabile 1989]
Amabile, T., "Improving Creativity in Systems Design", Panel 15, ICIS 89, Boston, December 1989.

[Andersen 1990]
Andersen, N,E. et.al., "Professional Systems Development: Experience, Ideas and Action", Prentice-Hall, 1990.

[Boar, 1984]
Boar, B.H., "Application Prototyping: A Requirements Definition Strategy for the 80's", Wiley, 1984.

[Brittan 1980]
Brittan, J.N.G., "Design for a Changing Environment", Computer Journal, Vol.23, No.1, 1980.

[Cutts 1987]
Cutts, G., "Structured Systems Analysis and Design Methodology", Paradigm, 1987.

[Dearnley 1983]
Dearnley, P.A. and Mayhew, P.J., "In Favour of System Prototypes and Their Integration into the System Development Cycle", Computer Journal, Vol.26, No.1, 1983.

[Donovan 1977]
Donovan, J.J. and Madnick, S.E., "Institutional and Ad-Hoc DSS and Their Effective Use", Database, Vol.8, No.3, Winter 1977.

[Earl 1978]
Earl, M.J., "Prototyping Systems for Accounting, Information and Control', in "Accounting, Organisations and Society", Vol.3, No.2, 1978.

[Floyd 1984]
Floyd, C., "A Systematic Look at Prototyping", in "Approaches to Prototyping", Eds. R. Budde, K. Kuhlenkamp, L. Mathiassen, H. Zullighoven, Springer-Verlag, 1984.

[Harker 1988]
Harker, S., "The Use of Prototyping and Simulation in the Development of Large-Scale Applications", The Computer Journal, Vol.31, No.5, 1988.

[Jenkins 1983]
Jenkins, A.M., "Prototyping: A Methodology for the Design and Development of Application Systems", Indiana University, Graduate School of Business, Discussion Paper 277, 1983.

[Land, 1983]
Land, F. and Hirschheim, R., "Participative Systems Design: Rationale, Tools and Techniques", in Journal of Applied Systems Analysis, No.10, 1983.

[Law 1985]
Law, D., "Prototyping: A State of the Art Report", NCC, 1985.

[Mayhew 1987a]
Mayhew, P.J. and Dearnley, P.A., "An Alternative Prototyping Classification", The Computer Journal, Vol.30, No.6, 1987.

[Mayhew 1987b]
Mayhew, P.J., "An Investigation of Information Systems Prototyping", Ph.D. Thesis, UEA, Norwich, UK (1987).

[Mayhew 1989]
Mayhew, P.J., Worsley, C.J. and Dearnley, P.A., "Control of Software Prototyping Process: Change Classification Approach", Information and Software Technology, Vol.31, No.2, March 1989.

[Mayhew 1990]
Mayhew, P.J. and Dearnley, P.A., "The Organisation and Management of Systems Prototyping " to be published in Spring 1990, in "Information and Software Technology".

[McCracken 1981]
McCracken, D.D., "A Maverick Approach to Systems Analysis and Design", in "Systems Analysis and Design: A Foundation for the 1980's", Elsevier-North Holland, 1981.

[Mumford 1979]
Mumford, E., "Consensus Systems Design: An Evaluation of this Approach", in "Design and Implementation of Computer Based Information Systems", Sijthoff and Noordhoff, 1979.

[Naumann 1982]
Naumann, J.D. and Jenkins, A.M., "Prototyping: The New Paradigm for Systems Development", MIS Quarterly, Vol.3, September 1982.

[Riddle 1984]
Riddle, W.E., "Advancing the State of the Art in Software Systems Prototyping", in "Approaches to Prototyping", Eds. R. Budde, K. Kuhlenkamp, L. Mathiassen, H. Zullighoven, Springer-Verlag, 1984.

[Vonk 1990]
Vonk, R., "Prototyping: The Effective Use of CASE Technology", Prentice-Hall, 1990.