Prototyping — a Technique for Participative Information Systems Design

David W. Wilson
Hong Kong Polytechnic, Department of Computing, Hung Hom, Kowloon, Hong Kong

Prototyping as a technique in Information Systems Development (ISD) is introduced and scrutinised. A number of taxonomies of ISD Prototyping are reviewed in the context of a some ISD process enactment models. Three contingencies, innovativeness, dialectic style and organisational affectedness, are suggested in addition to those promoted elsewhere for selecting development process paths. The need for Participative Systems Design is established and Hirschheim's classification of Participative Systems Design methodologies is used to examine whether Prototyping can be used with the methodologies in question. A research agenda is suggested for verifying the effectiveness of prototyping for participative Information Systems Development.

Keywords: Prototyping, Participation, Information Systems Development

1. Prototyping and Information Systems

Prototyping is an ancient Engineering Strategy. The concept is that the agent taking responsibility for some development will build a working model to explore some facets of the proposed end product. In Manufacturing Engineering it is usual that following the prototyping phase thousands of replications of the object will be created. In Civil Engineering models of artefacts to be instantiated only once are often built but whether these are regarded as prototypes or models is open to discussion. The difference between a model and a prototype is more subtle than is at first apparent and for the Information Systems (IS) community the difference may be something of a nuisance. This is because a great number of our "products" are in fact models of the real world that occasionally produce real objects such as invoices or signals. Almost invisible, intangible reification of the model into Information is more usual in IS than the opposite. Since information can only be received by cognisant beings we never actually see or touch information, only it's representations. What we are able to prototype, therefore, is merely the representation of Information. Both Zemaneck (Zemaneck, 1970) and Stamper (Stamper, 1987) throw considerable light on the applicability of semiotics (particularly, the sub-disciplines of pragmatics, syntactics and semantics) to the understanding of the process of giving data to, and receiving information from, computer systems. Prototyping is an experimental technique to provide a tangible basis for discussion of an Information System with legitimate stakeholders (the demand side) and /or to increase the confidence of the supply side that the envisioned system will operate within required performance criteria.

Information Systems Prototyping may be defined as an experimental technique whereby uncertainty is iteratively reduced. An Information Systems prototype is an experimental Information System evolved by an Information Systems expert to determine social, technical or socio-technical feasibility in areas of the Information System where there is reasonable doubt. It is useful to differentiate prototypes from templates. Neither prototype nor template is a fully satisfactory system. The difference of the prototype from the operational system is that areas confidently expected to be trouble free, especially technical areas may be ignored or simulated with operationally inadequate constructs. A template is a candidate object developed by an Information Systems expert to be specialised by the user to fit his particular problem without concerning himself with the linking to the technical platform. This paper will confine itself to exploring the use of prototyping in Participative IS Development (ISD).

Prototyping in ISD may be carried out in a number of ways. Commonly a working model of the system, or part of the System, will be built using powerful tools. Some (Ince and Hekmatpour, 1987) identify a wide range of those including very high level languages, application oriented very high level languages, functional languages, transformational programming and tool-sets. A commonly used tool set in commercial ISD leading to transaction processing systems and Management Support Systems consists of a minimum of a Data Base Management System, Screen Painter and Report Generator. Prototypes are rapidly and iteratively generated until satisfaction is reached in the dimension of the experimentation. The dimension may be social (business or organisational user satisfaction or indication of cost-oriented trade-offs) or technical (satisfactory component interaction or performance.) Apart from social benefits it is often claimed that prototyping leads to rapid delivery of the final system by reducing the human effort in the early stages of project definition. For instance, the prototype is self documenting or obviates the need for detailed documentation. It focuses the development on the user's concerns. Further, it is responsive to uncertainty and variability through its innovative and evolutionary aspects. An actual contribution to the eventual system, other than the specification, may be carried forward from the prototyping activity and the opportunity may be taken to evaluate possible variations in hardware configuration or the performance available. Product "quality" is maximised if product "quality" is defined as the closeness of the fit between the implemented system and it's objectives and the implementation environment. Project snags, such as incompatibility of hardware and software or timing cycle bottlenecks may become obvious during the prototyping cycle.

All projects run risks. These may be classified as financial (failing to benefit the commissioning organisation more than they cost), temporal (failing to be in place in time to be optimally useful), and social (being disruptive to the social fabric of the organisation.) Protoyping may be justifiably claimed to reduce all three. Normally the main advantages of prototyping occur in the reduction of social risks. Because a tangible object is before the participants in the developer/user dialogue the dialogue is enacted in tangible terms greatly reducing misconception and ambiguity where previously a great deal of description and discussion would have been generated. Since both parties may be shown to communicate sincerely and sensibly mutual esteem may be fostered. Problems can be immediately spotted, discussed and embryo solutions posed "on the fly". The developer will be encouraged to develop the system in an open-ended fashion to minimise his task of facilitating repeated changes. User input is maximised and developer's misconception minimised. Since user's have an intimate experience with the system during its development the need to subsequently train user's is greatly reduced and the need to "sell" the eventual system to the user is obviated since they will have come to regard it as their system anyway. Further, through the iterative interaction with the system the tendency of some users, particularly older and senior staff, to experience tension when using the system in public, is reduced.

Davis and Olsen (Davis and Olsen, 1984) citing Naumann et al (Nauman et al, 1980) point out that prototyping is inappropriate in situations of low complexity. It may also seem inappropriate in situations where some stakeholders are likely to lose out either in terms of internal organisational positioning (power) or even retaining a position in the organisation. It is probable that nearly every significant change to an organisation causes one or both of these effects adversely to some stakeholder. Provided trade-offs can be made, prototyping can provide an effective vehicle for open dialectic between managers and stakeholders. The role of the information systems specialist in such circumstances is to make clear to both parties possible consequences of various options.

Prototyping is a technique that focuses on effectiveness, the closeness of the solution fit to the problem environment, rather than efficiency. It is likely that there will be tools and methods available that render a more efficient product than the prototype as the system eventually implemented. Risk analysis will probably produce a settling on a halfway house where heavily used parts of the system are optimised but the laws of diminishing returns will eventually dictate that medially and less used routines are not worth optimising. Hence it is a reasonable objection that even an optimised system will be delivered in a less efficient state than one that has been designed from the outset for highly efficient environments.

Perhaps a more worrying fault of prototyping is its effect on the user and the developer. It is perfectly legitimate that the political imbalance in development whereby developers all too readily act like a kind of priesthood, asserting what can and cannot be done, without intelligible explanation should be torn down. However, if prototyping is not approached intelligently and professionally there can be negative effects to both parties. The developers may fall into a less than professional stance, that since responsibility for the system is shared all responsibility should be taken by the user. They may tend to bring problems they are aware of, to his attention reluctantly or not at all in a mistaken belief that when the user's system fails their priesthood will be restored.

The effect on users also relates to the shift in the balance of power. Often some of the users will be in powerful positions in the organisation. They will have been involved in a process where what they see of IS (the interface) has been repeatedly rapidly changed. They are unlikely to appreciate that what they have witnessed is not a full system and will not appreciate the complexity of that which they have never seen. For instance the security and backup systems are likely to be in place only in the most rudimentary fashion. Only standard transactions are likely to be catered for and peripheral functions may not have been built. Since the commissioning user has witnessed rapid development he may fail to appreciate why all development can not proceed at the pace he has witnessed. Worse still he may insist on the "temporary" use of an insecure system "for the time being" and later cancel the "enhancement". Developers educated mainly with technical skills may lack the social skills to persuade him of his folly.

Further there is a tendency for the user to make design decisions "on the fly" without much consideration of the cost of development or running. The developer may not be fully aware of cost implications as design decisions are made and the tendency to ignore cost implications may become part of the prototyping design culture. All participants need to be able to recognise their wilder dreams as such and when the costs will heavily outweigh the benefits both tangible and intangible.

A dysfunctional behaviour that prototyping may engender is the tendency to strive for perfection regardless of cost. This may be exhibited by either the developer, the user or both. Some practitioners advocate the setting of an arbitrary number of prototyping sessions or better still an estimate based on the uncertainty being faced. Of course, by definition, we can only estimate degrees of uncertainty. Both developer and user should enter into prototyping sessions with a non-perfectionist attitude and an awareness of cost.

There are several classifications of prototyping. Ince and Hekmatpour (Ince and Hekmatpour, 1987) classify by end-state, naming the end-states as "throw-it-away", evolutionary and incremental. As the name implies with throw-it-away prototyping the prototype is thrown-away after use as a requirements definition. Evolutionary protoyping implies that the prototype is refined through several iterations whereas incremental prototyping is the process of adding to the implemented product. Saxena (Saxena, 1988) presents a "T" shaped categorisation: horizontal prototyping providing the full functionality of the final system but ignoring performance issues as opposed to vertical prototyping where the implementability of parts of the requirement are explored before an attempt is made to implement the whole system. Mayhew and Dearnley (Mayhew and Dearnley, 1987) develop six categories (exploratory, experimental, performance, hardware, ergonomic and functional) in three classes (exploratory, experimental and organisational)to form an elegant pyramid (see fig. 1.) the corners of which represent the prototyper, the user, hardware and software. We will use the views of this triangle from the Prototyper (P) and the User (U) to evaluate Prototyping for participation.

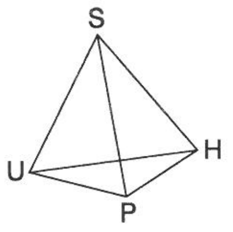

Fig. 1. Mayhew and Dearnley's "PUSH" Pyramid

The ISD process, or ISD Life-cycle, has traditionally been enacted following a "cascade" model. Wilson's (Wilson, 1990) representation of this well-known model (fig 2) shows its relationship to strategic planning and project management.

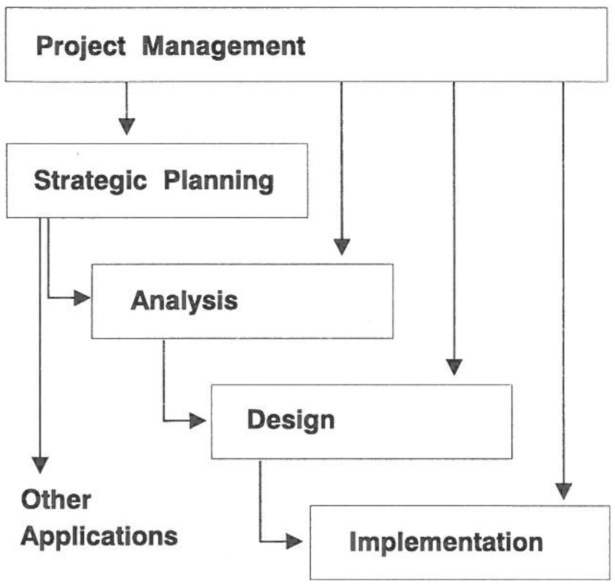

Fig. 2. The "Cascade" Process

Nauman, Davis and McKeen (Nauman et al, 1980) cited in Davis and Olsen 84 (Davis and Olsen, 1984) present a set of contingencies (project size, degree of structuredness, developer task comprehension and user task comprehension) for determining whether this cycle should be enacted straight through, iteratively or with a prototyping element. Nauman et al's (Naumann et al, 1980) model was developed for choosing a requirements assurance strategy. Where prototyping is the candidate strategy three other contingencies would appear to gain importance. Some would point out the importance of the degree of innovativeness which can be taken to mean the amount of change from the existing system. This is focusing on the immediate task distinct from Nauman et al's description of their contingencies "User task comprehension" and "Developer task proficiency" where the concern is with how many similar tasks either party has been involved in before. Another would be the dialectic style on a scale between Democratic and Dominated. This is often a reflection of the cultural background of the parties involved but may also reflect the political forces extant in their societies.

The other contingency to be considered can be described as the affectiveness. To what degree is the host organisation affected by the change process and how will the changes be manifested. On the negative side questions such as:

- will there be a reduction in staff beyond what can be coped with in an acceptable manner (eg natural wastage);
- will there be an effect on the promotion prospects of the staff overall;
- will the relative positions of the staff change;
- will the staff find their lives controlled by the automata.

Conversely, if the possibility exists to develop systems where staff are empowered to concentrate more effectively their efforts in situations of power over the automata then prototyping as an emancipatory vehicle would be indicated.

Mayhew and Dearnley (Mayhew and Dearnley, 1983) add a prototyping cycle to the cascade cycle, re-entering the cascade either at Analysis, Design or Detailed Design. Boehm (Boehm, 1988) proposes a spiral model integrating the activities of the cascade with periods of risk analysis, followed by prototyping.

2. Why Participation?

Hirschheim (Hirschheim, 1986) cites Lucas (Lucas, 1974) to assert the following about the effects of participative approaches to ISD.

Participation is ego enhancing. The user's self esteem will be built up in the process leading to positive attitudes to the proposed new system. It might also be added that enhanced user self-esteem creates a confident user happy to bring his ideas forward for scrutiny in a non-hostile discussion.

Participation can be challenging and intrinsically satisfying leading to positive attitudes to the proposals. It typically results in more commitment to the change process, likely leading to greater system usage. Through participation users become knowledgeable about the changes. This obviates much of the training needs at implementation.

Participation allows users to retain control over their activities leading to the maintenance of positive attitudes. Land & Hirschheim (Land & Hirschheim, 1983) add that participation allows the individual to safe-guard his own interests. Further it provides a mechanism through which individuals can use the system to redesign their own jobs and working environment. This is vital to the functioning of the system as in a fundamental insight Land and Hirschheim (Land and Hirschheim, 1983) assert that ultimately "activities are controlled by those who perform them".

Hirschheim, building on the foundations laid by Enid Mumford (Mumford and Weir, 1979), finds grounds for support to be ethical. It behoves us not to design the lives of others without their consent. Further participation improves decision making concerning proposals because the user's expertise in the problem domain can never be surpassed even through the most thorough analysis. Affective and contextual information pertinent to the problem area can not be discovered through techniques such as interviewing or process, data or state modelling. At the same time participative methods tend to ensure end-user commitment through the sharing of responsibility for the target system.

However, problems with participative methods are reported by some. It is recognised that they can lead to political problems where the solutions inevitably lead to changes in the operational social structure such as loss of employment, role reversals or enslavement to the automata. Some would assert that it is inherently manipulative — this assumes that the facilitator has specific motivations other than the realisation of the most effective system. Croisdale (Croisdale, 1982) is reported to have asserted that it is extremely difficult to operationalise in the kind of situation he was faced with — thousands of geographically dispersed users. The assertion that it leads to sub-optimal solutions seems to contend that an ineffective efficient solution is somehow more desirable than an effective solution. It is further charged that participation is highly time-consuming due to the numbers of people involved in meetings.

Hirschheim classifies participative approaches in two dimensions and presents an instantiated matrix. Figure 3. is an approximation of his matrix. The dimensions he selects are a). content — the kind of decisions the methodology users are involved in (i.e. purely social / socio-technical / purely technical) and the form of participation (i.e. consultative, representative, consensus.)

Hirschheim has carried out qualitative field study reseach using this taxonomy interviewing DP managers, Systems Analysts, Sponsors and End-users involved in the use of Participative methods. He found that positive feelings about participative design are widespread and it appears to be widely applicable. It was thought to promote an organisational learning experience.

The question arises — is prototyping an applicable technique in all sectors of Hirschheim's

Consensus	EUC & 4GL	Participative	Trade Union Based
Representative	Structured		
Consultative	"Clasic"	Socio-Technical	Organisational Development
	Tehnical		Social

Fig. 3. Hirscheim's Classification of Participative Approaches

taxonomy and if so what form might such prototypes take?

Classic approaches are "classic" because they have been used for some time. Exemplified by Lee (Lee, 1979) a historical perspective can be found in Avison and Fitzgerald (Avison and Fitzgerald, 1988). There is some claim that these approaches contained prototypes in the form of screen, form and print layout charts but this claim falls down if a prototype is taken to mean a working model. It is the author's experience (from semi-structured interview, "intern" supervision and multiple occasional questionnaire surveys) that the approach is still widely practised; the essence of the approach being a managerial user treated as expert (or at least authority) and a concentration on technology — automating aspects of the activity. In recent years the forms have been replaced by screen painters and report generators. Usually these have been used in association with a Database Management System. Whilst there hasn't been a conscious effort to improve the social aspects of the participation the user / developer dialogue has been greatly enhanced. In terms of Mayhew and Dearnley's (Mayhew and Dearnley,1987) pyramid, hardware is often a given in projects using this approach so decision making concentrates entirely in the triangle PUS. Interestingly a senior development manager in a highly respected, unique transaction processing installation reported that a contingency approach was used to decide whether prototyping was to be used on a project. One contingency used in this organisation was not to use prototyping on systems where several departments had an interest as discussions were believed to become too complex.

In ISD the socio-technical methods are exemplified by Mumford and Weir (Mumford and Weir, 1979) although the principles have been applied with other technologies. The essence is the balance in the dialogue between the technician and the users and the attempt to find a best fit. At various stages of this method proposals are made with a conscious social awareness and with a conscious technical awareness. It can easily be seen that substitution of these proposals by prototypes would provide excellent focus for the decision making processes. The prototypes might be brought forward from both sides of the discussion, one side bringing instances from the triangle PSH and the other from USH of Mayhew and Dearnley's pyramid.

Organisational Learning is by definition mostly concerned with the social aspects of the organisation as a system. Prototypes of functions and interfaces would be excellent vehicles for the exploration of this kind of development.

The structured approaches are essentially modelling methodologies. A great reliance is placed on ensuring correctness by cross-checking between models and checks that quickly identify model completeness. SSADM is a structured methodology that is evolving through several versions, rather like the software whose development it supports. At one point Business Systems Options are presented to users in the form of models and Prototypes of the Man/Machine interface are recommended . This activity might be improved as a prototyping session verifying the functionality of the system. The methodology has, over the years, attracted the support of a number of CASE tools. An early tool specifically for this methodology allowed the building of a user dialogue model that could then generate a prototype of the dialogue.

The trade union based methodologies are most commonly found in the social democracies of Scandinavia. Both representative and consensual forms of participation are catered for. Whether prototypes are useful in these contexts is to some degree contingent on the political nature of the dialectic. If the system is to be used as a bargaining item it matters little how it is represented. If there is a genuine attempt to foster industrial democracy (which there often is) then prototyping must be viewed as the most effective way of promoting meaningful dialogue about the proposed change with all classes of user see (Olle et al,1982.) for a pertinent user classification.

In the arena of End User computing the role of the developer becomes that of the facilitator of an environment in which the user can explore problems using the technology and database. It is the most emancipated form of computing — the user drives technology subordinated to solving his problem. Prototypes are not appropriate since there is no way to predict what tomorrow's problems will be. Early versions are therefore not appropriate. Problems can be viewed as "late-bound" to an exploratory solution workbench. A similar mechanism however, can be provided and that is the template — a device to support learning and avoid repetitive input.

3. Conclusion

The question addressed by this paper may be stated as:– to what extent is Prototyping an adequate method of involving users in ISD? By taking Hirschheim's taxonomy and Dearnley and Mayhew's model it has been found that for most approaches to ISD Protoyping is the most appropriate technique of involving users of various kinds. In classic and structured approaches the manager is most likely to meaningfully explore and define the user interface. In socio-technical and participative approaches the dialogue and functional aspects of a prototype could usefully form the focus of dialogue between the parties. Since, Hirschheim's description of Organisational Learning approaches pre-supposes an emphasis on the social aspects of the system the only way prototypes can enhance the situation would be as learning aids — a role to which they are most suited.

With trade-union based approaches, again prototypes would act as excellent dialogue enhancing devices whether used with representatives, operational users and their supervisors or indeed customers. It is only in the most emancipated situation, where the developer is subordinated to the role of facilitator and users have full control that we find that prototypes have no place.

In theory it is posited that the recent emergence of tools facilitating the rapid development of systems should be used to involve users more closely in the development of more effective and closer fitting systems. Research needs to be carried out verifying organisational satisfaction with systems developed through user centred prototyping as against processes aimed mainly at rapid implementation. Further we need to investigate whether practitioners are in fact finding difficulty with the iterative cycle. Are arbitrary limits set? Does the iterative nature of the process cause unacceptable project management problems? There is a need for cross cultural research to verify the role of dialectic style in requirements capture. It would appear that prototyping facilitates open discussion, enabling the surfacing of full requirements but whether this is possible in cultures or organisations where open discussion is not commonly practised needs to be discovered by investigation. Hirschheim's classification is a useful framework for seeking comparative case studies if organisational experience of using Information Systems prototyping can be found.

References

AKTAS, Z., Structured Analysis and Design of IS", Prentice–Hall, Englewood Cliffs, New Jersey (1987).

AVISON, D. E. AND FITZGERALD, G., ISD: Methodologies, Techniques and Tools, Blackwell, Oxford (1988).

AVISON, D. E. AND WOOD–HARPER, T., Multiview: An Exploration in ISD, Blackwell, Oxford (1990).

BOEHM, B., A Spiral Model of Software Development and Enhancement, *Computer*, 5 (1988), 290–302.

CROISDALE, D., Contribution to Seminar on Participative Methods at Civil Service College, London, 1982

DAVIS, G. B. AND OLSEN, M. H., Management IS: Conceptual Foundations, Structure and Development, McGraw–Hill Series in Management IS, ISE (1984).

HIRSCHHEIM, R., Participative Systems Design: User Experiences, Evaluation and Conclusions, *Australian Computer Journal*, 18, 4, (1986) 166–173.

LAND, F. AND HIRSCHHEIM, R., Participative Systems Design: Rationale, Tools and Techniques, *Journal of Applied Systems Analysis*, 10 (1983), 91–107.

LEE, B., Introducing Systems Analysis and Design, Volumes 1 & 2, NCC, Manchester (1979).

LUCAS, H. C., Information Systems Concepts for Management, 2nd ed., McGraw–Hill, New York (1982).

MAYHEW, P. J., AND DEARNLEY, P. A., In Favour of Systems Prototypes and their Integration into the Systems Development Life Cycle, *Computer Journal*, 26, No 1 (1983), 36–42.

MAYHEW, P. J. AND DEARNLEY, P. A., An Alternative Prototyping Classification, *Computer Journal*, 30, No 6 (1987), 481–484.

MUMFORD, E. AND WEIR. M., Computer Systems in Work Design: the Ethics Method, Associated Business Press (1979).

NAUMAN, J. D., DAVIS, G. B. AND MCKEEN, J. D., Determining Information Requirements: A Contingency Method for Determining Requirements Assurance Strategy, *Journal of Systems and Software*, 1 (1980).

OLLE, W. T., SOL, H. G. AND VERIJN–STUART, A. A., Is Design Methodologies: a Comparative Review, North–Holland, Amsterdam (1982)

SAXENA, K. B. C., IS Prototyping, Hong Kong Polytechnic Technical Report, TR–8–03 (1988)

STAMPER, R., "Semantics" in "Critical Issues in Information Systems Research", Ed. R. J. Boland and R. A. Hirschheim, John Wiley Information Systems Series, 1987

TRIST, E. L., HIGGENS, E. W., MURRAY, H. AND POLLOCK, A. B., "Organisational Choice, Tavistock, London (1963)

WILSON, D. W., What is CASE?, *Hong Kong Computer Journal*, Vol6, July (1990), 18–20.

ZEMANEK, H., "Some Philosophical Aspects of Information Processing" in "The Skyline of Information Processing", ed Zemanek H., North-Holland, 1972

Received: January, 1993
Accepted: October, 1993

Contact address:
David W. Wilson
Hong Kong Polytechnic,
Department of Computing
Hung Hom
Kowloon, Hong Kong
Tel. (852) 367 4914
Fax: (852) 367 4918
email cswilson@comp.hkp.hk

MR DAVID W. WILSON has taught Information Systems Development Methodologies at Hong Kong Polytechnic both as Lecturer and Senior Lecturer since 1979. He played a leading role in developing and implementing the Department's Masters of Science in Information Systems and teaches on the Polytechnic's MBA and Warwick University's IGDS Masters Programme. Prior to his career in Education he spent 11 years developing Information Systems for the UK Government significantly on a leading-edge database project amongst others. He holds a Post-Graduate Diploma in Information Systems Analysis and Design from the London School of Economics and is a Member of the British Computer Society.