Transactions on Information and Communications Technologies vol 9, © 1994 WIT Press, www.witpress.com, ISSN 1743-3517

PROCONCEPT: an evolutionary prototyping methodology for managing systems development

A. Howard

Faculty of Information and Engineering Systems, Leeds Metropolitan University, Leeds, LS6 3QS, UK

ABSTRACT

In spite of its well-documented advantages over other systems development approaches, prototyping is still viewed in many commercial organisations as an unsuitable approach for large-scale systems development. Many managers prefer to stick to traditional lifecycle project phases with a detailed user requirements document as the starting point for the development effort. Prototyping is seen as an unstructured exercise which is difficult to plan and control. In order for the approach to gain wider acceptance the discipline of a methodology is required. Methodologies provide frameworks for the management of projects and enhance the quality of both the system development process and the software products produced. This paper discusses the background to, rationale for, phases, tasks and deliverables from a proposed evolutionary prototyping methodology.

INTRODUCTION

Prototyping is an approach to system development which attempts to demonstrate how a system or component of a computer-based system will function in its environment. Users find such demonstrations extremely helpful in visualising what proposed systems will do for them. Developers find them helpful in establishing the feasibility of construction of a system or system component. Several authors (e.g Jenkins [1], Gomaa [2], Iggulden [3], Rowen [4]) refer to the difficulties experienced by users in visualising on the basis of a system

specification document what systems will do for them. Jenkins takes this further by asserting that one explicit assumption of the prototyping approach is that people can tell you what they don't like about an existing system more easily than they can tell you what they think they would like in an imaginary one. This assertion has been well-corrobated by failures in the traditional SDLC. Not knowing what can be done, and how easily or otherwise, leaves users at a disadvantage when attempting to describe what their system should do.

Prototyping restores users to a position of influence where they have greater involvement in system development. It has been used in three different areas:-

1. To assist in understanding, communicating and specifying user requirements

2. To verify the feasibility of system design

3. As an iterative tool in system development where the prototype gradually evolves into the final product.

Prototyping is strong in areas where other methodologies are weak. It supports the end-user as well as the development professional. West [5] identifies five key factors which highlight the advantages of the prototyping approach. The factors are collaboration, communication, conceptualisation, change and consensus. Collaboration in order to forge a partnership between users and developers which focuses on producing quality, useful systems and avoids antagonism and contractual arguments. Communication so that developers and users understand each others problems and speak the same language. Conceptualisation to help users and developers agree conceptual models of the required system(s). Change is frequently seen as something to be avoided in system development. In reality it is an inevitable consequence of turbulent business and real-world environments and has to be embraced into systems development philosophy. Consensus in system development has usually been based on specification documents and has frequently been reached to neither developers nor users ultimate

satisfaction. Appleton [6] points out that prototyping incorporates the process of discovery into systems development. It assumes that the pre-specification of exact requirements is not always possible as they are increasingly dynamic and a certain amount of trial and error is required to get things right.

In spite of the advantages of prototyping over the sequential development process its usage has been restricted by a number of factors. These include:

1. Sceptism about the approach
2. Problems over the system specification document and its role in the prototyping process
3. The lack of a suitable prototyping methodology and supporting tools
4. The difficulty in managing a dynamic and flexible system development approach
5. The lack of commitment to prototyping by senior development staff
6. Difficulties in integrating prototyping into existing system development methodologies

It is felt that most of these objections can be overcome by the development of a coherent prototyping development methodology. The methodology needs to be teachable, manageable, acceptable to project sponsors, users and developers, and provide a framework for the selection and development of support tools.

SCOPE OF PROTOTYPING

A number of reasons have been put forward for the development of prototypes. The most frequent arguments used refer to:-

* Evaluating the feasibility of proposed systems
* Identifying and verifying user requirements

* Improving the understanding of user issues

* Evaluating the performance characteristics of technical issues

* Encouraging management and user support by demonstration of a prototype and its capabilities

Blum [7] discusses five areas in which prototypes can be used to gain experience for system developers. The areas are user interaction, systems flow, data models, algorithms and user involvement. Blum points out that users often have a poor understanding of how a process is performed for them by the computer-based system they are using. Prototypes can serve as a vehicle for defining and validating the algorithms they rely on. Data models can be prototyped to check for completeness, access paths and current and future information requirements. User interaction and involvement act as catalysts for the developer to discover the real needs of the system and to ensure enthusiastic user acceptance of the outcomes. Comprehensive understanding of systems flow enables the identification of principal events, unexpected events and likely bottlenecks in systems utilisation. Each flow can be tested for robustness and unexpected outcomes.

Crinnion [8] identifies five types of prototype each with different levels of functionality. For each type of prototype a number of iterations and versions are required in the system development process. The first type of prototype is an initial prototype concentrating on the HCI and basic functionality. The second is a completeness prototype which includes the results of additional analysis to ensure that all user requirements have been identified. A controls prototype then follows which includes design decisions resulting from a controls audit. Fourthly there is a combination prototype which links several separate prototype models, and finally a performance prototype optimised to give the best operational performance.

The scope of prototyping has also been discussed from a number of perspectives not directly connected with prototype construction

but important nevertheless in terms of an overall appreciation of the possibilities for the approach. Earl [9] argues that part of an information systems department's budget should be allocated for prototyping ideas from an organisation's staff. The best applications ideas may not come from centralised strategic planning but from experimentation. Mason [10] suggests that prototyping could be used for establishing the future computing environment for an organisation. Using scenario-based prototyping the most appropriate corporate-wide IT environment could be modelled. Tillman [11] argues that although prototyping is user focused, some users are not able to pinpoint the underlying principles of a system or to see it in a wider context than their own narrow view. The solution in his view is to have management communicate their overall vision of the system. Without this vision we may make the same mistakes in prototyping as are made in other methods. These mistakes include automating manual processes which are fundamentally poorly designed for automation, customising processes without consistency, and perpetuating current organisational structures. Sobol and Kagan [12] published some research into which system developers are most likely to prototype. Not surprisingly perhaps they found that younger development staff are more likely to prototype. They also found a correlation between supervisory responsibilities and use of the approach. Staff with less management responsibility were more likely to prototype, possibly pointing to a recognition of the difficulties in managing the approach. Prototypers were also found to be much more user-oriented, reinforcing the essential user involvement aspect of the approach.

PROBLEM CONTEXTS FOR PROTOTYPING

Most system developers accept that there is no single development approach or methodology that is applicable in every project situation. Each project has its own characteristics which render it suitable or unsuitable for one or other development method. Different methods emphasise different aspects or phases of the development process. Structured methodologies such as SSADM concentrate on producing highly visible, meticulously documented system requirement specifications. Strategy-oriented methodologies such as Information Engineering concentrate on

justifying and prioritising information system developments in terms of an organisation's critical success factors and business plans. Software development methodologies focus on the design and construction of the software components of systems. Matching the development approach and methodology to the appropriate problem situation is an important issue for system developers.

The question arises as to which types of project or problem situation is prototyping applicable to. Two of the main factors in making this decision are system complexity and user participation. Simple systems are categorised by a relatively small number of components, stability of function and well-defined behaviour and structure. Complex systems contain large numbers of components, are dynamic and their behaviour and structure are difficult to analyse. User participation can be categorised as either unitary, pluralistic or coercive. Unitary users agree on ends and means and act in accordance with agreed objectives. Pluralistic user groups do not necessarily agree upon ends and means. Cohersive user groups definitely do not agree on ends and means and may not be willing to compromise. The particular strengths of prototyping would appear to point towards its main usage in complex-unitary, simple-pluralistic and complex-pluralistic problem contexts. For further discussion of this and other methodology categorisation and selection approaches refer to Avison and Fitzgerald [13], Olle [14], Flood and Jackson [15], Howard [16] and McGinnes [17].

MANAGING THE PROTOTYPING PROCESS

System developers are continually presented with new methods, techniques and tools which promise to improve their productivity and increase the effectiveness of the systems developed. Unless the usage and application of these new ideas is well-managed, however, what may result is chaos and substantially worse maintenance problems in the future. Prototyping is no exception to this rule. Evolutionary prototyping in particular can only take place in a strictly managed project environment. The flexibility in requirements determination and system implementation provided by prototyping presents project managers with the very difficult challenge of how to plan, organise and control

a dynamic development environment. It is much easier to work with a detailed, up-front requirements specification and to adhere to specific lifecycle stages. The justification for the increased management effort is that the systems produced will be more relevant to organisational requirements, will require less adaptive maintenance and will have committed users who feel a greater sense of ownership of their systems.

The methodology used is a key issue in the management of a project. It identifies tasks, resources required and milestones in a project. A good methodology is capable of being applied successfully to higher level and more complex problems as the learning curve develops. An approach that can be extended and generalised has greater power than one that is restricted to particular domains. Once a methodology is established tools to support and manage the process can be selected or developed. If prototyping is to become a major force in system development a methodology is required which has the support of managers, users, project sponsors and developers. PROCONCEPT is proposed as such a methodology.

RATIONALE BEHIND THE PROCONCEPT METHODOLOGY

Within the engineering profession, prototypes are used to demonstrate the feasibility of design concepts and to modify these concepts where necessary and beneficial so that the initial concept or a variant of it can be realised in a production model. Iggulden [3] points out that the general notion of prototypes is closely related to the concept of modelling and that models have three different uses:-

1. Simulation, where parts of the real world are abstracted and modelled

2. Realisation (or feasibility) with the model showing that the underlying ideas are consistent

3. Idealisation, a standard or ideal against which other forms of the basic idea can be measured

A concept prototype fits in well with the idea of a simulation model. Realisation equates to an engineering prototype. The idea of a

standard or ideal model against which other concepts or working prototypes can be compared is applicable to concept, engineering and production prototypes.

Models can be text-based, diagrammatic, physical, mathematical or philosophical. Prototyping should make use of whichever type of model is most appropriate in the circumstances. Most developers usually make the assumption that a prototype is a partially complete working physical model of an internal or external component of a system. Mason [10] questions whether a prototype actually needs physical embodiment in the early stages of development. To concentrate entirely on physical models denies the prototyper access to a wide variety of other models which could be just as effective in the development process.

Heisler *et alia* [18] point out that user requirements documents concentrate on the planned activities of a system such as data input, core processing and reporting. Unplanned or failure mode activities are added later by the system developers to cater for non-functional requirements. Whilst the requirements document focuses on the functional requirements of the system, often the real complexity of system development relates to the non-functional activities. This observation supports a multi-phase prototyping approach. The concept prototype can concentrate on the central logical requirements of the system. At the engineering prototype stage non-functional system components can be identified and prototyped.

A phased prototyping approach has been hinted at by several authors in the prototyping literature (e.g Carpenter [19], Carey and Mason [20], Tillman [11]). PROCONCEPT builds on this idea and identifies the tasks to be performed in each phase.

THE PROCONCEPT METHODOLOGY

PROCONCEPT is a three phase, evolutionary methodology which can be applied to system development projects within an established prototyping infrastructure. It can be applied to either new developments or to maintenance projects. The methodology provides a logical framework for the developers, telling them what

to do not how to do it. In this way it provides the opportunity for developers to be creative and innovative within a manageable framework.

Phase 1: Concept Prototyping

The purpose of the concept prototyping phase is to eliminate conceptual misunderstandings in the proposed system. Concept prototyping concentrates on the fundamental purpose of the system not on technical detail. The concept prototype is gradually expanded and refined as the project participants increase their comprehension of the proposed system. In building the concept prototype the user interface is emphasised in order to maximise user feedback. Internal components will in the main be prototyped in later phases once the overall system concept desirability and feasibility has been established. The main purposes of the concept prototype can be summarised as follows:-

1. To establish the initial scenario
2. To determine what the user wants and doesn't want from the system
3. To determine whether or not the concept is feasible
4. To identify how the proposed system will integrate with existing ones
5. To assess user reaction to initial prototypes
6. To ascertain initial estimates of effort, cost, resources and development time

Because it is the high-level ethos of the system that is under investigation in concept prototyping it is essential that user involvement is at a sufficiently high level that participants have a proper overview of requirements and the authority to make decisions. The tasks within the concept prototyping phase and the deliverables are described in table 1.

PHASE 1 CONCEPT PROTOTYPING

Tasks

1. Establish preliminary concept or idea
2. Set up concept prototyping team
3. Develop the concept or expand the idea
4. Agree the overall strategy, tools and techniques for this phase
5. Establish the components for prototyping - which aspects of the proposed system will be prototyped in phase 1 and how
6. Initiate developer/user relationships for this phase
7. Agree timetable for this phase
8. Build initial concept prototype components
9. Exercise prototypes
10. Revise and review prototypes (repeat as necessary)
11. Link prototype components as required
12. Exercise overall concept prototype
13. Revise and review overall concept prototype (repeat as necessary)
14. Document concept prototype model and requirements
15. Estimate and document the resources required for the engineering prototype phase
16. Review the concept prototype and further development requirements with the project sponsors
17. Obtain authorisation to proceed to the next phase
18. Hold team review of concept prototyping phase

Deliverables

1. Brief description of system concept and objectives
2. Concept prototype model and requirements
3. Estimate of resources required for engineering prototyping phase

Table 1 Tasks and Deliverables Within Concept Prototyping Phase

Phase 2: Engineering Prototyping

The purpose of the engineering prototyping phase is to determine the feasibility of construction of all system components and to ensure that they conform to the requirements of users. Early engineering prototypes should emphasise system

interfaces. Later prototype versions can deal with internal components once the interface requirements have been established. Engineering prototyping ensures that:-

1. Developers know *how* user requirements will be met
2. Prototypes meet specifications
3. System components are durable for testing
4. Engineering implications for system components are fully understood
5. Key architectural decisions are taken for the proposed system
6. In-house testing can be performed
7. End-users have a fuller picture of the system

The tasks and deliverables for this phase are described in table 2.

Phase 3: Production Prototyping

The purpose of the production prototyping phase is to produce a complete working model of requirements which can be reproduced for as many installations as required. The production prototype becomes the initial version of the system. The tasks and deliverables for this phase are described in table 3. The main purposes of this phase are:-

1. To make any changes to the engineering prototype components which are necessary for production purposes
2. To ensure complete implementation and integration of all system components
3. Final stress testing of the system
4. To produce required system documentation
5. Controlled release of the system in the user environment

PHASE 2 ENGINEERING PROTOTYPING

Tasks

1. Set up engineering prototyping team
2. Review documentation, models and requirements from phase 1
3. Agree the overall strategy, tools and techniques for this phase
4. Establish the non-functional requirements of the system eg. audit trails, error processing, maintenance requirements etc.
5. Identify all the system components for engineering prototyping
6. Initiate developer/user relationships for this phase
7. Agree timetable for this phase
8. Build engineering prototype components
9. Exercise, revise and review prototypes (repeat as necessary)
10. Link prototype components as required
11. Exercise overall engineering prototype
12. Revise and review overall engineering prototype (repeat as necessary)
13. Document the engineering prototype
14 Identify and list those features of the engineering prototype requiring special attention when the production prototype is being produced
15. Estimate and document the resources required for the production prototyping phase
16. Review the engineering prototype and the production requirements with the project sponsors
17. Obtain authorisation to proceed to the next phase
18. Hold team review of engineering prototyping phase

Deliverables

1. Description of required system structure and functional and non-functional requirements
2. Engineering prototype model
3. List of features requiring special attention in the production prototyping phase
4. Estimate of resources required for production prototyping phase

Table 2 Tasks and Deliverables Within Engineering Prototyping Phase

PHASE 3 PRODUCTION PROTOTYPING	
Tasks	
1.	Set up production prototyping team
2.	Review documentation, models and requirements from phase 2
3.	Agree the overall strategy, tools and techniques for this phase
4.	Initiate developer/user relationships for this phase
5.	Agree timetable for this phase
6.	Build production prototype components
7.	Exercise, stress test and revise components (repeat as necessary)
8.	Link prototype components as required
9.	Exercise, stress test and revise overall production prototype (repeat as necessary)
10.	Fully document production prototype
11.	Release production prototype
12.	Hold review of production prototyping phase
Deliverables	
1.	Final working prototype (version 0) of required system
2.	Full system documentation (operational/user/training)

Table 3 Tasks and Deliverables Within Production Prototyping Phase

REACTION TO THE METHODOLOGY FROM DEVELOPERS

PROCONCEPT has generally been welcomed by those developers and project managers who have come into contact with it. Developers who do not currently utilise prototyping have requested more details about techniques used to prototype and how the process is actually conducted. Developers who currently use prototyping have requested more information about potential tools to support the various tasks within the methodology and about prototyping strategy formulation. Many developers have been intrigued by the proposal that a prototype does not necessarily need to be a physical model. More details have been requested about the following aspects of the methodology.:-

* Composition of teams for each phase

* Identification of suitable components for prototyping
* The construction of concept prototypes
* Stress testing of prototypes
* Costing the prototyping process
* Documentation of prototypes
* Configuration management in PROCONCEPT
* Modelling techniques other than physical modelling

Whilst it is intended that PROCONCEPT should remain a methodological framework within which developers can exhibit their own creativity it is apparent that some developers prefer set techniques for the performance of each task. There are arguments for and against the prescriptive approach. These will be considered carefully as further versions of the methodology are developed.

CONCLUSION

The sequential system development process is not appropriate for many large-scale computer-based system projects where requirements are complex and susceptible to change. Developers and users must form a partnership to develop systems that meet corporate and user objectives. A dynamic development approach is required which embraces change and extensive user involvement. Evolutionary prototyping is such an approach but its adoption has been restricted by the lack of a clear, concise, teachable and manageable methodology by which projects can be progressed. PROCONCEPT provides an easily understandable framework which overcomes these objections. The two basic risk elements in system development are addressed. The application risk: will the system satisfy user needs and perform beneficially for them, is dealt with in the concept prototyping phase. The technical risk: can a system be built which meets user needs, is addressed in the engineering prototyping phase. The PROCONCEPT methodology has been welcomed by developers and project managers as it ensures that the implemented system meets user requirements at the time of release. System maintenance is aided by the detailed learning

process that has taken place between users and developers during the three development phases.

REFERENCES

1. Jenkins, A.M., 'Prototyping: A Methodology for the Design and Development of Application Systems' *Discussion Paper No. 227,* Division of Research, Graduate School of Business, Indiana University, April 1983.
2. Gomaa, H., 'Prototypes - Keep Them or Throw Them Away?' *State of the Art Report on Prototyping* Pergamon Infotech Ltd, pp. 41-53, 1986.
3. Iggulden, D., 'Prototyping Developments' *State of the Art Report on Prototyping* Pergamon Infotech Ltd, pp. 56-63, 1986.
4. Rowen, R.B., 'Software Project Management Under Incomplete and Ambiguous Specifications' *IEEE Transactions on Engineering Management*, Vol. 37, No. 1, pp. 10-21, 1990.
5. West, M.G., 'Taxonomy of Prototyping - Tools and Methods for Database, Decision Support and Transaction Systems' *State of the Art Report on Prototyping* Pergamon Infotech Ltd, pp. 105-120, 1986.
6. Appleton, D.S., 'Prototyping' *Proceedings GUIDE 57* Session No. DP-7252, November, 1983
7. Blum, B.I., 'Application Systems Prototyping' *State of the Art Report on Prototyping* Pergamon Infotech Ltd, pp. 5-14, 1986.
8. Crinnion, J., 'The Systems Implications of 4GLs *Journal of Information Technology*, Vol. 4, No. 2, pp. 71-80, 1989.
9. Earl, M.J., 'Putting It in its Place: A Polemic for the Nineties' *Journal of Information Technology*, Vol. 7, pp. 100-108, 1992.
10. Mason, R.E.A., 'Prototyping in Software Engineering - Prototyping the Future' *State of the Art Report on Prototyping* Pergamon Infotech Ltd, pp. 75-88, 1986.
11. Tillman, G., 'Prototyping for the Right Results' *Datamation*, pp. 42-46, April, 1989.
12. Sobol, M.G., and Kagan, A., 'Which Systems Analyst are More Likely to Prototype' *Journal of Information Systems Management,* Vol. 6, No. 3, pp. 36-43, 1989.
13. Avison, D., Fitzgerald, G., *Information Systems Methodologies* Blackwell Scientific, 1991.
14. Olle, T.W., *Information Systems Methodology,* Addison-Wesley, 1991.

15. Flood, R.L., Jackson, M.C., *Creative Problem Solving: Total Systems Intervention*, Wiley, 1991.
16. Howard, A.W., 'Methodology Selection Grids' *Computing Notebook*, Sept., 1992.
17. McGinnes, S., 'A Framework for the Comparative Evaluation of Information Systems Methodologies pp. 43-56, *Proceedings of the Conf on the Theory, Use and Integrative Aspects of IS Methodologies,* Edinburgh, 1993.
18. Heisler, K.G., Tsai, W.T., Ramamoorthy, C.V., 'Integrating the Role of Requirements Specification into the Process of Prototyping', pp. 348-357, *Proceedings of the 22nd Annual Conference on Systems Sciences*, Hawaii, 1989.
19. Carpenter, R.A., 'Designing and Developing Adaptive Information Systems' *Computer Technology Review*, pp. 19-29, Spring/Summer, 1982.
20. Carey, T.T., Mason, R.E.A., 'Information Systems Prototyping: Techniques, Tools and Methodologies' pp. 177-191, *Proceedings of INFOR CONFERENCE,* 1983.