THE USE OF SOFTWARE PROTOTYPES FOR REQUIREMENTS CAPTURE: AN EXPLORATORY STUDY

BY

LIEUTENANT COMMANDER D. G. ELFORD MSc, CEng, MRAeS, AMIEE, RN
(Directorate General Aircraft (Navy))

The article was originally submitted by the author to the Institution of Electrical Engineers, following a call for papers for the first International Conference on Customer-driven Product Quality in Production Design. The article is based on a thesis he had to write whilst undertaking an MSc course in computer science. The experimental element of the work was based on an IT project within the Fleet Air Arm.

ABSTRACT

Many prospective users of planned or proposed computer systems cannot imagine what the system will look like, or how it will support their work once it is implemented. Consequently, they are not able to make much of a contribution at the requirements capture stage. This article outlines the rationale for employing a system prototype, similar to the ones used in participatory and user interface design, to help users imagine a system and start to discuss functionality. It concludes with the results of a small study in which the approach was put into practice.

Introduction

Traditionally, the requirements capture stage of the software development process is performed using a series of meetings and interviews. During these sessions, prospective users are usually asked to describe existing work practices, and to state what they expect from the new system. The software developers generate a specification on the basis of this, which they present to the user organisation for approval. Once this approval is received from the users, the specification is frozen and the development begins in a manner prescribed by software engineering models, such as the 'Waterfall Model'.

It is, however, frequently very difficult for prospective users to imagine how such a system could be used. This has been recognised by those involved in software engineering. Sommerville[1] states that:

> 'The main problem with the process of requirements specification is the inability of the prospective users to imagine the proposed system'.

Often, they have little comprehension of the potential power of the new system, and of the changes to working methods that the effective use of such a system could bring. In short, potential users often do not know exactly what they want at this early stage of a development project.

Furthermore, since software developers and users often have entirely different backgrounds, communication of any requirements can be difficult. Consequently, initial requirements specifications may later be found to be incorrect and incomplete. Inadequate requirements inevitably lead to unsatisfactory and inferior software systems, which have to be modified at later stages of the development, often post-delivery. This costs a great deal of time and money and so it is vital that valid requirements are generated as early as possible in the software process.

The aim of this study was to demonstrate that the requirements capture process, for a software project, can be effectively supported by employing a prototype.

Prototyping

A prototype is defined as:

> An original example of something from which others have been, or will be developed.

Traditionally, there are three main functions that prototyping can perform:

1. The prototype is employed to validate requirements stated by the customer or user. One example is in using a scale model of a proposed architectural development, to allow the non-technical people who will be affected by the planned building to imagine how their environment will be altered.
2. The prototype may be used to validate design. For example, an electronic system is initially assembled as a prototype using available components and a 'bread-board'. Once the prototype is considered satisfactory and the design has been validated, investment is made in the expensive application-specific integrated circuits which will implement the production version of the system.
3. The prototype is used to aid communication between system developers and procurers. Effective communication is vital if the client is to obtain the required system, as the requirements and problems must be stated, and the developer must understand what is to be produced. Failures in this dialogue range from misunderstandings of terminology to requirements' oversights.

Prototyping in Software Engineering

Sommerville[1] maintains that, in software engineering, prototyping is solely intended to fulfil the first function, that of requirements validation. Indeed, Luqi[2] goes further by stating that:

> 'In practice . . . the design of accurate and stable requirements cannot be completed until users gain some experience with the proposed software system.'

Errors and omissions may be detected and rectified prior to implementation, thereby avoiding expensive modification. Validation involves ensuring that the requirements are consistent, realistic, complete and valid. Lantz[3] advocates the use of prototyping throughout the system development process, alongside traditional system development tools, mainly to aid communication between developers and users.

Prototyping in Participatory Design

Participatory design emanated from the Scandinavian countries, where end-user participation in system development was mandated in the early 1970s, with the introduction of 'co-determination' legislation. In general terms, these employment laws required that employees, through their unions, should be able to collaborate with management when any re-organizations or changes in work practice were proposed. Therefore, the proposed introduction of new technology into the workplace had to involve worker-collaboration. However, the popular software development methodologies of the time prescribed a structured approach to systems design, which made little provision for end-user involvement.

Since the legislation came into force, participatory design has been undertaken in many developments of all types. One of the first major projects which attempted to solve the above paradox was the Utopia project in Sweden and Denmark (Bødker et al[4]). System developers and researchers worked with a small group of news-medium employees, in order to enhance worker-skill and improve the typographic quality of newspapers. The prospective users of the new system, the journalists and typographers, were encouraged to conjure up a vision of their future working environment by using mock-ups. (This process has since become known as 'envisionment'; Bødker & Grønbæk[5])

The mock-ups, which included cardboard boxes, slide projectors, drawing boards, newspaper cuttings and sketches on paper, were used to simulate both the tangible artifacts of the new system and the actual work it would carry out. What was different here, was that the users no longer had to rely on creating conceptual models generated from written or verbal descriptions made by the designers. They could now use the mock-ups as a focus for their comments, suggestions and criticisms. They could more easily relate their imagined use of the new system to their current working practices and to their skills and experience. Furthermore, any 'breakdowns' (Winograd & Flores[6]) in the flow of a simulation, would initiate a discussion about the use of the mock-up itself (which had hitherto been transparent to the user). The discussion would often lead to alterations in the simulation and therefore to the emerging design.

Other prototypes which have been employed include storyboards (Andriole[7]), scenarios and simulations. A computerized prototype is a more sophisticated form of the more general mock-up and has two advantages:

First, there is the efficiency, flexibility and functionality that computers can bring to the prototyping process.

Secondly, if the prospective user is able to use a computer-based prototype, then the simulated work with the future computer system is more realistic.

Floyd[8] classifies prototypes as either *horizontal* or *vertical*:

Horizontal Prototype.

In a horizontal prototype the whole of the user interface is implemented so that the screen dialogues may be demonstrated, but the prototype has no further functionality.

Vertical Prototype.

In a vertical prototype, only a part of the new system's user interface is implemented. But that part is backed up by selected functions, so that data may be progressed and realistic work tasks may be performed with the prototype.

Horizontal and vertical prototypes may also be combined, so that the functionality of certain parts of a horizontal prototype may be implemented as a vertical prototype.

Prototypes play a pivotal role in participatory design. They act as a catalyst in the process of validating design and provide a focus for communication between designers and users. They introduce users to the systems in an interactive, hands-on way. This is very different from reading a description of the proposed system, or watching a demonstration[5]. They are, in short, a very effective 'way of enhancing users imagination' (Bødker[9]).

Prototyping for Requirements Capture

Requirements capture and analysis, is the process of establishing the services that the system should provide and the constraints under which it must operate. Traditionally, the requirements capture process is performed using a series of meetings and interviews. At these, the prospective users of the new software system describe the existing working practices and attempt to envisage the way in which a new computer system could integrate and improve the status quo. Often, however, users have little comprehension of the potential power of the new system and of the changes to the work methods that the effective use of such a system would bring. This is particularly true when the user has little knowledge or experience of computers.

Furthermore, the work of most experienced people is based on their 'tacit knowledge', which is never tapped in such interviews. Polanyi[10] defines this as:

> 'Knowledge which is used without any reflection or consciousness.'

The combination of both factors means that requirements elicited in this way tend to be incorrect and incomplete.

Potts[11] proposes that a procurement interface exists between customers (the 'users') and developers; customers must know what they will receive for their investment, and the developers must know exactly what they are being paid to deliver. Even when the users themselves are clear about their requirements, they are often unable to communicate these requirements across the interface to the developers. According to Potts, there are two main reasons for this:

1. The knowledge and skills of the software developer and the client representative are frequently disparate.
2. In order that it is complete and consistent, the information constituting the detailed requirements specification is often complex and extensive.

In order to elicit complete and correct requirements, users need the support of a visualization tool and a vehicle for communication with system developers. Prototypes have been successfully employed in participatory design to make designs tangible enough for users to validate proposals and to improve the communication between users and designers.

Could prototypes, then, be equally successfully employed in the earlier stages of requirements elicitation? Can prototypes:

(*a*) Stimulate users' imagination to generate requirements which go beyond current work practice and explore the potential of new technologies?

(*b*) Focus discussion between developer and user to ensure that communication across the procurement interface is effective?

The following study was undertaken to explore the potential of prototypes for the purpose of eliciting requirements.

THE STUDY

The aim of the study was to demonstrate the principle that, by presenting a prototype to prospective users of a proposed computer system, valid and realistic requirements could be elicited. In order to do this, an existing software project, which was in its early stages, was found. A prototype was then rapidly produced based on the preliminary requirements definition for the selected project. This prototype was presented to the prospective users, who were asked to play the role of future users, using the system in their jobs. In this way, it was hoped that each user (who, of course, had a detailed understanding of the present working-system) would gain an insight into the potential of the proposed computer system and therefore be able to suggest pertinent and valid requirements.

After reviewing a number of impending system development projects, an Executive Information System (EIS), for senior air engineering managers with the Fleet Air Arm (FAA), was selected. The planned system would access information held in a very large database, to be constructed for the support of several other IT projects, within the FAA. It was decided to mock-up a prototype and use it to distil some requirements for the proposed EIS. The prospective users of the system had limited experience with computers, which made the project ideally suited for an attempt to elicit requirements by employing a prototype.

The Users

All the user-subjects were senior Royal Navy engineer officers, in their late thirties or early forties. They had similar educational and work backgrounds, but varying familiarity with, and expectations of, computers. This ranged from simply being aware that others used them in their work, to having reached the stage of feeling that computers were indispensable. When asked by potential developers about their requirements for an EIS, very few could express any pertinent requirements.

The Prototype

The prototype was implemented on a desktop PC, using Microsoft Visual Basic running under Windows 3.1. The first version of the prototype was based on a document used within the current management system of the user organisation. This presented management information in the form of a 6-weekly Management Board Report which contained a great deal of 'business' performance information, most of which was presented graphically. The Management Board Report was, in fact, the current EIS, so it was natural that it was used as the basis of the first prototype.

The Procedure

The study consisted of three elements:

Individual sessions where the users explored the prototype.

A questionnaire.

A 'requirements conference'.

Each individual session was divided into two phases:

The First Phase.

In this phase the user-subject was briefed on the background of the study, the purpose of the session, and then shown how to operate the prototype EIS. The facilitator then started the prototype and asked the subject to use it to answer 8 questions in the context of a scenario (which was similar to the approach used by Andriole[7]).

After some trial and error, most users rapidly learned how to use the simple user interface to access the graphical data of the EIS. One or two of the less computer literate users, understandably, needed a certain degree of prompting before they were competently exploring the system and answering the scenario questions. Each user, together with his actions with the prototype, were recorded on videotape for later analysis.

The scenario was particularly useful in two respects. First, it encouraged the users to explore all of the prototype system and to do so with a certain degree of purpose. Secondly, it helped them, subconsciously, to play the role of a future executive-user who needed to interrogate the system to support particular aspects of his work.

The Second Phase.

On completion of the first phase, which was structured around the scenario, the second phase was initiated by the facilitator. This consisted of a type of brain-storming, where users were encouraged to explore the system, think aloud, and take the exploration in any direction they wished.

After the session, each user was asked to complete a questionnaire in his own time and to return it to the author. The aim of the questionnaire was to gauge the level of computer knowledge and experience of each subject and also to give them the opportunity to append any further requirements they might have thought of on reflection.

After the requirements were collected from individual sessions and the questionnaire, a requirements conference was held at which five out of the original six users were able to attend. The aim of the two-hour conference was to present the results of the individual sessions and to attempt to elicit any further requirements. The prototype had been enhanced to incorporate selected functions requested by users in the individual sessions. Again the whole meeting was recorded onto video tape for later analysis.

Results

Once the video tapes and the completed questionnaires had been analysed, transcripts were written and the requirements were distilled into a table. The purpose of the table was to list the more important requirements and to indicate which users had made them and when. On further analysis, the requirements and comments of the subjects were grouped into a requirements taxonomy.

The basic functionality of the prototype and style of interaction was generally approved, both implicitly and explicitly, during the sessions. This must partly be attributed to the fact that the main source of inspiration for the prototype was the black-and-white Management Board Report. Consequently, the effective use of colour greatly appealed to the users. The requirements discussed in the individual sessions focused on what information should be presented by the EIS, when and how. Most users emphasised the need to relate and compare information and to have the facility to attach questions and comments. System flexibility and adaptability emerged as important requirements, as did integration of communication facilities.

In the requirements conference, the improvements which had been made to the presentation of some of the data, following the initial sessions, were also approved. There was a definite change in attitude to a computer-based information system, and users started to explore interactively potentially wide-ranging changes in the way that information was presented and used in their organisation. Users realised that easy access to information would probably lead to increased demand for information, and pondered on how it could be managed. Also, the potential costs and benefits of accessing historical data were discussed.

Discussions and Conclusions

The prototype acted as a focal point which allowed the subjects to begin to imagine how the new system could be used to assist them in their work. This, supported by the scenario and the prompting of the facilitator, created a work-simulation which was considered vital if the users' tacit knowledge was to be revealed and expressed in terms of valid requirements. It confirmed that, as Ramsey and Atwood[12] suggest:

> '. . . users are expert at doing jobs, not describing them.'

A way to enable tacit knowledge to contribute to the development process, is to stimulate the users' hands-on experience by employing prototypes in work-like settings. So, in using a prototype that incorporated familiar elements from the users' own working experience and by setting a realistic scenario, the users were enticed into applying their intuitive knowledge. In this way, the subjects were able to reflect on their current work whilst imagining a future working situation. Such envisionment was achieved by creating a largely interactive user interface which was supported by simulated functionality. The simulations took the form of familiar pictures of graphical data which could be easily accessed by the user.

The results described above, confirm Grønbæk's observation that prototypes should have a degree of functionality in order to engage users in work-like tasks, thereby stimulating good response and useful participation. Purely horizontal prototypes do not provide the necessary coupling between the user's understanding of their current work and their visions of the future. Such prototypes are merely demonstrations and their function is that of a sales tool.

Vertical prototypes, with their deep functionality in narrow areas, provide the best basis for end-user involvement in system development[13]. Such prototypes, however, require large resources to implement. This is difficult to justify with prototypes for requirements elicitation, since, in contrast to prototypes used for design validation, they will be thrown away after the requirements capture phase.

Smaller prototypes, such as the one used in this study, with simulated functionality, hold the key to user participation requirements capture. This prototype was neither purely horizontal nor vertical. It contained selected parts of the user interface, supported by a degree of (mostly simulated) functionality. It might therefore be described as a diagonal prototype. The use of such a prototype, in the context of a scenario, is a very efficient way to elicit requirements.

An important insight gained in the study is that users need to be given time and space to generate requirements. The individual sessions with the prototype only primed users to think about requirements. A lot of important comments were collected from the questionnaires which users could complete after some reflection.

In the requirements conference, users spent a lot of time discussing the future direction of their organisation, and generated high-level goals and constraints for the system. It was subsequently felt that users should be given the opportunity to explore the prototype in small groups, and discuss requirements, without the presence of the facilitators or a camera. The more time spent with the prototype and in discussion with each other, the users felt much more confident about the benefits of the proposed EIS. It is therefore proposed that requirements elicitation using a prototype needs to take place in several phases, and should employ a range of formal and informal sessions.

References

1. Sommerville, I.: *Software engineering*, 1992, 4th edition, Addison-Wesley, Workingham, UK.
2. Luqi, L.: *Software evaluation through rapid prototyping*, 1989, IEEE Computer, Vol 22, pp 22–56.
3. Lantz, K.E.: *The prototyping methodology*, 1986, Prentice-Hall, Englewood Cliffs, New Jersey, USA.
4. Bødker, S., Ehn, P., Kammersgaard, J., Kyng, M. and Sundblad, Y.: *A utopian experience—on design of powerful computer-based tools for skilled graphic workers*, 1987. In G. Bjeknes, P. Ehn and M. Kyng (Editors): *Computers and Democracy—a Scandinavian challenge*, Aldershot, Avebury, UK.
5. Bødker, S. and Grønbæk, K.: *Design in action from prototyping to demonstration to co-operative prototyping*, 1991. In J. Greenbaum and M. Kyng (Editors): *Design at work: Co-operative design of computer systems*. LEA, Hillsdale, New Jersey, USA.
6. Winograd, T. and Flores, F.: *Understanding computers and cognition: A new foundation for design* , 1986, Ablex, Norwood, New Jersey, USA.
7. Andriole, S.J.: *Storyboard prototyping—a new approach to user requirements analysis*, 1989, QED Information Sciences, Wellesley, Massachusetts, USA.
8. Floyd, C.: *A systematic look at prototyping*, 1984. In R. Buddle, K, Kuhlenkamp, L. Mathiassen and H Zuellighoven (Editors): *Approaches to prototyping*. Spinger, Berlin, Germany.
9. Bødker, S.: *Through the interface: a human activity approach to user interface design*, 1991, LEA Hillsdale, New Jersey, USA.
10. Polanyi, M.: *The tactic dimension*, 1966, Routledge and Keegan Paul, London, UK.
11. Potts, C.: *The other interface: Specifying and visualising computer systems*, 1988. In G.C. Van der Verr, T.R.G. Green, J.M. Hoc and D.M. Murray (Editors): *Working with computers: Theory versus outcome*, Academic Press, London, UK.
12. Ramsey, H.R. and Attwood, M.E.: *Human factors in computer systems: A review of the literature*, 1979, Science Applications Inc, Boulder, Colorado, USA.
13. Grønbæk, K.: *Office: Technology and People*, 1989, vol 2, pp 105–125.