Chapter 5 Specific Techniques 3: Interactive Approaches

Objectives

- To address one cause of interaction failure:
  - *that designers*[1] *do not not fully understood the work of users (Greenbaum and Kyng, 1991)*
- To introduce techniques which encourage designers to interact with users
- To discuss the notion of designers as apprentices to users
- To introduce future workshops and metaphorical design
- To distinguish between prototyping and cooperative prototyping
- To introduce Cooperative Evaluation as a requirements engineering technique

5.1 Introduction

The aim of this chapter is to introduce techniques which will assist the requirements engineer in understanding the work of users. The techniques introduced here have a common theme of 'interaction', that is they cause the requirements engineer (RE) to interact with the users. This approach is in contrast to more traditional 'requirements elicitation' techniques where the RE has a predefined model of what he or she is looking for. For example, using structured analysis techniques the RE studies the work of the user in order to find processes, data, inputs and outputs; using an object oriented approach the RE is looking for objects, services and messages; or using a task analysis approach the RE is looking for user tasks, task hierarchies and actions on objects. Using other techniques such as interviewing or issuing questionnaires also relies on the RE having some pre-defined notion of what he or she wants to find.

The problem with these traditional approaches is that the RE may only see the work of the user in terms of some predefined model and fails to see much of what the user is actually doing. The work of ethnographers has raised the awareness of the requirements engineering community of the shortcomings of existing models of the work of users.

Ethnography is a method derived from anthropology and is based on observing the behaviour of groups. Professional ethnographers are employed to observe users over a long period of time and to make detailed observations about work practices. Analysis of audio-visual recordings and results from field studies: " reveal the delicate and complex web of interactional practices through which information is communicated and tasks accomplished.....even apparently individual tasks like reading, writing or typing into a computer are embedded in interactions with others and are designed in relation to another's activities" (Luff et. al, 1993).

[1] in this chapter the terms designer and requirements engineer are used interchangeably, both refer to the person who defines what the system will do.

Their studies suggest that the traditional understanding of tasks as activities carried out by individuals is flawed and that all work activities involve some level of social interaction. By presenting a different model with which to view the work of the users, the ethnographers have pointed to weaknesses in our present models.[2]

The failure to understand the importance of social interaction stems from a general weakness of not understanding the intricacies of what people do and how or why they do it. Requirements engineers typically have some predefined notion of what they are looking for and do not develop a sufficiently intimate knowledge and understanding of users to see clearly what is actually happening.

Thus the problem faced by requirements engineers is how to develop relationships with users to a sufficient level of intimacy to understand the intricacies of their work and to discover how it might be improved.

The next section illustrates some of the problems that arise when the designers of the system fail to properly understand the work of users.

The remainder of the chapter is given over to the description of a number of techniques which were designed to support interaction between the designers and the users. The techniques covered are focus groups, the designer as apprentice, future workshops, prototyping, cooperative prototyping and cooperative evaluation. A complete step by step guide to cooperative evaluation can be found in appendix C.

All the techniques described in this chapter contribute to twenty eight to thirty three from the 'wish list', that is they support development of:
28. Relevant structures on the users' present work
29. Visions and design proposals
30. Overviews of technological options
31. Concrete experience with the users' present work
32. Concrete experience with the new system
33. Concrete experience with technological options

5.2 An illustrative problem situation

The purpose of this section is to present number of examples which illustrate that designers have not fully understood the work of users. The examples are taken from a variety of studies of the real life situations.

The first example illustrates that designers haven't thought of things which are obvious to users:

[2] Ethnography is not a requirements technique nor is it normally appropriate for a requirements engineer to carry out an ethnographic study, however, the results of ethnographic studies can be used to inform the requirements investigation, see Sommerville et. al., 1993, for a fuller explanation

Bravo, 1993, relates experiences of using a database program:,

“ If you have ever had to enter data into a database and had to manage it, you know that one of the most common things you have to deal with is duplications. You enter a name and then find that person was already on your list and you want to go back and delete the dup. Say you have two “Gloria Williams”. - if you delete “Gloria Williams”, you have no “Gloria Williams”. You have to trick the computer: change one of the “Gloria Williams” to “Gloria Wilhelm” and the delete “Gloria Wilhelm” so that “Gloria Williams will be still on your list. Why isn’t there a simple thing that says: “dup, delete one”? The computer would know there are two; take out one and you have what you need”

This illustrates quite clearly that the designers have not studied how or why users delete records from a manual system. In this case not only have they not fully understood the work of users in the first place but neither have they taken the trouble to evaluate the system once in use.

The second example illustrates that designers haven’t thought about the affect of the system on the users:
Bravo 1993:

“ TWA has a new call distribution system. It has eliminated the 6 seconds that you used to have between calls to finish scribbling your paper work, or take a sip of water, or maybe crack your neck. Clearly, whoever designed the system had no idea what it would feel like the instant you hang up to have to pick the phone up again.”

This illustrates one of the key weaknesses within requirements engineering, that is the failure to realise that change in the computer system will cause change in the users job. In this case there was probably only a minor change in the software but the quality of the users job deteriorated dramatically.

The third example illustrates that designers haven’t thought about the affect of the system on social interaction. The example is taken from an ethnographic study of doctor and patient during consultation sessions. The doctor has a computer on his or her desk and enters patient details into the computer in order to obtain a printed prescription :

Luff et al., 1994 :
“.....They (the doctor) may also need to attend to output messages, such as requests for clarification or correction of input and /or warning beeps. In addition, once they have started along the prompt line, they will have little control over the order in which information is entered.
On occasions, these constraints appear to undermine the doctor’s ability to participate simultaneously in discussions with patients concerning topics that are not directly related to the production of prescriptions.....”

This illustrates that the doctor patient relationship is being affected by the presence of the computer. The patient can no longer chat to the doctor once the formal consultation is

complete because the doctors attention is drawn towards the computer. Even as the doctor becomes more experienced in using the system, the situation does not change:

Luff et al., 1994 :
" However, with one exception, the doctors' increasing familiarity with the system has not as yet led to a marked reduction in the extent to which its use adversely affects social interaction."

Thus not only is the system affecting the way in which the doctor performs his or her job, it is also affecting the relationship between the doctor and the patient.

The remainder of this chapter presents a number of techniques which help to create situations in which the requirements engineer can develop a more intimate knowledge of users. The techniques vary from group sessions in which users participate, to designers acting as apprentices to users.

5.3 Designer as Apprentice

Beyer and Holtzblatt, 1995, describe the fundamental problem as creating the right relationship between the user and the designer which will enable learning. They take the master-apprentice relationship as a model in which the user is the master and the designer is the apprentice. The master teaches the apprentice by doing the work and talking about it while the apprentice learns by watching and by listening.
According to Beyer and Holtzblatt, 1995, the designer taking on the role of the apprentice:
"automatically adopts the humility, inquisitiveness, and attention to detail needed to collect good data. The apprentice role discourages designers from asking questions in the abstract and focuses them on ongoing work."

However, the user-designer relationship differs from the master-apprentice relationship in two important ways:

- The designer must understand and be able to articulate the structure of the work, the constraints on getting the work done, how work is allocated between users and the physical environment in which the work is carried out.
- The designer must think of ways of improving the work and ask users for feedback on proposals for change.

The apprenticeship model provides the designer with an opportunity to explore the work of the user beyond the initial assumptions. It encourages a more intimate relationship in which the designer can develop an empathy with the intricacies of the users job.

Designer as apprentice can :

- support articulation of the product concept
- support the user in reviewing models developed
- support users in analysing their own problems and identifying the need for change
- support the development of a 'shared meaning' of the system being specified

- enable designers to get concrete experience with the users' present work
- support identification of constraints such as cost, time and security
- support identification and specification of quality attributes: usability, reliability, portability, performance, security, maintainability, acceptability and so on depending on the proposed system.
- support identification and specification of requirements for user documentation, requirements for training, requirements for user support

Beyer and Holtzblatt, 1995, describe a number of examples of 'designer as apprentice' in practice.

The next section presents a different approach to designer user interaction which involves the designer/requirements engineer in organising focus groups for users.

5.4 Focus Groups

The purpose of a focus group (Draper, 1991) is to allow a group of users to talk in a free-form discussion with each other, in the presence of the requirements engineer.

Focus groups will give the requirements engineer insights into how users think and what things are important to them. It is difficult to achieve this with other techniques.

When a focus group is used for the purpose of understanding user needs and requirements, then the requirements engineer can act as facilitator. In this case the role of the facilitator is relatively passive.

If a prototype or mock up of a system exists, then the focus group can take place after the target users have had some exposure to the system. The requirements engineer will need an audio recorder, ideally between four and six users, and should prepare a list of topics for discussion. The topics should be introduced one at a time, the main aim being to find out what people think about the topics. The requirements engineer should not take part in the discussion but should facilitate free discussion between the users. A focus group would typically last about 45 minutes.

If the users have been carrying out particular tasks on a prototype or mock up then the topics for discussion could relate to those tasks. For example, topic 1: how did they get started with the first task?; topic 2: were they able to complete the first task, if not what difficulties did they have? Such open questions should trigger the group to discuss problems and share experiences. The requirements engineer can listen to the group at the time and listen to the audio recording after the focus group has finished. Often it is useful to have a transcript of the discussion, though this is very time consuming to do.

Focus groups can:

- support articulation of visions and design proposals
- support articulation of the product concept
- support users in analysing their own problems and identifying the need for change

- support the development of a ‘shared meaning’ of the system being specified

The next section introduces an approach which requires more interaction between designers and users than the focus group.

5.5 Future Workshops, Metaphorical Design and Design Mock-ups

Future workshops were originally developed by Jungk and Mullert (1987) for citizen groups (in Scandinavia) who wanted a say in the decision making processes of public planning authorities. Kensing, 1987, was one of the first to propose it’s use in system development. The technique meant to ‘shed light on a common problematic situation, to generate visions about the future, and to discuss how these visions can be realised.’ (Kensing and Madsen, 1991). Participants should share the same problematic situation, they should share a desire to change the situation according to their visions, and should share a set of means for that change.

Future workshops are run by one or two facilitators and no more than twenty participants. The facilitator’s role is to ensure an equal distribution of speaking time and to ensure that all participants can follow the discussion. The facilitator could be a designer or a requirements engineer.

Kensing and Madsen, 1991, describe a scenario in which future workshops are used together with metaphorical design. Metaphorical design essentially consists of identifying possible metaphors which could be used to help the participants think about possible alternative future situations. The steps followed in the scenario can be summarised as follows:

SET UP:

- The facilitators met with the project initiator (in this scenario the project initiator was the chief librarian)
- A project group was formed consisting of the two facilitators and six workers (in this case the six were three librarians and three clerical staff)
- A deadline was set for delivery of the proposal for future computer usage, the project group to deliver to the project initiator (in this case the deadline was in four months)

PREPARATION:

- The group developed a common background by: (a) the facilitators working as ‘apprentices’ to the workers and (b) the facilitators demonstrating hardware and software by arranging visits to other workplaces. (in the scenario the other workplaces included libraries, a store, a museum and a community centre)
- The facilitators identified possible metaphors (in this case the metaphors for a library were: a warehouse, a store and a meeting place)
- Invitations were sent to a one day Future Workshop (in this case invitations were sent to all employees but not to library heads)
- A suitable location and materials were found (tape, markers, large sheets of paper)

FUTURE WORKSHOP: INTRODUCTION

- Introduction to the technique and plan for the day by the facilitators.

FUTURE WORKSHOP : CRITIQUE PHASE:

- Critique Phase part 1: This was a structured brainstorming session where participants were asked to focus on current problems at work. Speaking time was restricted to 30 seconds and short statements were written on the wall charts. Statements do not need to be justified by the speaker, and other participants are not allowed to criticise the statements of others. (In this scenario typical statements included: we never talk to borrowers; librarians are just attendants; we only provide self-service; the library is like a supermarket...)
- Critique Phase part 2: Facilitators encouraged the participants to group the short statements under four or five headings and then to separate into small groups for discussion. This was then followed by a plenary session in which all groups presented back to the meeting. (The headings in this case were: the library as a warehouse; relation to borrowers; the organisation of the library; the library as a store; the role of DMK)

FUTURE WORKSHOP : FANTASY PHASE

- Fantasy phase : warm-up: Facilitators encouraged participants to draw pictures of the a library of the future and hung these on the wall around the room.
- Fantasy phase : brainstorming: Participants were encouraged to write short statements on the wall chart about the future situation. No criticism of ideas was permitted. A 'utopian outline' was produced by getting participants to ranked the statements in order of those they favoured most. Each participant was given five votes.(Statements were made such as: tear down the walls, arrange small reading groups, access to the library from home)
- Fantasy phase : use of metaphors : Participants were encouraged to think of the library as a warehouse, in what way was it the same and in what way was it different. A view emerged from this discussion and from the 'utopian outline' that although the library had much in common with a warehouse, it should also be like a meeting place.

FUTURE WORKSHOP : IMPLEMENTATION PHASE

- Each group discussed the 'utopian outline' and how it might be realised. The suggestions were co-ordinated in a plenary session and became the basis of a common strategy for library staff.
- Follow-up actions: designers were charged with developing a prototype of part of the proposed system. Users were charged with getting support from other library staff and with developing a set of criteria for evaluating the prototype.

As Kensing and Madsen point out the future workshop provides a good basis for further development of the project, and additional workshops may be held to solve specific problems encountered later in the project.

Future workshops can :

- support articulation of the product concept
- support problem analysis
- support users in analysing their own problems and identifying the need for change
- support the development of a ‘shared meaning’ of the system being specified
- encourage intuition, imagination and common sense among participants
- support communication between people from a diversity of backgrounds
- provide facilitated meetings with predefined agendas and problem solving strategies
- support the development of listening skills among participants
- support the development of visions and design proposals

Examples of the use of Future Workshops can be found in various reports within Greenbaum and Kyng 1991, Schuler and Namioka (1993), Kyng, 1991, Ehn and Kyng, 1991 and Ehn et. al. 1990.

The next section considers the role of prototyping in involving users in design. Traditional approaches to prototyping are contrasted with cooperative prototyping.

5.6 Prototyping

Prototyping is the technique of constructing a partial implementation of a system so that customers, users, or developers can learn more about a problem or a solution to that problem. (Davis 1993)

There are a variety of approaches to prototyping, each providing an opportunity for users to gain hands on experience before the final application is built. The two main approaches are incremental prototyping, where the prototype incrementally becomes the system, and ‘throwaway’ prototyping, where the prototype is a ‘mock-up’ or simulation of some aspect of the system designed to get feedback from users.

Although one of the main reasons for prototyping is to involve users, they are normally involved in evaluating some prototype which already been designed. Designers start from the users’ more or less articulated requirements and develop a prototype which tends to take the perspective of the designers and the software engineers themselves.

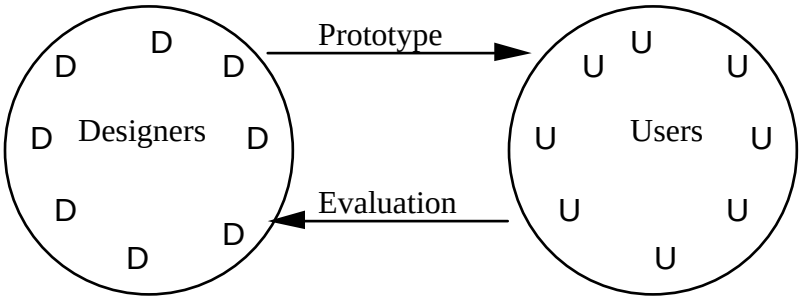


Figure 5.1 Prototyping: The designers and users belong to separate workgroups

Users are not normally involved in the design of the prototype itself. Figure 5.1 illustrates the relationship between users and designers. Traditional prototyping approaches involve the designers in designing the prototype and then getting the users to use it to carry out some specific tasks. The results of the user evaluation will then be used by the designers to improve the design of the prototype or will act as input to the requirements document.

Evaluation can be carried out in a usability laboratory or in the user's normal place of work. In a usability laboratory, video cameras and data logging allow designers or evaluators to view users using the system and to analyse any problems after the user has completed their tasks. In the user's normal place of work the evaluator would need to observe the user in an unobtrusive manner and make a log of user actions and user problems. Users can be asked to say out loud what they are doing and this can be recorded on audio tape for later analysis. Further details of evaluation techniques can be found in Macaulay, 1995b.

Prototyping used as a requirements technique:

- supports an iterative approach to requirements
- gives the users concrete experience with the proposed system
- gives the users concrete experience of technological options
- helps to identify support for user training
- helps to identify support for human computer interface design

The next section introduces a variation of prototyping which assumes much greater user involvement.

5.7 Cooperative Prototyping

The cooperative prototyping approach advocated by Bødker and Grønbaek, 1991, seeks to involve users in the design of the prototype. The aim being to establish a design process where both users and designers are participating actively and creatively. It requires access to flexible computer-based tools for the rapid development and modification of prototypes. Figure 5.2 serves to illustrate the difference between traditional prototyping and cooperative prototyping.

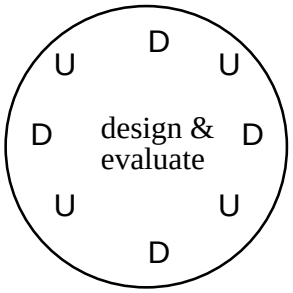


Figure 5.2 Cooperative Prototyping: Designers and users belong to the same workgroup

In cooperative prototyping the designers and users belong to the same workgroup, and both participate in the design of the prototype.

Bødker and Grønbaek, 1991, view cooperative prototyping as a learning process in which the prototype itself has a secondary role. The process of developing the prototype is of primary importance. The users who belong to the designer-user workgroup learn from the process and can then educate other future users while the final computer system is being developed.

For organisations with a large user population, Pape and Thoreson, 1986, suggest a process in which an intermediate prototype is built in one organisational setting, designers then take that prototype to a second organisational setting and develop it further with a second group of users. Figure 5.3 illustrates this process.

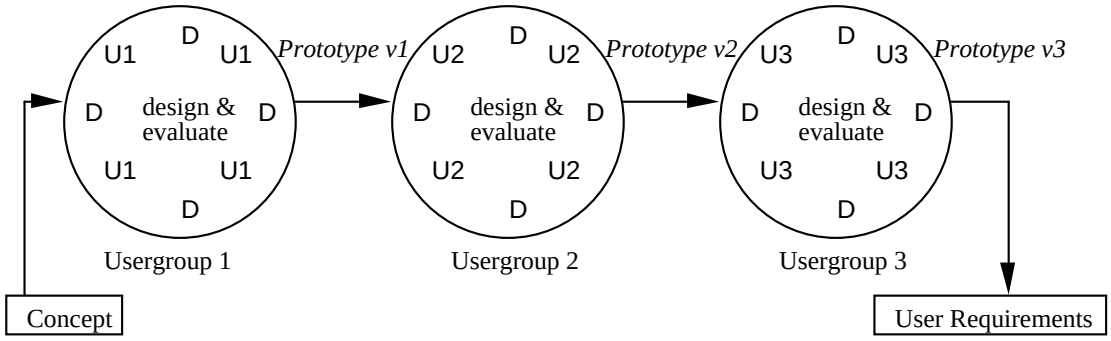


Figure 5.3 The role of Cooperative Prototyping in Requirements

Figure 5.3 shows one possible role for cooperative prototyping in requirements engineering. First there must an understanding of the overall aim of the proposed system, then appropriate people must be found to participate in each usergroup. Each usergroup may belong to different departments, their needs may be different from each other but ultimately they must all use the same computer system.

Designers work with usergroup 1 to develop the first version of the prototype according to their knowledge and understanding. Version 1 of the prototype is evaluated by a second usergroup who then, together with the designers, develop version 2 of the prototype according to their knowledge and understanding. Each usergroup learns something of the needs of the other workgroups and suggests changes according to their own needs. The designers should also learn about the core requirements and variants for each usergroup.

Cooperative Prototyping contributes to requirements engineering by:

- supporting communication between designers and users
- helping to resolve conflicts between different usergroups
- helps to develop design proposals
- gives the users concrete experience of the proposed system
- gives the users concrete experience of the technological options

- helps to identify key workgroups and their objectives
- helps to identify which work roles should be supported and why
- helps to identify requirements for user support, training and human computer interface design.
- contributes to an understanding of the quality attributes: usability and acceptability
- helps to identify core requirements and variants for each usergroup

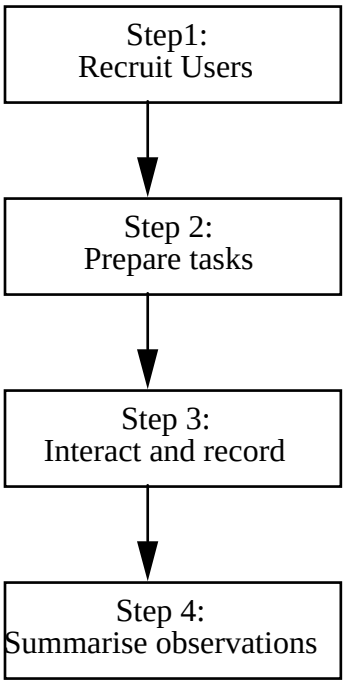
Bødker and Grønbaek, 1991, describe several case studies in the application of cooperative prototyping. The next section considers an approach which involves designers in user evaluations of prototypes.

5.8 Cooperative Evaluation

Cooperative Evaluation (Monk et. al., 1993) is a technique to improve a user interface specification by detecting possible usability problems in an early prototype or partial simulation. Cooperative Evaluation was developed at York University as an inexpensive means of improving the user interface. It involves users in design by having them complete tasks set by designers.

Monk et al. claim:
"Its distinctive features are:
- *it is practical - it can be carried out by designers with very little training and without expensive facilities such as 'usability laboratories';*
- *it is cost effective in that it reveals important usability problems in a relatively short time;*
- *it is for use with early designs and prototypes that may not be complete;*
- *it brings together designer and user in a cooperative context; the user completes work using the prototype and is encouraged to think aloud about the problems encountered."*

Cooperative Evaluation has four main steps:



In the first step the designer has to identify the target user population and to recruit some typical users. The next step is to identify and prepare some tasks which are representative of the tasks users will do and which will enable the users to explore the features of the prototype. The third step involves the designer in preparing the evaluation by deciding what needs to be done before the users arrive, when they arrive (but before starting the tasks), while they are using the system and during the debriefing session. This step involves the designer and users actually carrying out the evaluation. Finally the designer must make a summary of observations, mainly in terms of unexpected behaviour and user comments.

The outcome of the whole process will be the development of concrete experience of users using the prototype, a report of observations made and a list of suggestions for improving the user interface specification.

- Cooperative evaluation contributes to requirements engineering by:
- supporting an interactive approach to user interface specification
- providing a systematic step by step approach
- supporting communication between users and designers
- providing the designer with concrete experience of users using the prototype

- contributing to providing the user with concrete experience of technological options

- providing a basis for identifying usability quality attributes

Appendix C provides a step by step user guide to Cooperative Evaluation. This is reproduced with permission from Monk et al., 1993

5.9 Summary

This chapter presented a number of techniques for encouraging designers to interact with users. A lack of understanding of users work is seen as a key cause of low system usage i.e. interaction failure. The techniques covered were focus groups, the designer as apprentice, future workshops, prototyping, cooperative prototyping and cooperative evaluation.

The next chapter enables the reader to identify the circumstances under which the techniques discussed in chapters three, four and five might be used.