In E. Frøkjær and K. Hornbæk (eds.), *Proceedings of the Second Danish Human-Computer Interaction Research Symposium* (Copenhagen, November 7, 2002), pp 26-27. DIKU Report 2002/19. University of Copenhagen, Copenhagen, Denmark.

Requirements Specifications and Scenarios: Two Design Artefacts in Software Engineering

Morten Hertzum
Centre for Human-Machine Interaction
Risø National Laboratory
DK-4000 Roskilde, Denmark
+45 4677 5145
morten.hertzum@risoe.dk

1. INTRODUCTION

Requirements specifications are an established element of software-engineering projects, and scenarios have gained acceptance in both research and practice as a way of grounding projects in the users' work. However, the research on requirements specifications and scenario-based design includes very few studies of how such design artefacts are actually used by practising software engineers in real-world projects. This study [3, 4] investigates how a requirements specification and a set of scenarios entered into defining how software engineers and users envisioned the future interaction between tasks, users, and the system under development.

The company where the study took place is a large software house, which has developed and marketed a range of systems for use in municipal institutions. The studied project concerns a system to support municipal authorities in the handling of cases concerning child support and alimony (CSA). The CSA project is to completely redevelop the company's existing CSA system, which has been in operation for almost two decades. The CSA project is staffed with 17 people with an average of more than ten years of professional experience, and the project will, according to the project plan, last three years.

The data collected for this study cover the first year of the project and comprise attendance at the two-day start-up seminar, observation of the fortnightly project status meetings, interviews with core project participants, and inspection of project documentation. The meetings and interviews were recorded on tape and transcribed. The documents, which provide evidence of the evolution and intermediate outcomes of the project, include among other things the final as well as several preliminary versions of the requirements specification and the scenarios.

2. REQUIREMENTS SPECIFICATION

The requirements specification consisted of 221 requirements, which were maintained as individual entries and organised by means of a classification scheme. The initial purpose of the requirements specification was to facilitate communication with the user representatives during the requirements-engineering process. After its completion the requirements specification assumed a double role of, on the one hand, contract between users and development organisation and, on the other hand, checklist for the CSA engineers during the development and evaluation of subsequent design artefacts. In these roles, the requirements specification and its classification scheme had a primarily indirect effect on the design process. For example, the scenarios were not generated from the requirements specification. Rather, they were developed on the basis of the CSA engineers' knowledge of the domain and the users' tasks, supplemented by discussions with the user representatives and some reading of CSA legislation. The requirements specification was used most visibly when it was brought in at selected points in the process, for example to validate that design artefacts such as the scenarios met the full range of requirements. However, the requirements specification also affected the design process in another, more fundamental way as a constituent part of the assumptions about the scope of the project.

The requirements specification for the new CSA system inherited a lot of its structure from the existing CSA system. This introduced a potentially undue bias toward preserving existing system facilities and ways of working. The CSA engineers were aware of this risk but explicitly argued that it was more important that the requirements classification depicted the world in a way recognisable to the user representatives. While this is convenient, it also illustrates how the requirements classification indirectly constrained the requirements-engineering process to requirements that could be conceived of within the framework of the existing system [see also 1]. This is apparently at odds with the activities undertaken to facilitate the user representatives in an open-ended search for the optimal balance between tradition and transcendence (e.g., a vision workshop conducted as part of one of the meetings). The CSA engineers were, however, faced with two contradictory concerns. On the one hand, they needed to conduct the requirements-engineering process in a way that honoured expectations of adequate user involvement. On the other hand, they needed to maintain some level of control over the direction, scope, and outcome of the requirements-engineering process, which concluded in a specification of what the customers had requested and the developers agreed to deliver – a contract. The requirements classification played a discrete but important role in the CSA engineers' handling of these two concerns in that it enabled the CSA engineers to act in accord with expectations of adequate user involvement while at the same time constraining the process. On several occasions, the CSA engineers explicitly asked

the user representatives for new ideas and visions regarding the system but, at the same time, the meetings with the user representatives evolved around a walkthrough of the classified requirements, one category at a time. Under these circumstances, the user representatives had few ideas for new facilities that would enhance the system.

The tension between open-ended user involvement and the contractual aspect of requirements specification was rooted deeply in the CSA engineers' perception of their work, and they considered disregard of this tension tantamount to being unprofessional. This was, for example, a problem in their relations with a usability specialist who considered it her role to systematically "adopt the users' perspective". To the CSA engineers handling these conflicting interests was normal, natural practice [2] to the extent that they probably remained largely unaware of how effective the requirements classification was as a means of controlling the scope of their project.

3. SCENARIOS

The scenarios were schematised descriptions of the courses of activities that constitute CSA work. The grounding in the flow of CSA work means that the scenarios are rich in the information needed in the day-to-day management of CSA cases, such as how activities are sequenced, what triggers them, and when they trigger other activities. This means that the scenarios make the users' work recognisable to the CSA engineers as a complex but organised human activity.

Each scenario consists of a chronological progression of activities. Typically, CSA work progresses continually for only brief intervals of time; then further progress must, for example, await that the person entitled to receive CSA supplies additional information. Consequently, most of the steps in the scenarios are triggered by events. These events define the information that must be provided before further progress can be made or they lead to the execution of subtasks that are only relevant when certain conditions occur. Consequently, the scenarios preserve the real-world ordering of the activities involved in performing a task and also delineate the events or circumstances that affect whether and when various activities are performed. The CSA engineers perceived the scenarios as quite coherent descriptions of CSA tasks and considered this a valuable and distinguishing feature of the scenarios.

When the scenarios were discontinued to free key CSA engineers for other project activities several of the CSA engineers were concerned that the discontinuation of the scenarios would deprive them of valuable information about the various aspects of CSA work. This concern was partly an appreciation of the scenarios and partly instigated by the common impression among the CSA engineers that the other design artefacts did not provide them with an equally good tool for understanding CSA work. What the CSA engineers lost with the scenarios was a design artefact that aimed at describing the users' work as tasks consisting of a structured sequence of interrelated activities. Contrary to the scenarios, the requirements specification can best be characterised as an extensive list enumerating large amounts of separate details. The requirements specification provides no information about how the 221 requirements impact on each other. It is, for example, left entirely to the reader of the requirements specification to determine whether it contains conflicting requirements.

The scenarios were developed as a tool for the stakeholders internal to the CSA project. The descriptive nature of the scenarios made them accessible to all CSA engineers and meant that the scenarios were not biased toward, or owned by, a subgroup of CSA engineers responsible for a specific part of the project. Further, all CSA engineers considered it natural to relate their work to the users' tasks, which were the common referent of the scenarios. This can be illustrated by some of the uses to which the scenarios were put. The scenarios generated a number of the events and elementary processes, which made up the business model, and they were a defining input in the development of the dialogue flow of the user-interface prototype. In addition, the CSA engineers preferred the scenarios as their base representation in a joined effort to establish the status of their project after six months had elapsed.

Johnson-Laird and Wason [5] have vividly illustrated people's superiority in dealing with concrete descriptions of real-world affairs, as opposed to abstract descriptions. Whereas abstract descriptions tend to be experienced as logical puzzles, concrete descriptions of real-world affairs seem to tie in with people's general abilities to deal with their world and to be experienced as much more straightforward. Thus, the coherence and concrete, real-world feel of the scenarios may be distinct advantages, which made the CSA engineers better able to grasp CSA work and reason about the suitability of different design ideas.

4. CONCLUSION

It is inherently difficult for people to transcend their current way of perceiving things and envision how tasks, users, and technology should interact in constituting the future use situation. Design artefacts, such as requirements specifications and scenarios, may affect this complex process in very different ways and, thus, play different roles in software-engineering projects.

5. ACKNOWLEDGEMENTS

This work has been supported by the Danish National Research Foundation through its funding of the Centre for Human-Machine Interaction. Special thanks are due to the members of the CSA project group who have put up with my presence in spite of their busy schedule.

6. REFERENCES

[1] Bowker, G.C. & Star, S.L. (1999). *Sorting Things Out: Classification and Its Consequences*. Cambridge, MA: MIT Press.

[2] Garfinkel, H. (1967). "Good" organizational reasons for "bad" clinic records. In H. Garfinkel, *Studies in*

Ethnomethodology, pp. 186-207. Englewood Cliffs, NJ: Prentice Hall.

[3] Hertzum, M. (2002). Making use of scenarios: A field study of conceptual design. Submitted for publication.

[4] Hertzum, M. (2002). Small-scale classification schemes: A field study of requirements engineering. Submitted for publication.

[5] Johnson-Laird, P.N. & Wason, P.C. (1977). A theoretical analysis of insight into a reasoning task. In P.N. Johnson-Laird & P.C. Wason (eds.), *Thinking: Readings in Cognitive Science*, pp. 143-157. Cambridge, UK: Cambridge University Press.