Scandinavian Journal of Information Systems

Volume 17 | Issue 1 Article 7

2005

Filtering, Negotiating and Shifting in the Understanding of Information System Requirements

Päivi Ovaska
South Carelia Polytechnic, Finland, Paivi.Ovaska@scp.fi

Matti Rossi
Helsinki School of Economics, Finland, Matti.Rossi@hkkk.fi

Kari Smolander
South Carelia Polytechnic, kari.smolander@scp.fi

Filtering, Negotiating and Shifting in the Understanding of Information System Requirements

Päivi Ovaska
South Carelia Polytechnic, Finland
Paivi.Ovaska@scp.fi

Matti Rossi
Helsinki School of Economics, Finland
Matti.Rossi@hkkk.fi

Kari Smolander
South Carelia Polytechnic, Finland
Kari.Smolander@scp.fi

Abstract. Traditional requirement engineering approaches pay little attention to how the requirements are interpreted and shared by different parties in an organization. Our study extends the previous research of social and organizational requirement elicitation by suggesting that requirement shaping during a project can be described as a process where attitudes and expectations are filtered, shifted and negotiated repeatedly. We studied a large e-commerce platform development project by applying grounded theory and observed that preconceptions, attitudes and expectations about systems development among project participants *filtered* the understanding of software requirements, *negotiating* between project participants resolved the issues caused by filtering and *shifts* in these attitudes and expectations facilitated changes in the understanding of requirements. In spite of this observed filtering, shifting, and negotiation, the developed system exceeded the customer's needs and expectations even though it was delivered late. We approached the subject with technology frames as an a priori construct and were able to provide a new interpretation of how technology is collectively interpreted in organizations.

Key words: Information systems development, Requirement engineering, Case study

1 Introduction

> The hardest single part of building software is deciding precisely what to build. No other part of the conceptual work is as difficult as establishing the requirements. No other part of the work so cripples the resulting system if done wrong. No other part is so difficult to rectify later. (Brooks 1987)

Studies on large software projects show that they fail at an unacceptably high rate (Lyytinen and Robey 1999). Even if a software project is completed in time, it produces software which is of heterogeneous quality and often exceeds its budget (Lyytinen and Hirscheim 1987; Keil et al. 1995). Many of the problems encountered during systems development are attributable to shortcomings in the software product's requirements. The problems reported in earlier research on requirement engineering typically involve the following issues: insufficient user involvement, ambiguous, changing and incomplete requirements (Pohl 1994; Jarke et al. 1999; Kotonya and Sommerville 1989).

Intensive research on software tools, modelling methods and processes for performing requirement elicitation has not yet delivered tools or techniques that could guarantee foolproof success in software projects. Traditional requirement engineering (RE) approaches offer poor understanding of how to specify and manage requirements for large evolving systems, how the requirements are understood and what kind of problems exist in the commercial practice (Bubenko 1995). Recent research directed into the social and organizational processes in requirement elicitation sees it as an emergent political process (Bergman et al. 2002) constructed through conflicting interests and agendas, resource constraints and political influences or as a socio-cognitive problem solving process (Orlikowski and Gash 1994; Davidson 2002). This socio-cognitive problem solving process is characterized by differences and repeated shifts in assumptions and expectations of technology, which disrupt the project participants' understanding of requirements.

We studied, using a grounded theory approach and technology frames as an a priori construct, a large-scale e-commerce platform project with internal customers within a large telecom operator. The initial goal of this study was to investigate the problems and deviations that caused delays for the fulfilment of the project. Already in the beginning of the study we observed that the requirements did not stabilize during the project but instead kept changing, causing problems and delays. This led us to further study how the software requirements were shaped and interpreted during the project.

The studied project highlights problems in current approaches to requirement elicitation and systems development in general, which still largely assume that projects proceed with distinct phases and more or less in a waterfall fashion from a vague understanding of the idea of the system into a concrete system, which satisfies the originally found requirements. Instead, we observed that the requirement understanding was filtered by project participants in the beginning of the project, *negotiating* between project participants resolved the issues caused by filtering and *shifts* in these attitudes and expectations facilitated changes in the understanding of requirements. This process of filtering, negotiating and shifting can be described as an ad-hoc and iterative process where the software requirements unfold during social interaction, communication and negotiation between parties.

The rest of this paper is organized as follows. In the next section (section 2) we review requirement engineering approaches in systems development literature. In section 3, we outline the research method and process including a description of the organization studied. In section 4, we present the findings of the study as a summary and as a detailed project narrative. In section 5, we discuss these results and their implications for research and practice along with topics for further studies. Finally, we provide conclusions.

2 Related Studies

In this section we look at different approaches to requirements elicitation. We classify the studies into traditional requirements engineering and social and organizational approaches.

2.1 The Requirements Elicitation Problem

Requirements engineering as a field originates in software engineering. It was born of the early observation that regardless of how good the specification techniques for software are, they do not help if the developers do not know which problem to solve. In this view, requirement elicitation deals with *detecting* and *representing* requirements (Pohl 1994). In software engineering, the requirement engineering process is traditionally seen as the systematic techniques of requirement elicitation, documentation and management. This process is described by Pohl's three dimensions of requirements engineering (Pohl 1994). His idealized model assumes that requirement specifications are developed through a process that leads from vague ideas, presented in textual languages and without consideration to agreed viewpoints, into a desired end state where there is a common agreement on a set of relatively formalized requirements that serve as a blueprint for information systems design and implementation (Pohl 1994). This *systematic process* is considered to ensure that system requirements are complete and consistent (Kotonya and Sommerville 1998).

In the information systems (IS) field, requirements are seen as a more *socially constructed phenomenon*. The classical view of Davis (1982) already saw that the effective inclusion of end-users and other stakeholders was vital to this process. The modern view assumes that requirements are defined in a political process (Bergmann et al. 2002). In the following we will briefly look at how each of these approaches has dealt with requirement elicitation and define our own approach to it. We will first look at traditional requirement engineering and then at the social and organizational approaches.

2.2 Traditional Requirement Engineering Approaches

The first major stream of requirement elicitation has been the development of more or less formal specification languages for specifying requirements. Several formal languages have been proposed (e.g., Z, VDM, etc.), but these have been difficult to grasp for the users and managers (Balzer et al. 1978; Balzer 1985; Glass 2004). More informal techniques have been developed to involve the end users (e.g., Joint Application Design), as the process has been seen as a set of successive transformations from informal to formal (Fraser et al. 1991). These techniques have evolved into requirement engineering methods, which provide tools and templates for performing the requirement engineering activities in a certain order (Kotonya and Sommerville 1998; SEI 2002). Furthermore, the process approaches have developed tools for requirement traceability (Ramesh and Dhar 1992) and argumentation during the process (Conklin and Begemann 1988). There are plenty of small scale examples of the success of all of these approaches, but they have been difficult to integrate into a method supporting the whole life-cycle of systems development, and practitioners have mostly ignored them.

2.3 Social and Organizational Approaches

Recently a new approach, which can be called the social and organizational approach, has attracted attention especially in the IS field. It sees requirements emerging from a social and organizational process among the various stakeholders of the RE process. Within this approach, researchers have empirically studied the social, cognitive and political processes in systems development that have an influence on the interpretation of requirements (Dougherty 1992; El Sawy and Pauchant 1992; Orlikowski and Gash 1994; Davidson 2002; Bergman et al. 2002).

Some studies emphasize the *political nature* of requirement elicitation arguing that requirement elicitation is a political process that includes selecting whose goals are addressed and whose are not (Markus 1983; Keen 1981; Boeghm and Ross 1989; Bergman et al. 2002). Bergman et al. (2002) see large-scale requirement engineering as iterations between political and functional ecologies, and the dynamics between these ecologies define the success or failure of the project. They base their approach on Simon's theory of bounded rationality (Simon 1982; Lindblom 1979) and on a behavioural theory of a decision-making model called the Garbage Can Model (March and Olsen 1976; March and Heath 1994). Simon's theory suggests that we can never find an optimal solution for organizational problems, but at most a satisfying one. The Garbage Can Model proposes that organizational decision-making forms protracted processes of iteration in which problems search for solutions, solutions search for problems, and decision-makers search for decisions to be made (Bergman et al. 2002).

Other studies emphasize the *social nature* of requirement elicitation by suggesting that the requirements for systems development do not exist a priori, but they are socially constructed through interactions among systems development participants (Curtis et al. 1988; Dagwell and Weber 1983; Newman and Nobel 1990; Waltz et al. 1993; Davidson 2002). In these interactions, the style of interaction and language use influences which requirements are identified and legitimized (Boland 1979; Boland and Greenberg 1992; Davidson 1999; Mason 1991; Davidson 2002). Information artefacts themselves are socially constructed, and the uses found for such artefacts depend on the meanings that participants assign to them (Grint and Woolgar 1997). Some socio-cognitive researchers (e.g., Orlikowski and Gash 1994; Davidson 2002; Lin and Cornford 2000) have based their findings on the frame of reference concept from the socio-cognitive theory first introduced by Bostrom and Heinen (1977).

The basic assumption behind the frame of reference concept is that people act on the basis of their interpretations of the world, and in doing so enact particular social realities and endow them with a meaning (Berger and Luckmann 1967; Smircich and Stubbart 1985). The frames of reference held by organizational members are implicit guidelines that serve to organize and shape their interpretations of events and organizational phenomena and give these a meaning (Mock and Bartunek 1990; Weick 1979). By shaping individuals' interpretations of organizational phenomena, frames implicitly guide them to make sense of and take action in organizations (Bartunek 1984; Moch and Bartunek 1990). Frames typically operate in the background facilitating and constraining effects (Orlikowski and Gash 1994).

2.4 Technological Frames Approach

Orlikowski and Gash (1994) introduced the concept of technology frame of reference as a subset of members' organizational frames that describes the assumptions, expectations and knowledge that systems development participants use to understand technology in organizations. Orlikowski and Gash define a technology frame as follows:

> a set of assumptions, meanings, knowledge and expectations that people use to understand the nature and role of technology in organizations. This includes not only the nature and role of technology itself, but the specific conditions, applications, and consequences of that technology in a particular context (Orlikowski and Gash 1994).

While technological frames are individually held, they are also social phenomena. When individuals have similar expectations, for example about the technology in business processes or the nature of technology use, it affects their social reality (Orlikowski and Gash 1994). They suggest that differences, or incongruence, in the frames of the key stakeholder groups is a source of disruption in information systems development. They also state that the social nature of technology frames contributes remarkably to the research on collective interpretations of technology. It leads to an increased understanding of the underlying processes of cognitive formation, stagnation and modification, or how shared frames of a group become embedded in technology design and work routines (Orlikowski and Gash 1994, p. 199).

The researchers using technology frames suggest that frames can act as templates for problem solving as well as filters for new information (Davidson 2002). Orlikowski and Gash (1994) found in their study of the adoption of Lotus Notes software three frames that characterized technologists' and users' understanding and use of Notes, i.e., the nature of technology (understanding its features and uses), the technology strategy (assumptions about management motivation for implementing the technology and criteria for judging the success of the IT), and technology-in-use (expectations about priorities and resources, training approaches, ease-of-use, and policies for security and quality). In her study of requirement determination in an R&D project, Davidson (2002) observed the following types of technology frames that influenced the participants' requirement understanding: IT delivery strategies (expectations about how IT functionality should be developed and provided to an organization), IT capabilities and design (knowledge and expectations about relational databases, client-server architectures etc.), the business value of IT (assumptions about how IT can be used to influence and improve an organization's operations) and IT-enabled work practices (expectations about how an IT application will fit into work practices).

We saw the importance of these technology frames as an analytic lens in the interpretation of the requirement shaping and understanding. In the next section (section 3) we will describe our research process and how we used this lens for the interpretation of the findings.

3 Research Method and Process

3.1 Studied Organization and Project

This study was carried out in a software development department of an international ICT company. The software development department was an internal partner for the company's newly formed communication services unit. Marketing of the company's new ICT business ideas and services was the responsibility of this unit. The unit was divided into subunits, which specialized in different application areas, such as mobile gaming, electronic and mobile commerce. These subunits were small, typically ten to twenty young employees with business education. The development of new applications and services was assigned to an in-house software development unit (Internal Development Unit, later referred as IDU) or outsourced companies. The use of

IDU for development of new services was mandated by the company's top management. IDU had approximately 150 employees that had formerly focused on R&D work in the company. During the past few years it had tried to improve its software skills and processes in order to make its development more effective and also to prove its capability to other business units. All the business units of the company did not agree with IDU's processes and did not trust in its software development capability. Their attitudes towards IDU competencies in software development were quite suspicious, mainly because of IDU's history as an R&D department. Quite often business units preferred outsourcing instead of developing in-house.

The business owner of our case study project was the business unit (later referred to as BU) responsible for marketing new business ideas related to electronic and mobile commerce services. The project developed a mobile commerce service platform. The system was intended to enable organizers or their sponsors to promote their products in all kinds of events, such as ice hockey and football games. The system was composed of two subsystems: the platform in which the services were running (Platform subsystem) and the toolbox, which allowed adding, configuring and simulating these services (Tool subsystem). This toolbox was intended to run in a Windows PC and the service platform in UNIX environment. The architecture of the case study system is represented in Figure 1 based on the project's architecture specification document.

The actual systems development project took place in IDU during the year 2001 and it was planned and organized according to a traditional waterfall model with distinct requirement elicitation, analysis, software design, implementation and testing phases. The following table (table 1) represents the key characteristics of this project.

Initially, the project goal was the following:

> to enhance the previous version of the platform by possibility to establish WAP (Wireless Application Protocol), SMS (Short Message System), web services and Content Administrator features that make service establishment easier (previously made with Oracle procedures) (Project Setup Letter 1.0, Feb 2, 2001).

BU urgently needed an enhanced version of the platform; they especially felt that improved Content Administrator features (Tool subsystem in figure 1) would help them to rapidly create new services for the platform. They thought that these Content Administrator features were mostly user interfaces for the service platform (Platform subsystem in figure 1) and they had already outlined some sketches to illustrate them. Because of the time-to-market pressures, the timing of the project became very important. The BU thought that

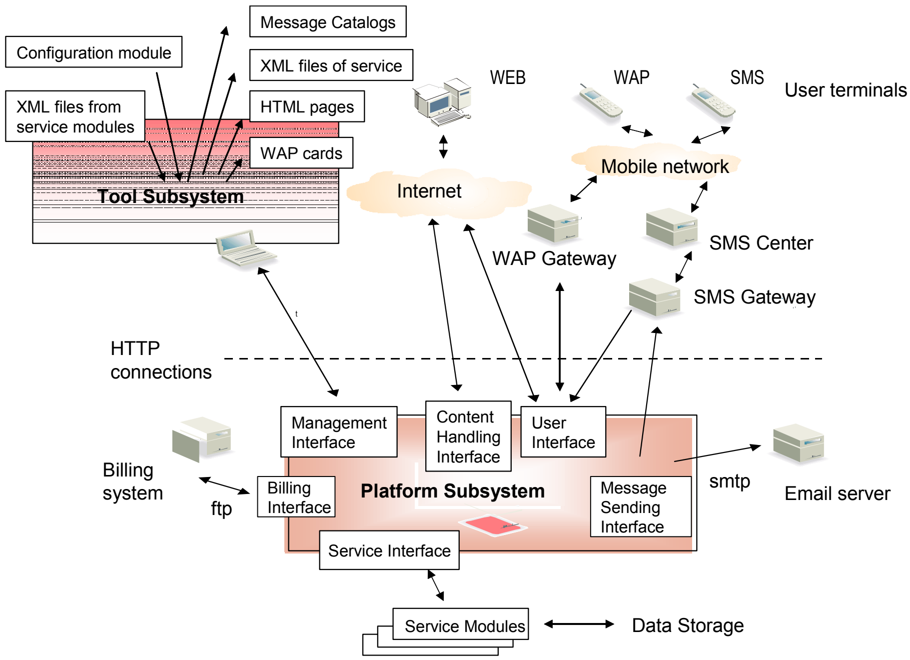

Figure 1. Architecture of the system

four months was a reasonable timeframe to develop just a few user interfaces and some new services for the existing platform. However, during the project, their understanding of the features of the Tool subsystem changed and they realized that it was a much bigger system than what was originally thought.

Characteristic	*Project*
Project execution start and end dates	Start date 23.3.2001 End date 16.10.2001
Software size (Line Of Codes, LOC)	32 000 LOC
Project cost	In total 1217 man days, Execution Phase 780 man days
Projecting	One project, but two subsystems developed quite separately
The goal of the project	Platform enhancement, content administrator
Targeted markets	International

Table 1. Characteristics of the case study project

3.2 Research Question and Method

Initially, our research focused on finding out why there were so many problems and delays in the case study project. We expected coordination problems based on our previous study (Ovaska et al. 2004), but did not find any. Instead, we noticed during our analysis that the requirements did not stabilize, but instead kept changing during the development, causing problems and delays for the project. Furthermore, we observed that it was the understanding of the requirements, which changed during the development. After this notion, we formed our initial research question as: *How were software requirements shaped and interpreted during systems development?*

The nature of our research problem led us to use a qualitative approach. The research methodology for this study followed the grounded theory approach (Glaser and Strauss 1967, Strauss and Corbin 1990, Eisenhardt 1989). The grounded theory method (Glaser and Strauss 1967, Strauss and Corbin 1990) is a "qualitative research method that uses a systematic set of procedures to develop an inductively derived theory about a phenomenon... can be used to study organizations, groups, and individuals" (Strauss and Corbin, 1990). A requirement of the grounded theory is that the researchers demonstrate theoretical sensitivity (Glaser and Strauss 1967). Theoretical sensitivity comes from familiarity with the literature, and from professional or personal experience (Strauss and Corbin, 1990). In our study, the principal researcher worked in the case company for five years as a head of a department, supervised developers in several projects and acted as a steering group member in many projects. In this particular project, she was only a supervisor of the developers, not a steering group member.

The qualitative data analysis was performed in three phases: open coding, axial coding and selective coding (Strauss and Corbin 1990). The notion of a technology frame from socio-cognitive research was used as an a priori construct (Eisenhardt 1989).

3.3 Research Process

The research process proceeded in four broad phases: data collection, formulation of a conceptual model, data analysis and formulation of a project narrative (figure 2).

The study gathered data from extensive documentation available from the project (table 2) using a theoretical sampling strategy. Based on the analysis of

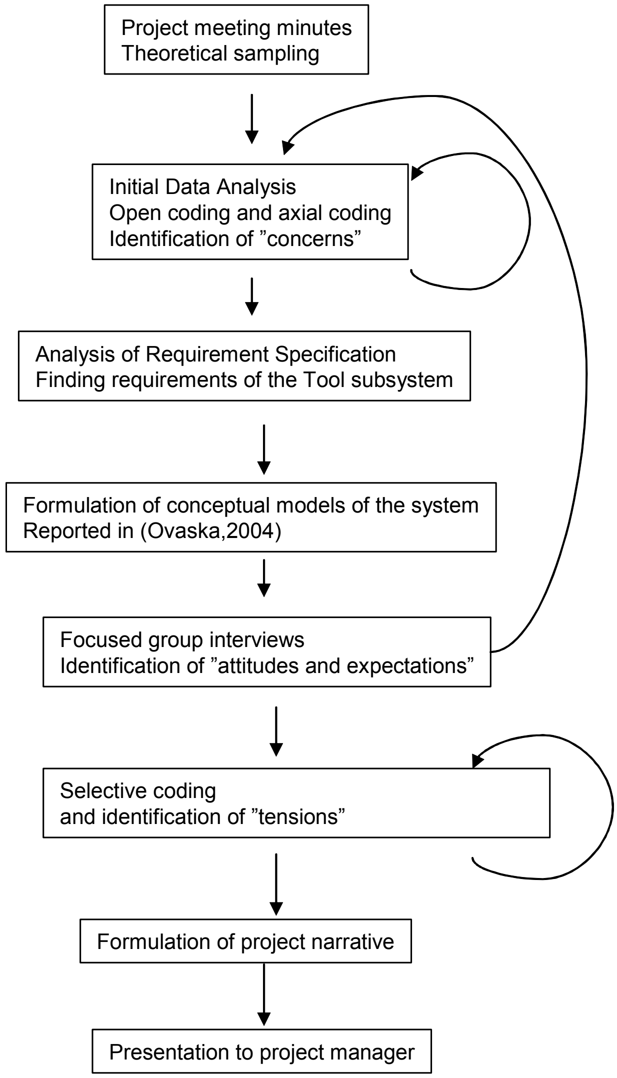

Figure 2. Research process

the documentation we decided to complement the written project material with focused interviews among project participants. Glaser and Strauss (1967) call this a dynamic process of data collection where the sample is extended and focused according to emerging needs as theoretical sampling. We developed conceptual models (Ovaska 2004) of the system to gain a deep understanding of requirement evolution in the project. The models were used as basic information for focused group interviews and they deepened our understanding of changing requirements.

The open coding started with the identification of problems and deviations in the project progress. During development these were issues that were brought to the project meetings for discussion and decision-making. The steering and project group meeting minutes were the main sources for the problem and change identification.

We developed three conceptual models of both subsystems in a former study that measured requirement evolution (Ovaska 2004). Through these models we were able to grasp how the subsystems evolved through different phases of systems development. The content of these conceptual models suggested that the Tool subsystem requirements changed considerably during the process. However, we were able to extract from the requirement specification document only four initial requirements that were related to the Tool subsystem and it was seen as a small user interface part by IDU. This is in sharp contrast with the design documents. This led us to analyse further why the Tool subsystem requirements changed so much while the Platform requirements remained stable.

To answer this question, we performed focused interviews among the project participants to identify the reasons for the changes. Project participants were asked to reflect on the project's history by showing the analysis and implementation models of the system and to describe their understanding of what happened in the project between the requirement analysis and implementation phases of the project. Four of the interviewed project participants were from the development side (IDU) and three of them were from the business side (BU). BU representatives were also asked about the competences and processes of IDU during the time the project was running and how these competences and processes have evolved after that. The interviews were audio taped and fully transcribed to preserve all the details.

The open coding proceeded in parallel, treating each written material and interview as confirmation or further development of results from earlier findings. During this process, the categories developed gradually. First we identified quite concrete concerns of systems development. Most of the concerns were related to the life-cycle model in use, the use of in-house or outsourced development, discussion difficulties between parties, competencies and resources and their allocation to the project.

In further analysis we found more subtle contextual assumptions and expectations about how systems development should be performed. These assumptions and expectations were so strongly visible in the interviews and the meeting minutes that we could interpret them as different technology frames. Thus we decided to use technology frames, which have been used in several previous studies, (e.g., Orlikowski and Gash 1994; Davidson 2002; Lin and Cornford 2000), as an a priori construct to interpret and understand the role of technology and its development in the organization. Orlikowski and Gash (1994) claim that technology frames are expressed symbolically through language, visual images, metaphors and stories. Since individuals employ frames when they produce speech or written materials (Moch and Fields 1985), frames can be evident in oral dialogue as well as in written materials.

The technology frame lenses were applied when examining interview transcripts and written project data. We identified statements or actions of different actors that reflected assumptions or expectations about the systems development and their implications for their work and the firm's operation as a whole. The assumptions and expectations of different groups, such as line managers, project managers, customer representatives, architects and designers, divided the data into groups. Once the data for each group was examined, a cross-group analysis followed (Eisenhardt 1989), consisting of comparing categories generated by each group to determine whether it reflected common themes, such as the technical relationship between business and systems development dominated by managers of the business side. When these themes were identified, the data from each group was re-examined and recoded using the proposed themes. The goal of this re-examination was to determine the set of categories that covered as much of the data as possible. This iterative examination of the data yielded five categories of technology frames, of which one that declared the origin of each technology frame of each group (e.g., technical education) was used as background information during the next phase.

During the next selective coding phase, we identified stereotypical 'tensions' between these attitudes and expectations, which affected how the project participants emphasized technology frames in different phases of the project. These tensions had had important effects on the changes in the project course and on the way the participants saw the requirements. The coding was done using ATLAS.ti (Scientific Software 2001).

Based on our analysis, we formulated the project narrative that traces the process of requirement interpretation and shaping in the project. At the end of the research process, the project narrative was sent by email to the project manager to get her opinion about whether the narrative corresponds with reality. She suggested some minor changes, which did not affect the main findings.

4 Findings

In this section we briefly present the identified categories of technology frames and the tensions between them. Then we present our explanation of the findings in the form of a project narrative.

4.1 Technology Frames and Tensions

During the analysis, we observed four categories of technology frames that could explain the attitudes and expectations that affected the understanding of requirements of various project participants. The identified categories were:

- *Business value of system development*, i.e. the attitudes and expectations about the relationship between business and system development.
- *System development strategy*, i.e. the attitudes and expectations about the suitable system development life cycle model and processes.
- *System development capability*, i.e. the assumptions and expectations about competencies in different areas of system development, such as user interfaces and databases.
- *System development resource allocation*, i.e. the assumptions and expectations about scheduling, budgeting, and priorities of systems development projects in time-to-market pressures.

Three of these four categories match the categories identified by Davidson (2002), namely the business value of system development, system development strategy and system development capability. These three categories contain many similarities, but they are not exactly the same as Davidson's. Davidson's categories are more related to technology itself whereas our categories are related to system development.

Within these four frames, we identified a process of stereotypical 'tension' that had important effects on how the project participants emphasized technology frames in different phases of the project. We named these tensions as:

- *Filtering* that occurred when a stakeholder of the development process left something out of the scope because of his/her understanding, attitudes, expectations, or experiences.
- *Negotiating* that tried to resolve the incongruence between stakeholders. Incongruence happened when understanding, attitudes, or expectations differed among the stakeholders, causing conflicts and misunderstanding. After negotiating the understanding of attitudes and expectations were the same

- *Shifting* that took place when the understanding of a frame changed. After a frame shift, the parties involved got an understanding of a frame that was more aligned with and suitable for the current situation than before the shift

In the following narrative the process of requirements shaping and interpretation during the project is explained in more detail.

4.2 Project Narrative

The following project narrative traces how the participants' attitudes and expectations (i.e., technology frames) influenced the development through a process of filtering, negotiating and shifting tensions in the project through six episodes. Tensions and states (i.e., incongruence) in technology frames are noted in italics to indicate their influence on the perceptions and actions of the participants at a certain point in time.

Episode 1. Negotiations begin: Disagreements among project participants. When the negotiations started between BU (Business Unit) and IDU (Internal Development Unit) managers, BU only expressed a need for some consultation regarding user interfaces of the existing Electronic Commerce platform. At this time, IDU and BU managers' thoughts about the project goals and development strategies differed in a significant way. BU representatives were technically oriented persons, and they thought that they understood clearly how the existing platform should be enhanced. On the other hand, IDU representatives were business oriented persons who wanted to know more about business issues, not just technical ones. They regarded themselves as business partners to BU, not only software developers. BU managers were pressured by the company's top management to use IDU services, and they had negative assumptions about the competence of IDU in software development, especially in the area of user interfaces (*incongruence* in *system development capability*).

In a meeting held in the requirement analysis phase (further described in Episode 3), IDU's competencies and processes in systems development were questioned by BU managers (*incongruence* in *system development capability)*. The BU managers raised this issue with the IDU managers in a Steering Group meeting. The following examples demonstrate the BU managers' concerns:

- One BU manager was concerned that IDU's process model did not match the situation and worried that "IDU did not have the big picture of the project, even though the requirement capture phase took 3 weeks" (Steering Group Meeting Minute Feb 14, 2001).

- Both BU managers had a strong feeling that the "product development model of IDU is inflexible and they had spent too much time for the requirement capturing phase" (Steering Group Meeting Minute Feb 14, 2001).
- While one BU manager advised that the "requirement capturing phase could be carried out in 3 days, when the whole project group is somewhere together and defines the content of the project there" (Steering Group Meeting Minute Feb 14, 2001).
- There was a special concern about the exact point of freezing the requirements and how the changes to the requirements would be handled after that. This became clear when one BU representative wondered "if it is possible to make little changes to the functionality of the project during the project" (Steering Group Meeting Minute Feb 14, 2001).
- BU representatives were also worried about the capabilities of the IDU when one BU representative complained: "there has been no technical person in the beginning of the project… how is it possible that technical persons understand the requirements if they are not in the project and attend workshops in the beginning of the project?" (Steering Group Meeting Minute Feb 14, 2001)
- The IDU representatives saw these issues in different ways. They thought that the requirement capturing phase should be long enough to clarify all the requirements. The following are examples of how the project manager of IDU considers this issue:
- the workshops had been held very often and the working days between the workshops are used for documenting the results of the last workshop and planning the next one… Findings of the exact requirements later than in the beginning of the project will make more costs and delays to the schedule of the project. If the requirements are not clarified, the design, implementation and testing are more difficult (Steering Group Meeting Minute Feb 14,2001)
- The project manager also stressed that "the requirement specification has to be unambiguous, testable and detailed enough to avoid misunderstandings of the requirements" (Steering Group Meeting Minute Feb 14, 2001)
- The project manager responded to the concern expressed by BU on the technical capabilities of IDU that the "architect had another project to do and there is no need for technical people in the beginning of the project … the main goal in the beginning is to collect the customer's

> business requirements …" (Steering Group Meeting Minute Feb 14, 2001).

These excerpts clearly demonstrate the disagreements between the parties. BU was concerned about the rigidity of the IDU process and whether it was possible or not to respond to changes in the business and environment within the process model of IDU.

BU would have preferred external consultation, but IDU managers as well as the company's top managers preferred the use of IDU services (*incongruence* in *business value of system development)*. BU did not trust IDU competencies in software development (*incongruence in system development capability*), which naturally affected their willingness to use IDU services. These disagreements got deeper during the negotiations between BU managers and IDU managers when it was revealed that the IDU strategy was not to offer consultation, but rather software solutions (*incongruence* in *business value of system development)*. The company's top managers wanted to employ IDU services and to get the Electronic Commerce Platform service ready as soon as possible. Because of these pressures, the BU managers decided to change the goal of the project to enhance the current Electronic commerce platform with additional services and use IDU as the developer. In this way they resolved the disagreement about whether to use in-house or outsourced development (*shifting business value of system development)*.

After these tough negotiations between BU and IDU managers about the project goals, the aim of this new project was formulated as: to make a new, enhanced version of the current Electronic commerce platform. The enhancements were: "*a possibility to establish WAP, SMS and web services"* and "*Content Administrator features that make service establishment easier (currently it is made with Oracle procedures)*" *(Project Setup Letter 1.0, Feb 2, 2001)*. The Electronic Commerce project was not a high priority in IDU's project portfolio. There were many reasons for this situation, one of which was apparently the disagreement about the system development strategy between IDU and BU, which affected the managers' prioritizing decisions. This disagreement about the strategy can be seen in the following quotation (*incongruence* in *system development strategy*):

We wanted a more flexible and interactive way to develop, the IDU wanted to develop strictly according to their processes… Their process is a poor waterfall/stage model (Mary, Development Manager, BU).

As the timing of the project was extremely important, there was clearly a conflict between the actual resource allocation and the criticality of the situation (*incongruence* in *system development resource allocation*).

The project held three requirement workshops, or brainstorming meetings, where the participants discussed the system and its features. The conversation was difficult: the Business Unit (BU) preferred technical conversations, whereas the Internal Development Unit (IDU) preferred conversations from the business and end user point of view. BU and IDU saw the role of the development unit in a different way: BU viewed IDU as a technical resource and IDU considered themselves more as a business partner (*incongruence* in *business value of system development)*.

Episode 2. New version of Electronic Commerce platform: Filtering requirement understanding. These workshops produced the requirement specification document, which described the common understanding of the system requirements. This document listed only four requirements related to the Tool subsystem (table 2).

At that time, the Tool subsystem was thought to consist only of a few user interface components and these were planned to be procured from a third party. The project developers did not want to concentrate on the user interfaces because the process handbook of IDU mandated that user interface specification should start at the beginning of the system design phase. Furthermore, according to the IDU resource strategy and processes, a user interface designer is involved in the project only after the beginning of the design phase.

Req 4.	".. must have Text based UI (must have support for stand-alone version) for making/modifying services: personal computers are used as terminals."
Req. 6	"..must have a possibility to modify services online."
Arch 3.	"..Architecture must offer a 'stand alone version' for making services. This means a possibility to test services and to make demo versions easily and fast"
Arch 3.1.	"..The application for service building must have tools for importing and exporting data from and to the server"

Table 2. Requirements related to Tool subsystem

IDU's reliance on their processes, which used traditional waterfall phasing of development work, seemed to inhibit and filter the shaping and interpretation of requirements in the project (*filtering* based on *system development strategy).* Also BU's view of IDU as mainly a technical resource influenced their attitudes about the requirement elicitation process. BU viewed itself as the main responsible of the requirements (*filtering* based on *business value of system development*).

According to the IDU process model there was only an architect, a project manager and a system analyst involved in the project during the requirement elicitation and analysis phases. The lack of knowledge about user interfaces affected their attitudes towards UI development (*filtering* based *system development capability, filtering* based on *system development resource allocation)*. The architect and the system analyst did not want to be involved in user interfaces. Instead, they wanted to concentrate on the other subsystem, the service platform. This is exemplified by the following quotations:

> We did not know anything about designing UI, and what we knew, was that UI implementation is time consuming and boring… Nobody wanted to be involved in the UI thing… So we left it out in the requirement gathering and concentrated on the platform only…We left these user interfaces to the UI designer, who came into the project in the design phase according to our process model… (Tim, project architect).

Their unwillingness to be involved in the UI development led to problems with the Tool subsystem requirement analysis and restricted the designers' understanding of its requirements. Later we will see that the omission and neglecting of the user interface requirements had important repercussions for the progress of the project.

Episode 3. Requirement analysis: Negotiation between project participants to resolve disagreements. The project continued with requirement analysis and project planning for later phases. The resulting artefact of this phase, the analysis model of the system, is visualized in figure 3. Initially, the Tool subsystem was seen as a PC with an RMI connection in the right upper corner, and the Platform subsystem was the biggest part of the system. The model is based on the requirement specification document. During the requirement analysis phase, eleven new requirements concerning the Tool subsystem were discovered. These discoveries were made during the discussions between the project architect and one customer representative. These two technical persons found that the Tool subsystem was not only comprised of several user interfaces, but it was also a separate subsystem with its own functionality. This conversation drew IDU and BU closer to each other (*negotiating business value of system development)*. At the same time, IDU and BU together decided not to use a third party to implement the user interfaces of the Tool subsystem, because it proved difficult to separate user interfaces from the logic of the Tool subsystem. This decision was an important step from outsourcing towards in-house development *(negotiating business value of system development*).

According to the IDU process model, the requirements were frozen at the end of this phase. All the changes that happened during the later phases were made according to the change request procedure that was described in the project plan. This procedure included a written change request, an analysis of

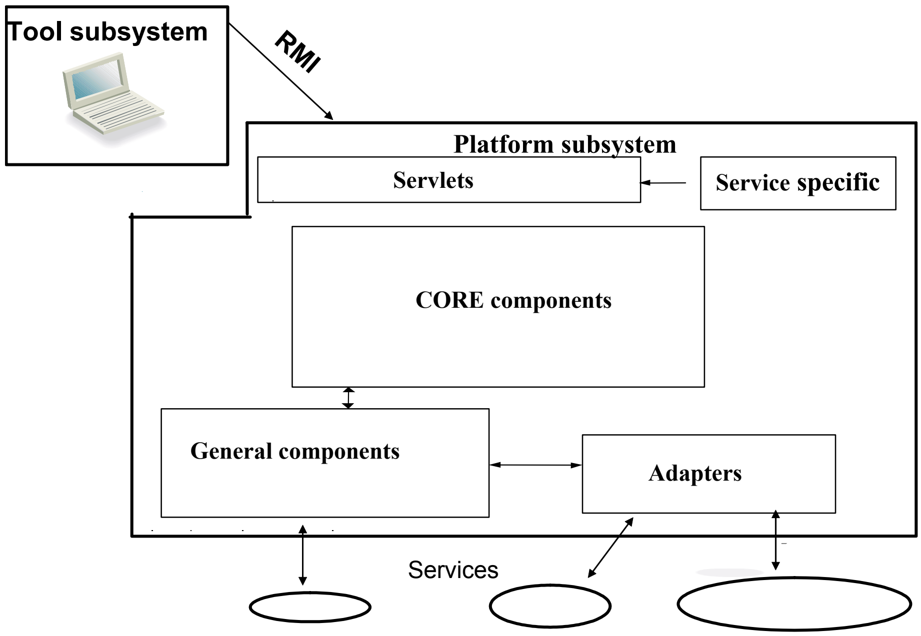

Figure 3. The analysis model of the system

the effects of this change on the project and a project steering group decision to accept or reject the change. Despite the changing requirements of the Tool subsystem throughout the project, there were only a few written change requests in the project documentation. All the changes to the requirements in the later phases were initiated in technical discussions between the designers of IDU and the BU representatives (*negotiating business value of system development)*.

Episode 4. System design: Resolving disagreements in development strategy and resource allocation among project participants. According to the project plan, the goals of the actual system development were:

to design, implement and test the new version of the platform according to the requirement catalogue

to design, implement and test the service-building tool that enables fast service building (Project Plan part 2 v29).

The project plan stated that the system will be developed according to the requirement specification. On the other hand, due to lack of time the project steering group decided that the requirement specification will not be updated, and instead, technical specifications to represent the requirements of the system will be used. The following example shows this decision.

The project decided that the:

> requirement specification document will not be updated now because of the lack of time. The update methods of the documentation will be decided later. This decision means that the approved technical specification documents are the baseline of the project requirements (Steering group meeting minutes, May 17, 2001).

This decision suggests that the project had already accepted the fact that the Tool subsystem would not be implemented according to the requirement specification and that the requirement specification document would not be updated any more. The project had to change its development strategy from a strict waterfall model to a more iterative one because of the timetable pressures (*negotiating systems development strategy*).

A user interface designer joined the project at the beginning of this phase and the first user interfaces belonging to the Tool subsystem were designed. Intensive discussions with the customer about the Tool subsystem and its features and functionality started.

The IS artefact of this phase (Design model) is presented in figure 4. As we can see from the figure, the Tool subsystem (right) has grown considerably in relation to the Platform subsystem (left).

During the user interface design of the Tool subsystem, the developers realized the challenges of its design and the size of the effort that was needed to implement it. The project manager asked IDU line managers for more resources for the design and implementation of the Tool subsystem. The IDU managers realized that the only way to continue the project was to change its priorities and to reconsider how to get through the situation (*negotiating system development resource allocation*). The following example identifies this:

> We discussed this project in the resource allocation group meeting. The project was not in a very high priority, mainly because we have more important projects going on and also because we had so many problems with the customer. .. During the discussion, we realized that we cannot go on with this project like this … We have to change the priorities and give more capable resources to the project…There were also pressures from the company's top management to change the priorities of the projects …" (Meggy, Head of department).

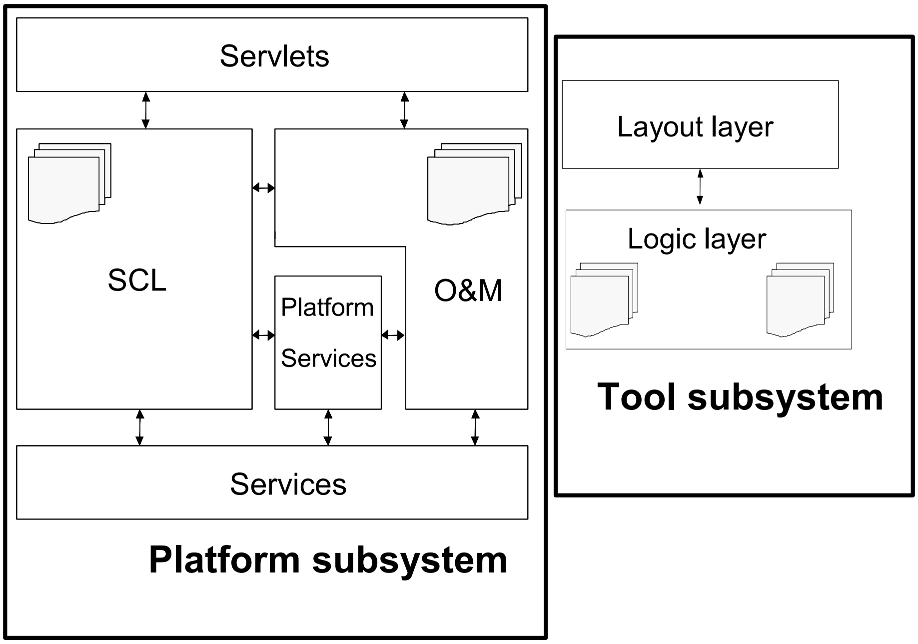

Figure 4. The design model of the system. The model is based on the design specification documents

Thus the parties had underestimated the size of the Tool subsystem and the power of the top management was used to resolve this resource allocation problem. This led to changing the focus of the development efforts.

Episode 5. Iterative development in Tool subsystem: Frame shifting. Two more people were assigned to the project, one to the Platform subsystem and another to the Tool subsystem (*shifting system development resource allocation*). Perhaps the most important push towards faster progress of the Tool subsystem implementation happened when the main designer of the Tool subsystem joined the project team. Her specialization did not include user interface implementation in the Windows environment, but she had a very positive attitude towards its challenges (*shifting system development capability*). The following quotation exemplifies shifting in the system development capability:

> the development of the Tool subsystem changed immediately after Ann came to the project… actually she did not have any experience in making user interfaces, but she had such an attitude that, yes, we can dot it … (Hannah, Project Manager, IDU).

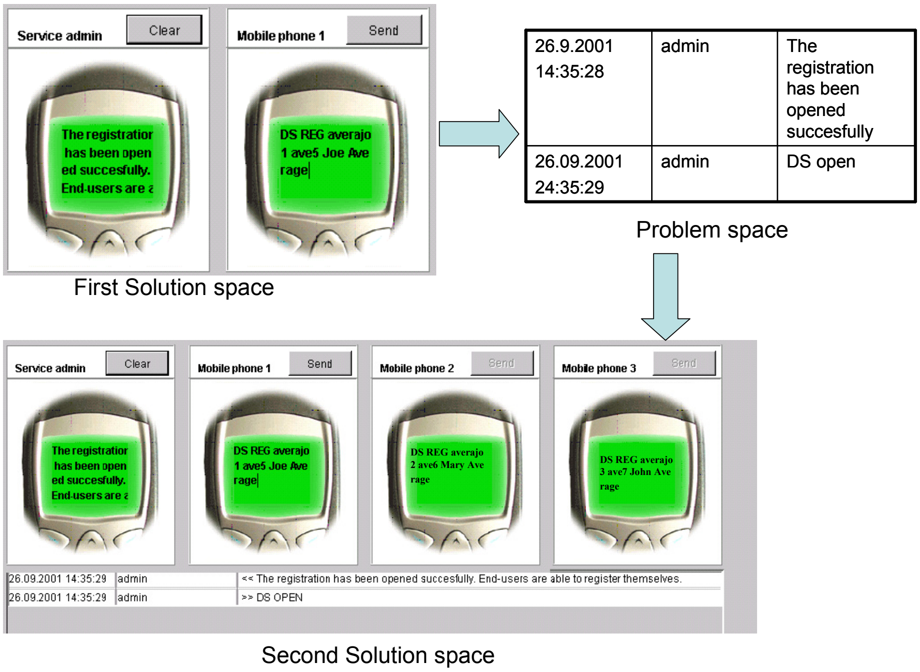

Figure 5. An example of an iteration between solution and problem spaces. Arrows represent iteration, the UI layouts the solution and problem spaces.

They formed a working team with the user interface designer and after that the implementation of the Tool subsystem really started. The architecture and implementation model of the Tool subsystem changed. Its development moved towards interactive conversations with the customer to better satisfy the customer's needs and expectations. The responsible designer of the Tool subsystem and the user interface designer made over twenty user interface specifications based on their discussions with the customer. The implementation changed from a pure waterfall style of development to an iterative walk between solution and problem spaces (Purao et al. 2002) (*shifting system development strategy*). Figure 5 shows this iteration process.

As we can see from the figure, designers in the project devised one initial solution (First Solution space) for the two user interface layouts of the Service simulation tool (Tool subsystem) and showed it to the customer. This was followed by a conversation with the customer. They discussed how this service creation process should be shown to the administrator and this raised new problems (Problem space). The problem was how the service administrator should be informed of the service simulation progress. To resolve these problems further designs were made. The designers decided to show the progress in the form of a table beneath the user interface layouts. This produced new user interface sketches (Second Solution space), which formed the starting point for a new cycle. The final IS artefact (implementation model) is shown in figure 6. The boxes inside the subsystems represent class diagrams. The size of the system is determined both by the number of class diagrams and the size of the boxes in the class diagrams. The diagrams were generated from the source code. This implementation model arose from results of over twenty iterations between problem and solution spaces (22 design specifications of the Tool subsystem). As we can see from the figure, the Tool subsystem (below) has grown to be the bigger part of the system and the Platform subsystem (above) has become smaller compared to the Tool subsystem. At the end of the project, the Tool subsystem was 60% of the size of the whole software application in terms of code size and the required development effort.

Episode 6. Outcome: Budget overrun but good IS product quality. The schedule of the project exceeded the original plan by over 40%. The estimated time schedule was planned according to the initial requirement specification. However, as described above, the requirements kept changing all the time in the Tool subsystem, causing delays in finishing it. Because of this project managers could not estimate a new time budget during the development phase. The project management could not forecast the project completion but could only react to the schedule changes. Nobody in the project could see the current state of the project and it was obvious that the changes to the project schedule always came as a surprise to most of the participants. These changes were made eight times during the second half of the development phase, and every time the project added two weeks to the schedule. The reason was always the same: the Tool subsystem was not ready.

Thus planning-wise the project itself was not successful. It was clearly a runaway project. However, BU was satisfied with the final software product. In fact, it even exceeded the customer's expectations.

Epilogue. Business partners from both inside and outside the company were satisfied with the final system. Since 2001, the system has been used in many events successfully, for instance in promoting ice-hockey games and in informing of social events inside the company. BU and IDU together have developed its features further and there have been four releases in two years after the case project ended. The development work has been more successful than during the first iteration. The interviews with project participants

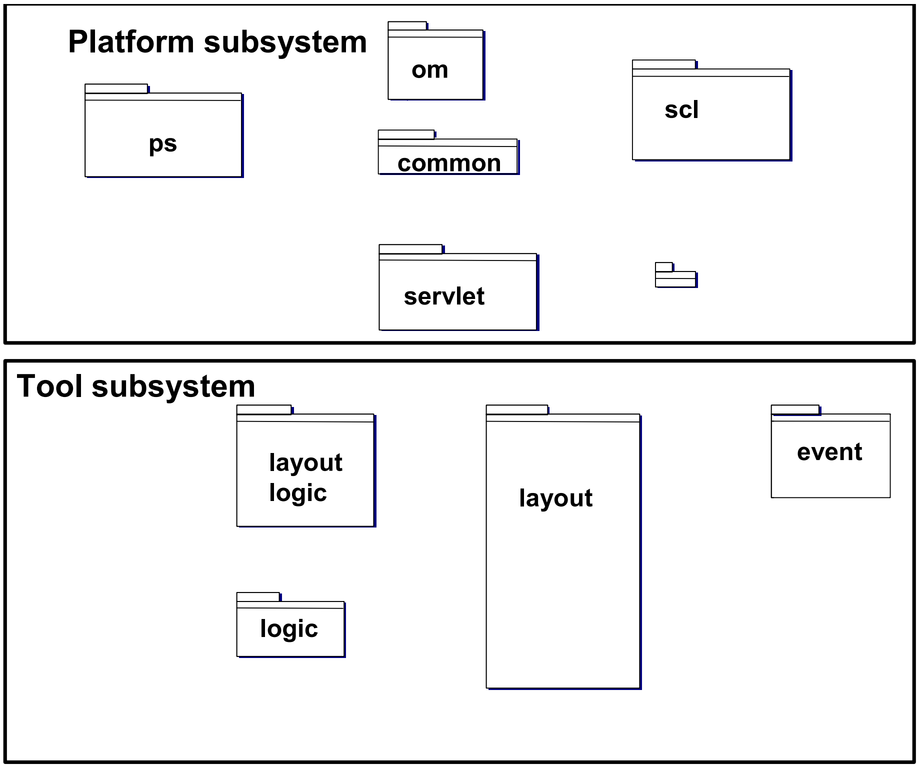

Figure 6. Implementation model of the system

revealed that they consider small projects, a limited number of new features, and small development teams to be key success factors in the later phases.

The quality manager of the company did an internal post mortem audit of the project. He was not at all happy with the project. He treated it as a failure and stated for example the following about the requirement engineering process in the project:

> The requirements were gathered in the three requirement workshops and the responsibilities of the requirements were assigned… Formally this was according to company processes, but, as later observed, all the requirements were not identified in these workshops. When the new requirements were discovered the development turned into more a 'heroic achievement' meaning working according to the situation (Project Assessment Report, September 20, 2001).

This quotation shows a very typical view of the practitioners who highlight the importance of the methods and processes and their formality in systems development. The belief that the quality of the final product is derived from formal methods and processes is not on a par with the recent survey of Neil and LaPlante (2003). They observed that although requirements are often specified informally it has no direct impact on the quality of the final product.

5 Discussion

Our observations of the Electronic Commerce project suggest that requirements are shaped and interpreted through a social and organizational process of filtering, negotiating and shifting. Four domains of technology frames—the business value of system development, system development strategy, system development capability, and system development resource allocation—affected how project participants understood the requirements. The following table (table 3) summarizes the findings of our study and explains how these tensions were present in the technology frames through the six episodes of the project narrative during the project.

In episode one, the differences in attitudes and expectations (incongruence in all technology frames) were a source of disruption in systems development. Differences in participants' attitudes and expectations redirected their attention away from the information and led them to reinterpret the final product requirements. These differences in attitudes and expectations were not directly related to the system, but they were more concerned with organizational politics related to business value and the strategy of systems development. Bergman et al. (2002) call this the political ecology of systems development. Davidson (2002) argues that these disrupting elements are attitudes and expectations towards the business value of IT and IT delivery strategies. We identified a similar phenomenon, but the attitudes and expectations observed in our study were related to systems development particularly. Furthermore, different assumptions about system development capabilities and resource allocation, also called resource politics by Bergman et al. (2002), made the development process more complicated in our case project.

Episode one showed differences in attitudes and expectations, which filtered (Filtering column in table 3) the requirement understanding and made the development process more complicated in episode two. The filtering of 'unwanted' pieces was serious, as it caused incorrect interpretation of the requirements, especially in the Tool subsystem. The observation that requirements were heavily filtered by the Internal Development Unit (IDU) because of their reluctance to deal with the user interface was of particular interest. This incorrect understanding of requirements then led to poor resource allocation and delayed especially the later phases of the development.

Tension (episode)/ Technology frame	*Filtering (episode 2)*	*Negotiating (episode 3, episode 4)*	*Frame shifting (episode 5)*
Business value of system development	BU's view of IDU as a technical resource	Emphasis in business vs. in technology among the parties The decision of outsourcing vs. in-house development	The decision to use in-house development and the change of IDU's role in the process (episode 1)
System development strategy	Strict reliance on organization's processes	The use of waterfall model vs. iterative/interactive way of development	Changes in system development towards more iterative development
System development capability	Avoiding UI software development because of the lack of skills and experiences	BU's negative attitudes towards competences of IDU vs. IDU's own reliance on their capability	Changes in capability of understanding of UI requirements through iterations between problem and solution spaces
System development resource allocation	Lack of project resources, especially UI expertise	Pre-planned priorities, schedule and budget vs. the necessity of the situation. Resource allocation according to official process description vs. the necessity of the situation	Changes in project resources

Table 3. Summary of the findings

In the later phases (episodes three and four), the project was able to resolve differences in assumptions and expectations through intensive negotiations (Negotiating column in table 3) between project participants indicating more successful elements of the project. In these negotiations the project participants' attitudes and expectations towards systems development shifted (Frame shifting column in table 3) towards mutual understanding of the situation. The negotiation process facilitated new interpretations of the requirements, which is consistent with findings in the research of information in organizations (Fiol 1994; Walsh et al. 1988; Walsh 1995). The main forces in the new interpretation of the requirements were shifting attitudes and expectations concerning the systems development strategy and systems development capability. The changed interpretations of requirements between project participants imply that the project requirements did not reside in the customer's head, but were socially constructed through interactions among project participants, proposed already by social and organizational researchers (Dagwell and Weber 1983; Newman and Nobel 1990; Waltz et al. 1993; Davidson 2002).

The project narrative addresses the elements of both the failure and the success of the project. In episodes one and two, the differences in technological frames between the groups of actors filtered the understanding of requirements and delayed the project. Episodes three to six showed a more successful performance when the participants were able to resolve these differences through negotiating, which led to shifting in the technological frames of the actors. While Davidson (2002) argues that continuous shifts in the technology frames of participants disturb the understanding of the system, we suggest further that although differences in technology frames can disturb the development, they can also improve the understanding of the final system when the main differences in the technology frames between the parties can be resolved. Bergman et al. (2002) stress that it is important to agree on functional, political, and organizational issues in the project and stabilize the situation to achieve a 'controllable' process. Our study reveals a slightly alternate view. Instead of reaching an agreement, our study shows the importance of *recognizing* and *explicitly acknowledging* attitudes and expectations that inhibit the *understanding* of the requirements.

When political researchers explain requirement elicitation as a process of loosing or gaining power by a group of participants, we explain the requirement elicitation as an organizational process of different interpretations and negotiations concerning technology between the participants of systems development. Some studies state that the initial political ecology often loses relevance while requirement construction proceeds and the goal, problems and solutions are modified and new ones are discovered (Ulrich 1983; Beath and Orlikowski 1994). This was not observable in our project and indicates that the requirements did not change during the project but the understanding of them. Episodes five and six showed that when the project could stabilize the political ecology and resolve the main differences in attitudes and expectations, new differences of technological frames were not observed any more.

As shown in episodes three to five, the project was becoming a runaway that would continue without a clear direction. In this way, the case project exhibits quite interesting phenomena regarding the escalation of commitment. However, when the parties identified the lack of direction, they were able to deescalate it by using techniques suggested already in (Keil and Robey 1999). In other words, the management resolved the differing opinions and new members were added to the staff that could handle the problematic UI issues. These seemingly simple but organizationally difficult decisions brought the real issues into focus and allowed the project to be finished successfully.

Despite the fact that our case project exceeded its deadlines quite badly, the product was of satisfactory quality for the customer. The observation that the project was satisfactory to the project participants and the customer but that the management was not satisfied with the development process corresponds to Linberg's earlier findings on project estimation (e.g. projects can feel that they were internally successful, while the management treats them as failures) (Linberg 1999; Glass 2002).

5.1 Implications for Research

The results of our study suggest that the current conceptions regarding requirement elicitation do not correspond with the needs of practice. The traditional requirement engineering research concentrates on detecting and representing requirements and ensuring that they are complete and consistent. It sees requirements mainly in a system context assuming that they already exist somewhere ready to be picked in the requirement elicitation phase. The social and organizational research provides a wider view on the requirements elicitation process. It includes political, cognitive and social processes that disturb the interpretation and shaping of requirements during the project. Our study has consequences beyond such a view. It suggests that requirement elicitation is in fact and in practice an ad-hoc and iterative process involving political, cognitive and social aspects that affect the interpretation of requirements during the whole project lifetime. Although our case project tried only to gather the requirements, present them in the requirement specification document and proceed to the design and implementation phases in a waterfall fashion according to company's process model, it turned out to be impossible. In the requirement elicitation phase of the project, the parties disagreed about business value and strategy issues, and their technology frames redirected their attention away from relevant information and filtered their understanding of system requirements. During the negotiations in the next phases, the parties negotiated, communicated and iteratively enhanced their understanding of the requirements.

The main implication of our study for the future requirement elicitation research is that it should examine requirement elicitation as a heterogeneous organizational process continuing the whole project lifetime. When these processes are understood more deeply, it is possible to develop methods that are more suitable to practical requirement elicitation and systems development work.

We observed in our study that filtering requirement understanding based on specific technology frames hindered the participants' view of the important aspects of the system. It almost resulted in a system that would have had very limited business value. Social interaction and negotiation between the parties improved the situation and facilitated the new understanding of the requirements. In this regard, further studies of requirement shaping dynamics could enhance our understanding of de-escalation in information systems projects.

5.2 Implications for Practice

The core challenge in organizational requirement engineering in practice is recognizing and explicitly acknowledging the conflicting assumptions and expectations among stakeholders inhibiting the understanding of requirements. As our study suggests, it is also important to identify the organizational assumptions and expectations which are not directly related to the system and its context. It is also important to understand that changes in understanding some requirements during the project could have a far reaching ripple effect for other requirements and the project.

The concept of a technology frame (identifying participants' attitudes and expectations about systems development) could be a useful tool when identifying these organizational obstacles, assumptions and expectations. They can help to identify the effects of developers' and managers' varying interpretations of organizational issues. Instead of trying to identify all requirements in advance, the requirement engineers should identify obstacles and emerging opportunities of requirement understanding and improvise on, or around them (Nandhakumar et al. 2003). Our study also highlights the problems of having too narrow a scope of requirements gathering or interpretation, which could lead to the omission of key information by the developers.

5.3 Topics for Further Studies

Our study suggests that the nature of systems development and requirement elicitation is not necessarily like current requirement approaches largely assume. The studied project developed new business ideas between new development partners. Four domains of technology frames—the business value of system development, system development strategy, system development capability, and system development resource allocation—appeared to be relevant in the context of a telecom operator developing electronic commerce technology. We are sure that in other situations and contexts and in a more mature business, the technology frames of individuals and groups would affect the shaping and interpretation of requirements quite differently. They can be more related to the assumptions and expectations about development technologies, such as databases or mobile phones, or perhaps their business value or their use. We will continue with our study of how requirements are shaped and interpreted in other projects and organizations and pay more attention to the recognition of the contextual issues in the project. Consequently, we can get a wider understanding of organizational processes affecting the requirement shaping in development projects.

Identifying and explicitly acknowledging different assumptions, expectations and knowledge of project participants poses a number of methodical challenges. We will seek different kinds of interviewing techniques, which can provide guidelines for eliciting assumptions and expectations. The use of metaphors (Kendall and Kendall 1993; Schultze and Orlikowski 2001) and narratives (Davidson 1997) are also worth exploring in revealing the technology frames.

6 Conclusions

Systems development practitioners and researchers are becoming aware of the messy nature of requirement elicitation. In this paper, we traced the requirement shaping and interpretation process in a project of a large telecommunication company with a single, interpretive case study. We observed that preconceptions, attitudes, and expectations among participants had a severe impact on requirement understanding. Requirements were understood through a continuous process of filtering, negotiating, and shifting.

In the beginning of our Electronic Commerce platform project differences in attitudes and expectations redirected the participants' attention away from the relevant information and filtered their understanding of the project requirements. In the later phases, the ability of the project to resolve these differences in the negotiations between participants redirected their focus towards the relevant information and led to shifting in their attitudes and expectations helping the project to make sense of the information in a new way. This sense making in the system context was an iterative ad hoc-process that happened through social interaction, communication and negotiation between the parties. In this process the participants iteratively increased their understanding of the system requirements.

The results of our study contribute to existing requirement research in an important way. This study makes a substantive contribution to the understanding of the requirement elicitation process and systems development in general. While the current approaches still largely assume that projects proceed with distinct phases in a more or less waterfall fashion and the system is developed from an understanding of the idea into a final system, which satisfies the originally stated requirements. Instead, our study implies that the requirement shaping is an ad-hoc and iterative process in which filtering, negotiating and shifting of different attitudes and expectations about systems development change the participants' interpretation and understanding of requirements during the project.

In our study, we were able to explain the failure and the success of technology development by referring to the significant differences in the technology frames between the groups of participants and their ability to resolve these differences through negotiating and shifting within these frames. The differences in technology frames between the groups of participants filtered the understanding of technology, whereas the participants' ability to resolve these differences led to a new and changed understanding of technology. In this way, our study contributes to the collective interpretations of technology by deepening the understanding of shared frames of a group and their effect on technology development.

References

Balzer, R. "A 15 Years Perspective to Automatic Programming," *IEEE Transactions on Software Engineering*, (11:11), 1985, pp. 1257-1268.

Balzer, R., Goldman, N., and Wile, D. "Informality in program specifications," *IEEE Transaction on Software Enginering* (4:2), 1978, pp. 94-103.

Bartunek, J. "Changing interpretive schemes and organizational restructuring: The example of a religious order", *Administrative Science Quarterly*, (29:3), 1984, pp. 355-372.

Baskerville R., Travis, J., and Truex, D. "Systems Without a Method: The Impact of New Technologies on Information Systems Development Projects," in *Impact of Computer Supported Technologies on Information Systems Development*, K. E. Kendall, K. Lyytinen, and J. I. DeGross (eds.), North-Holland, Amsterdam, 1992, pp. 61-93.

Beath, C. M., and Orlikowski, W. J. "The Contradictory Structure of Systems Development Methodologies: Deconstructing the IS-User-Relationship in Information Engineering", *Information Systems Research*, (5:4), 1994, pp. 350-377.

Berger, P.L., and Luckmann,T. *The Social construction of Reality*, Anchor Books, New York, 1967.

Bergman, M., King, J.L., and Lyytinen, K. “Large Scale Requirements Analysis Revisited: The need for Understanding the Political Ecology of Requirements Engineering”, *Requirements Engineering* (7:3), 2002, pp.152-171.

Boehm, B.W., and Ross, R. “Theory-W Software Project Management: Principles and Examples”, *IEEE Transactions on Software Engineering* (15:7), 1989, pp. 902-916.

Boland, R., Jr. “The Process and Product of Systems Design”, *Management Science* (24:5), 1987, pp. 887-898.

Boland, R., Jr., and Greenberg, R. “Methods and Metaphor in Organizational Analysis”, *Accounting, Management and Information Technology* (2:2), 1992, pp. 117-141.

Bostrom, R. P., and Heinen, J. S. “MIS problems and failures: A socio-technical perspective, Part I – The causes”, *MIS Quarterly*, (1:3), 1977, pp. 17-32.

Brooks, F. P., Jr. “No Silver Bullet: Essence and Accidents of Software Engineering”, *Computer* (20:4), 1987, pp. 10-19.

Bubenko J.A. Jr. “Challenges in Requirements Engineering,” *Second IEEE International Symposium in Requirements Engineering*, March 27-29, 1995, York, England.

Conklin, J., and Begeman, M. L. “gIBIS: A Hypertext Tool for Exploratory Policy Discussion,” *ACM Transactions on Office Information Systems* (6:4), 1988, pp. 303-331.

Curtis, B., Krasner, H, and Iscoe, N., “A Field Study of the Software Design Process for Large Systems,” *Communications of the ACM,* (31:11), 1988, pp. 1268-128.

Dagwell, R., and Weber, R., “System Designers User Models: A comparative Study and Methodological Critique”, *Communications of the ACM*, (26:11), 1983, pp. 987-997.

Davidson, E. J. “Examining Project History Narratives: An Analytic Approach,” in *Information Systems and Qualitative Research*, A. S. Lee, J. Liebenau, and J. I. DeGross (eds.), Chapman and Hall, London, 1997, pp. 123-148.

Davidson, E. J. “What’s in a Name? Exploring the Metaphorical Implications of Data Warehousing in Concept and Practice,” *Journal of End-User Computing*, (11:4), 1999, pp. 22-32.

Davidson, E. J. “Technology Frames and Framing: A Socio-Cognitive Investigation of Requirement Determination,” *MIS Quarterly*, (26:4), 2002, pp. 329-357.

Davis, G. B. “Strategies for information requirements determination,” *IBM Systems Journal* (21:1), 1982, pp. 4-30.

Dougherty, D. “Interpretive Barriers to Successful Product Innovation in Large Firm,” *Organizational Science* (3:2), 1992, pp. 179-202.

El Sawy, O., and Pauchant T. “Triggers, Templates and Twitches in the Tracking of Emerging Strategic Issues”, *Strategic Management Journal*, (7:2), 1988, pp. 445-474.

Eisenhardt, K. M. “Building Theories from Case Research”, *Academy of Management Review*, (14:4), 1989, pp. 532–550.

Fiol, C. “Consensus, Diversity, and Learning in Organizations”, *Organization Science* (5:3), 1994, pp. 403-420.

Fraser, M., Kumar, K., and Vaishnavi, V. “Informal and Formal Requirements Specification Languages: Bridging the Gap,” *IEEE Transactions on Software Engineering* (17: 5), 1991, pp. 454-466.

Glaser, B., and Strauss, A. L. *The Discovery of Grounded Theory: Strategies for Qualitative Research*, Chicago, Adline,1967.

Glass, R. *Facts and Fallacies of Software Engineering*, Boston, Addison-Wesley, 2002.

Glass, R., “The Mystery of Formal Methods Disuse,” *Communications of the ACM,* (47:8), 2004, pp 15-17.

Grint, K., and Woolgar, S., *The Machine at Work: Technology, Work and Organization*, Polity Press, Cambridge, 1997.

Jarke, M., Rolland, C., Sutcliffe, A., and Dömges, R. *The nature of Requirements Engineering*, Aachen, Shaker Verlag GmbH, 1999.

Keen, P. G. W. “Information Systems and Organizational Change” *Communications of the ACM* (24:1), 1981, pp. 24-33.

Keil, M., Mixon, R., Saarinen, T., and Tuunainen V. “Understanding Runaway Information Technology Projects: Results from an International Research Program Based on Escalation Theory,” *Journal of Management Information Systems* (11:3), 1995, pp. 65-85.

Keil, M. and Robey, D.,”Turning around troubled software projects: An exploratory study of the deescalation of commitment to failing courses of action”, *Journal of Management Information Systems*, (15:4), 1999, pp. 63-88.

Kendall, J. E., and Kendall, K. E. “Metaphors and Methodologies: Living beyond the Systems Machine,” *MIS Quarterly,* (17:2), 1993, pp. 149-171.

Kotonya G., and Sommerville,I. *Requirements Engineering – Processes and Techniques*, John Wiley & Sons, Chichester, 1998.

Lin, A., and Cornford, T. “Framing Implementation Management”, in *Proceedings of the Twenty-First International Conference on Information Systems*, S. Ang, H. Krcmar, W. J. Orlikowski, P. Weil, and J. I. DeGross (eds.), Association for Information Systems, Atlanta, 2000, pp. 197-205.

Linberg, K. “Software Developer Perceptions about Software Project Failure: A Case Study”, *Journal of Systems and Software* (49:2&3), 1999, pp. 177-192.

Lindblom, C. E. “Still Muddling, not Yet Through”, *Public Administrative Review*, (39), 1979, pp.517-526.

Lyytinen, K., and Hirschheim, R. “Information Systems Failures: A Survey and Classification of the Empirical Literature,” *Oxford Surveys in Information Technology* (4), 1987, pp. 257-309.

Lyytinen, K., and Robey, D. “Learning Failure in Information System Development”, *Information Systems Journal* (9:2), 1999, pp. 85-101.

Mason, R. "The Role of Metaphors in Strategic Information Systems Planning", *Journal of Management Information Systems* (9:2), 1991,pp. 11-30.

March, J. G., and Heath, C. *A primer on decision making: how decisions happen*, Free Press, New York, 1994.

March J. G., and Olsen, J. P. *Ambiguity and choice in organizations*, Universitetsforlaget, Bergen, 1976.

Markus, M. L., "Power, Politics and MIS implementation", *Communications of the ACM*, (26:6), 1983, pp. 430-444.

Miles M., and Huberman, A. M. *Qualitative Data Analysis: A Sourcebook of a New Methods*, Sage, Beverly Hills, 1984

Mock, M., and Bartunek, J., *Creating Alternative Realities at Work: The Quality of Work Life Experiment at Food Com.*, Harper Business, New York, 1990.

Nandhakumar, J., Rossi, M., and Talvinen, J., "Planning for 'drift': Implementation process of enterprise resource planning systems," in *36 th Hawaii International Conference on Systems Sciences (HICSS)*, Big Island, HI, USA, 7-10 January, 2003, IEEE Computer Society.

Newman, M., and Nobel, F. "User Involvement as an Interaction Process: A Case Study," *Information Systems Research* (1:1), 1990, pp. 89-113.

Orlikowski, W. J., and Gash, D. C. "Technological Frames: Making Sense of Information Technology in Organizations", *ACM Transactions on Information Systems*, (12:2), 1994, pp. 174-207.

Ovaska, P. "Measuring Requirement Evolution: A Case Study in the E-commerce Domain," in *Proceedings of the 6th International Conference on Enterprise Information Systems (ICEIS 2004)*, Porto, Portugal, 2004, INSTIC, pp. 669-673.

Ovaska, P., Rossi, M., and Marttiin, P. "Architecture as a Coordination Tool in Multi-Site Software Development", *Software Process: Improvement and Practice*, (8:4), 2004, pp. 233-248.

Pohl, K. "Three Dimensions of Requirements Engineering: framework and its application," *Information Systems*, (19:3), 1994, pp. 243-258.

Purao, S., Rossi, M., and Bush, A. "Towards an understanding of the use of problem and design spaces during object-oriented system development," *Information and Organization*, (12:4), 2002, pp. 249-281.

Ramesh, B., and Dhar, V., "Supporting Systems Development by Capturing Deliberations during Requirements Engineering," *IEEE Transactions on Software Engineering* (18: 6), 1992, pp. 498-510.

Schultze U., and Orlikowski W.J, "Metaphors of virtuality: shaping an emergent reality," *Information and Organization*, (11:1), 2001, pp..45-77.

Scientific Software, Atlas.ti – The Knowledge Workbench, 2001. http://www.atlasti.de/, accessed May 5, 2005.

SEI CMMI® Product Development Team, "CMMI for Systems Engineering, Software Engineering, Integrated Product and Process Development, and Supplier Sourcing Version 1.1" (CMMISE/SW/IPPD/SS, V1.1), Staged Representation. Technical Report CMU/SEI-2002-TR-012, 2002.

Simon, H. A. *Model of Bounded Rationality,* MIT Press, Cambridge, MA, 1982.
Smircich, L, and Stubbart, C. "Strategic management in an enacted world", *Academy of Management Review* (10:4), 1985, pp. 724-736.
Strauss, A., and Corbin, J. *Basics of Qualitative Research: Grounded Theory Procedures and Applications*, Sage Publications, Newbury Park, CA, 1990.
Ulrich, W. *Critical heuristics of social planning: a new approach to practical philosophy*, P.Haupt, Bern, 1983.
Walsh, J., Hendeson, C., and Deighton, J. "Negotiated Belief Structures and Decision Performance: An Empirical Investigation," *Organizational Behavior and Human Decision Processes* (42:2), 1988, pp. 194-216.
Walsh, J. "Managerial and Organizational Cognition: Notes from a Trip Down memory Lane," *Organization Science* (6:3), 1995, pp. 280-321.
Waltz, D., Elam, J., and Curtis, B. "Inside a Software Design Team: Knowledge Acquisition, Sharing and Integration," *Communications of the ACM* (36:10), 1993, pp. 63-77.
Weick, K. E "Cognitive processes in organizations", in *Research in Organizational Behavior* , B. M. Staw (ed.), Greenwich, CT, JAI Press, 1979, pp. 41-74.