Challenges in Requirements Engineering

Daniela E. Herlea Damian

Dept. of Computer Science
University of Calgary, Calgary, T2N 1N4, Canada
danah@cpsc.ucalgary.ca

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

Abstract

This paper presents a exposition of requirements engineering concepts and challenges, targeted on the research division of a large software company. It is argued that doing requirements is a difficult task and its critical problems are identified. The activities involved in requirements engineering are described and the aspects of an iterative process are presented. The state of art of the research in two of requirements activities: requirements elicitation and negotiation are described. The paper ends with a detailed proposal for an investigation of groupware support to distance requirements negotiation.

1 Introduction

In general terms, Requirements Engineering is one of the early processes of software development life cycle that involves the stakeholders of the system in an iterative process of problem analysis, requirements elicitation, validation and specification. It is about deciding precisely what to build and documenting the results.

It is an important phase in the software development, as field studies show that misidentification of requirements is one of the most significant sources of customer dissatisfaction with delivered systems (Macaulay 1996), that the incomplete requirements and specification, together with changing requirements and specifications are the main causes of system failures in North America (The Standish Group Report 1995). For many developers of large, complex software systems, requirements are their biggest software engineering problem (Curtis, Krasner and Iscoe 1988) and few would disagree with Brooks' assessment that no other part of a development is as difficult to do well or disastrous in result when done poorly.

It is not the author's intention to present a comprehensive tutorial on requirements engineering nor she claims of covering all details of all activities part of the requirements engineering. Rather, an attempt to construct a conceptual framework for requirements engineering and its challenges is made. The paper is structured as follows: the first section presents critical aspects related to requirements engineering, its processes and activities. Then, the author focuses on two activities of requirements engineering, elicitation and negotiation, and presents the state of the art of the research in each. The last part of the paper identifies the need for support of distributed requirements engineering, paralleled by developments in computer-supported collaborative work;

it concludes with a detailed proposal for an investigation of groupware support to distance requirements negotiation.

2 A CONCEPTUAL FRAMEWORK FOR REQUIREMENTS ENGINEERING

2.1 Requirements engineering: critical aspects

Although empirical evidence underscores the importance of an effective identification of requirements, there is a number of difficulties in doing requirements well. Broadly speaking, doing requirements means managing complexity and communication among people. It is about analyzing change in the customer business/operational environment and managing fluctuating and often conflicting requirements.

Although researchers in this area believe that the requirements phase has its own life cycle, a survey of literature proves difficult to give an exact or universal description of requirements activities. In the 80's Krasner identified five phases: need identification and problem analysis; requirements determination; requirements specification; requirements fulfillment; and requirements change management. More recently, Jarke and Pohl (1994) propose a three-phase cycle of elicitation, expression and validation. Potts and colleagues (1994) define an iterative cycle of documentation, discussion and evolution. It seems that different approaches use different labels for the requirements activities and this brings about one of the critical problems in requirements engineering: lack of a systematic process. This problem was brought to light by field studies (Macaulay 1996; Curtis et al 1988; Easterbrook and Al-Rawas 1996), along other requirements problems: poor communication between people, lack of appropriate knowledge and shared understanding, inappropriate, incomplete or inaccurate documentation and a poor management of people or resources in the context of continuously changing requirements. These problems are introduced here and a discussion of their impact on particular activities in the process is presented in the subsequent sections of the paper.

Requirements engineering is about people communicating with people. Communication difficulties, lack of appropriate knowledge and shared understanding translate directly into misunderstandings for technology application design and development. From a business perspective, lack of understanding about business situations and system requirements always results in less than optimum design. Business processes can be overlooked and system features might be unresponsive to the business environment or incomplete for regular operations. From a technological perspective, lack of understanding about computer potential can result in disappointment based on unrealistically high expectations for system excellence.

The failure to create and maintain appropriate requirements documents has an impact on the quality of the process outcome, as these documents capture representations of the stakeholders' expectations (Macaulay 1996). Moreover, the fluctuating nature of requirements require a good management of the process activities, people and resources, and empirical evidence (Chatzoglu and Macaulay 1995) shows that the less time is spent in the requirements stage, the less time is spent in the whole development process; hence the management of the requirements process affects not only requirements stage itself, but the whole of the project development.

To add to the complexity of the situation, it is recognized that the problems of requirements engineering cannot be solved in a purely technological way; the social context is much more crucial than in the design and programming phases. The information needed for system design is embedded in the social worlds of users and managers and, at its source, tends to be informal and highly dependent on its social context for interpretation. On the other hand, the representations used in the design are formal and relatively independent of social context. A reconciliation of these socio-technical aspects of information is regarded as the very essence of requirements engineering (Goguen 1994).

Group session approaches to requirements engineering (e.g. Joint Application Design (Crawford 1994)) that attempt to involve the right people in the process and promote cooperation, understanding and teamwork among users, developers and customers have been proposed to cope with some of these problems. While these approaches have advantages, they are sensitive to psycho-social issues in a meeting situation: for example, an effective participation of all group participants may be affected by power relationships in the group; for example, dominant behavior of individuals with organizational authority may inhibit equal participation and communication of requirements (Macaulay 1996). Other examples of such psycho-social aspects will be given throughout the paper.

In fact, a large number of approaches have been developed to address these key aspects, many more than one can cover in a paper such as this one and it is not our intent to do so. It is important to remember though that doing requirements requires the application of technical solutions in the context of human factors and any approach that does not account for both technical and human aspects is bound to have limited success (Goguen 1994; Faulk 1997).

It seems evident that the construction of a conceptual framework for requirements engineering that considers these critical aspects would be done in relation to a well-defined process of requirements engineering. However, the lack of a universal requirements process represents a problem in itself and has been mentioned above. Thus our intention is to present the activities involved in requirements engineering and discuss the impact of the above mentioned problems on these activities.

2.2 Requirements engineering: processes and activities

Activities of requirements engineering become central to the development team once a need for change has been perceived in the customer organization and a statement of organizational needs is produced.

Although the benefits of a disciplined approach to requirements engineering are known and include (1) the ability to control the project by producing standardized maintainable outputs, (2) the possibility to measure the effectiveness of the process and hence to seek ways of improving it and (3) the ability to use automated tools in increasing productivity and reducing administrative costs (Macaulay 1996), it has been proved difficult to define a universal requirements process and ad hoc processes remain the norm in much of the software industry.

A number of process phases/activities have been proposed, although there is little uniformity in authors' terminology or decomposition of requirements process. The best we can do in this report is to define a number of main phases/activities and the relationships between them. We define:

- Requirements elicitation
- Requirements analysis
- Requirements negotiation
- Requirements documentation

In practice, the process proves rather dynamic and not sequential, with a conceptual distinction among these activities rather than temporal and with the team switching back and forth among these phases. These activities are interleaved in an iterative process, and they will be described in an order that resembles the model proposed in Figure 1.

A description of each phase/activity together with illustrative examples follows in subsequent sections of the paper. The impact of the above mentioned critical on each of these activities will be discussed, together with specific psycho-socio-technical aspects.

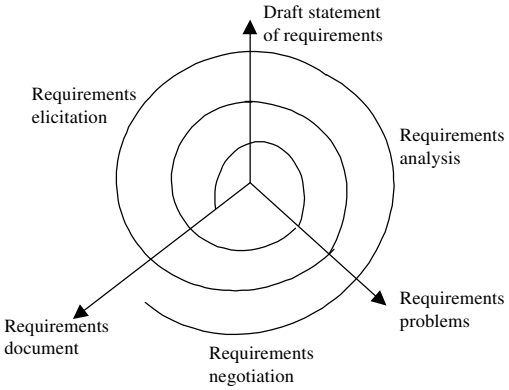

Figure 1. Requirements engineering process model (from Sommerville and Sawyer 1997)

2.2.1 Requirements elicitation

This phase is characterized by a close interaction with customers, system users and others involved with or affected by the system. Requirements elicitation is the usual name given to activities involved in discovering what the system requirements are, including: the identification of all stakeholders of the system, analysis of the problem application domain, the system's operating environment and of the customers' organizational and business environment. An effective elicitation is important because the system may not be acceptable unless it meets the customers' real requirements.

The social-technical problems introduced in section 2.1 seem to have great impact on elicitation, represented by the difficulty of expressing tacit knowledge, of making explicit what is rather implicit and situated in the customers' business and working environment. This results in statements about requirements that are ambiguous, incomplete and subject to multiple interpretations (Curtis et al 1988):

> Customer representative: I think [we] had to learn as well as [the developers]… At the time we wrote the specification, we did not appreciate that it could be interpreted any other way… This particular thing was so obvious to me as an operator, you know, it's common

The communication with the stakeholders becomes an organizational communication issue and more often restricts the opportunities for developers to talk with end users. The importance and benefits of communicating with the end users, of collaboratively analyzing their work environment and understanding the rationale for their requirements become clear in the context of a lack of application domain knowledge.

The cooperation with the end-users is essential for an understanding of the system's usage and for their understanding of the change and final acceptation of the system. This is illustrated by the following situation (Crawford 1994, p. xix) that portrays the resistance of users to give up their current work pattern. It presents a short narrative sequence of a JAD session through which the analyst attempts the redesign of a work order maintenance system whose operating environment was manual and paper intensive:

> … the center of the manual operation was a maintenance request which was written out on multi-part forms sent to various departments involved. The form has been used for twenty years by the shop foreman to receive a request to start the work and see it through to completion… In the workshop we used illustrative icons [on a board] to show the request form starting out and traveling to several involved departments… the foreman resisted the idea [of sorting a backlog of requests on the computer] and told his colleagues he would still have to use the paper form… he insisted on a carbon copy returning to his office [and] I had to place another icon of the same form in another department. Five copies in all! … the project leader and systems analyst were getting frustrated with the result…
> … in the second day, at about lunch time, the foreman mentioned, 'maybe we don't need the form in our department after all'. I started to take the icon away from the board and watched his expression pale at the thought of losing his beloved work order document. Even though others did not want it, I put it back to see his face return rosy… At the end of the day he gave us permission to remove the form and the team gave out a cheer…
> … the result was an excellent system design and twenty years of tradition changed in three days.

A good requirements elicitation supports the development of a specification of requirements which are unambiguous, complete, verifiable, consistent, modifiable, traceable. However, there is a number of specific problems that inhibit the definition of requirements with such attributes and they are described in detail in section 3.1.

2.2.2 Requirements analysis

This phase is concerned with checking the set of elicited requirements for conflicts, ambiguities, overlaps, omissions and inconsistencies. Common activities include the use of checklists for requirements analysis, the prioritization and classification of requirements, the use of interaction matrices to find conflicts and overlaps, and an assessment of requirements risks (Sommerville and Sawyer 1997). Requirements elicitation and analysis are more often interleaved activities. As requirements are discovered during elicitation, some analysis is carried out and, if problems are recognized, they need to be discussed with the source of the requirements and resolved.

Both elicitation and analysis phases make use of models for a better communication in the process. During elicitation, models of the system operational environment such as observed in the previous example (the icons placed on the board) could be used not only in expressing requirements, but also in validating the understanding of the information acquired. More abstract models such as data flow diagrams could be developed during the analysis, together with mockups, to help understand the requirements and detect inconsistencies and incompleteness.

Requirements analysis can be an expensive and time-consuming process involving skilled and experienced analysts reading documents carefully and thinking about implications of the statements in the documents, or it can be performed in cooperation with other stakeholders of the system.

The importance of analyzing the elicited requirements within the larger system operational environment together with the socio-technical problems in requirements analysis are emphasized by the following illustrated situation (Crawford 1994, p. 21): At a grain company, an inventory accounting system is being developed to be used at several grain elevators in farm country and …

> … the design included a data entry screen to capture truck weights on arrival and departure, for one truck at a time, but … would severely limit the operations needed when the traffic volume increased to typical levels. In the discussion that followed about changing the system for a more practical approach, the analyst became defensive and would not hear of redesign. The meeting degenerated into a stand-off. It ended in frustration for one in the team who said, "if you think it's good, then we think it's good".

The lack of appropriate application domain knowledge and the context-situated character of information makes the designers unable to think in the same terms as the user; this is a situation which could have ended in agreement, an effective communication among team members should have been in place.

This illustrative example provides in fact more insight into the requirements problems than merely in relation to analysis. On one hand, it shows that some requirements are realized only after some prototype has been designed and may produce a reiteration of the entire requirements cycle. Then, the designer's reaction to change could be characterized as a psycho-social effect during requirements analysis; the group becomes dysfunctional without the chance of a negotiation of the system functionality. The negotiation of requirements is the activity described next and it represents the attempt to resolve the conflicts found during the analysis phase, a better course of action than the one illustrated by this example.

2.2.3 Requirements negotiation

Conflicts appear as a source and result of the change the system brings in the users organization; conflict is inevitable when the concerns of multiple stakeholders are considered and their differing perceptions of the system are analyzed. The organizational roles and the relationship with the new system give the stakeholders diverse (sometimes conflicting) views of the system and therefore different requirements (Easterbrook 1994; Sommerville and Sawyer 1997), causing breakdowns in the communication process (Al-Rawas and Easterbrook 1996).

Without attempting a thorough classification of conflicts, we can name a few of the situations that require some sort of resolution between two or more parties in the process:

- users may be resistant to change in their own working environment, and the analyst negotiates the place of the new system in their working environment.
- stakeholders may have different interpretations of the system's role in the organization, source of possibly differing or conflicting requirements.
- without having the necessary knowledge of technological options, the users may have a great number of requirements which are not feasible from the technological or delivery schedule/cost points of view.
- stakeholders use different terminology in expressing requirements and this can create an apparent conflict.
- within the customer organization, feuds among different departments could cause conflicting requirements, as a result of trying to gain power over the system's usage in the organization.
- developers would want to produce a technically excellent system and to use latest techniques, proposing designs that are in conflict with the real customers' needs.

As it can be observed, lack of knowledge of application domain/technological options, together with political and organizational factors, contribute to these situations. Most often than not they require resolution through human communication and this is where poor communication among team members seem to have a great effect.

One of the fundamental problems here is the detection of conflict and its type (Easterbrook 1994). Then, a resolution through negotiation is desired, as in the illustrative example below (Robinson 1990). Perceiving the specified system as conflicting with her objectives, the user negotiates. At issue is the amount of computing time in which U must engage:

U: As it is now, the variety at least, I get to take my eyes away from the computer. I'm spending half of my time writing letters on the computer, and to spend the other half putting information into the computer…
U: Is the purpose to eliminate the card file?
A: Yes, yes.
U: Then I'll restate my original objection…
A: Right.
U: …of sitting in front of the computer all day and entering this information is exceedingly boring…
A: Yeah.
U: …and very hard on my eyes…
A: Right. Well, for one thing you wouldn't be entering the information. And I admit that there's no way around looking at a computer screen if this is automated. So, if that's real distasteful to you, then that could be a problem.
M: Weren't you talking about having the computer print cards with all the information…
A: That's true.
M: Or, just generate a hard copy so that you have some backup.
A: That's true, but I guess we'd have to think about it more, it seems like the best reasons for doing this - ah, that's true, maybe a card file would be useful. It seems to me that a person that likes to use computers, one of the best reasons to have this automated is because it's a lot easier to get access to information rather than looking through the log file or card file.
U: I think I find it personally easier to pull out a card file, to get the name alphabetically than it is to punch it into the computer and wait for it to come…
A: Yeah, right.

While it is important to consider the perspectives of more than one stakeholders and deal with the conflict, requirements negotiations are situations that seem to be most affected by psycho-social factors (Short, Williams and Christie 1976). While conflicts caused by different interpretations or use of different terminology may be resolved through a mutual collaboration and exploration of the differing views, the resolution of conflicts of interest may be more difficult to tackle. These are situations when both parties attempt to gain something at the other's expense, and they seem to be characterized by the need to easily perceive the other's emotions and reactions and to manipulate the other party.

2.2.4 Requirements documentation

While the previously described activities would be considered as communicative, the documentation of requirements could be described as the activity of creating and maintaining the documents involved in the process. The products of this activity, the requirements documents usually contain problem descriptions, statements on the application domain, requirements representations and models. They are used to communicate system requirements to customers, system users, managers and system developers.
According to IEEE (1984), a 'good' requirements document should contain statements which are unambiguous, complete, verifiable, consistent, modifiable, traceable and usable during the operation and maintenance phases. However, this is more often a difficult task as requirements engineering involves large amounts of information, coming in a variety of representations, all contributions of multiple sources. One dilemma faced by requirement engineers is the level of formality contained in requirements documents. If such document is to be used in the communication with system users and customers, informal and natural language representations are more likely to be successful, as they usually are not familiar with computer notations. On the other hand, if this document is to be used as the specification of what a computer system is required to do, advocates of formal representations argue that such format brings benefits in the communication with system developers, in terms

of understanding the application, the absence of ambiguity, and the identification of useful abstractions. Approaches to this problem have been proposed, by developing hypertext based mediating representations to bridge the informality/formality gap (Kaindl 1996).
To add to the complexity of requirements documents, the maintenance of requirements traceability information is an important issue given the changing nature of requirements. Traceability information concerns the source and rationale of a particular requirement, the related requirements and links to other information such as system designs and implementation. In case of a change in requirements, traceability information helps in discovering what other requirements might be affected. The fact that careful maintenance of dependencies between requirements have long-term benefits is illustrated by the following example (Crawford 1994, p. 32):

> A two-day planning workshop to define functional requirements for an inventory management and customer billing system was followed by several design specification workshops and a review meeting over a three-month period, when … a business manager from corporate finance announced a misunderstanding concerning invoice data and an interpretation of legislative requirements for reporting information on the invoice. This resulted in a change to the invoice layout and data content of summary and detail information.
> … the changes were not trivial, but the project leader easily found the affected pages and discussed the revisions in the presence of the team. This included the business planning documents and related system specification documents.

3 A CLOSER LOOK AT THE ELICITATION AND NEGOTIATION ACTIVITIES

This section presents the state of the art of the research in two requirements engineering activities: elicitation and negotiation.

3.1 Requirements elicitation: a closer look

As described in the previous section, requirements elicitation consists of the earliest activities in the requirements engineering process and even if introduced separately, it cannot be divorced from the subsequent activities. It is an important phase as many of the problems in creating and refining a system can be traced back to elicitation issues. This section describes the state of the art of the research in requirements elicitation by detailing the issues related to elicitation and introducing techniques developed to address these issues.

Yet, much of the work conducted in requirements engineering has ignored elicitation; there is a concentration of research in this area on precision, representations and modeling techniques, verification and proofs as opposed to improving the elicitation of requirements. And there is a good reason for this: requirements elicitation is a difficult process in which one has to deal with ambiguity, informality, incompleteness and inconsistency, in which the "knowledge" of the requirements is anything but overt. The emergent nature of requirements made the researchers critique the concept of "requirements capture" as if "requirements were butterflies to be caught and pinned down in a specification cabinet" (Dobson, Blyth, Chudge and Strens 1994). The actual knowledge that informs the system design is seen as the result of a modeling since what is elicited did not pre-exist but has come into being through elicitation through the elicitation process (Gaines and Shaw 1996).

Problems of requirements elicitation have been grouped into three categories as follows (Christel and Kang 1992):

- problems of scope
    - the boundary of the system is ill-defined
    - unnecessary design information may be given
- problems of understanding
    - users have incomplete understanding of their needs
    - users have poor understanding of computer capabilities and limitations
    - analysts have poor knowledge of problem domain
    - user and analyst speak different languages
    - ease of omitting "obvious" information
    - conflicting views of different users

    - requirements are often vague and untestable
- problems of volatility
    - requirements evolve over time

Despite the lack of research activity in this area, a number of techniques and methods have been developed to address these problems. Traditional and mature information gathering techniques include brainstorming, interviewing (informal or structured), use of questionnaires, and team approaches (e.g. JAD). These approaches sometimes emphasize the use of graphical notation to make the requirements models easy to read, thereby allowing these models such as data flow diagrams to form a good basis for communication between the analyst and the users.

Other elicitation approaches attempt to gather and represent the information on requirements from different viewpoints. Examples include CORE (Mullery 1987) and Easterbrook's framework (1991) of elicitation from multiple perspectives.

Recent developments in requirements engineering include methodologies and frameworks that integrate techniques for information gathering, modeling and representation of information of requirements to better support the elicitation process.
The ORDIT methodology (Dobson et al 1994) is based on soft systems methodology (Checkland 1981) and emphasizes the definition of organizational requirements as part of the elicitation process, by placing the system within the broad operational environment, with the user part integral of it.
The NATURE project (Jarke, Pohl, Doemges, Jacobs and Nissen 1994) is a large ESPRIT initiative investigating novel approaches to requirements engineering. It developed specific theories of requirements domain, process and knowledge representation, as part of a framework in which requirements engineering is a continuous process of establishing the visions of different stakeholders in a complex context.
The AMORE project (Christel, Wood and Stevens 1993) is concerned with ways in which large amounts of multimedia information can be visualized, stored and retrieved. The premise is that raw requirements can arise in many formats such as informal technical notes, meeting notes, statements of work, interviews with customers or users, operating manuals, technical and graphical notations. AMORE provides ways to capture and organize such requirements in their original form, and provide the basic environment to facilitate navigation and browsing of large quantities of material.
More recently, developments in knowledge elicitation methodologies and tools such as concept maps, repertory grids and Webgids have some applicability to group requirements elicitation (Gaines and Shaw 1996).

Ethnographic techniques as one way of revealing actual rather than formal organizational structures have been proposed for requirements elicitation (Sommerville and Rodden 1994). Goguen also uses ideas from ethnomethodology, such as the concept of situatedness, to discuss the limitations of traditional elicitation methods (e.g. interviews or questionnaires) and to better understand the information relevant to requirements. He argues that one reason for the difficulty of extracting tacit knowledge in requirements engineering is the social situatedness of the information involved. It is believed that methods of ethnomethodology such as conversation and interaction analysis embody guidelines directly applicable to many problems in requirements engineering. However, the limitation of the use of these methods alone for requirements elicitation is recognized and a "zooming" method that combines the use of ethnographic studies followed by interviews and discourse or interaction analysis in requirements elicitation is proposed (Goguen and Linde 1997).

3.2 Requirements negotiation: a closer look

As mentioned in the previous section, conflict is an integral part of requirements engineering. Considering the concerns of multiple stakeholders means dealing with a fundamental problem in requirements engineering: different groups of people perceive the system differently. During this process, the participants will disagree over how to interpret features of the application domain, what the requirements for a new system are, and how to meet those requirements. Conflicting situations have been illustrated and discussed in section 2.2.3. Requirements negotiation could be thus seen as a situation in which participants find themselves rather than a well-defined activity of the requirements life-cycle.

Although research in software engineering acknowledged the existence of conflicting requirements in software development a decade ago (Curtis et al 1988; Anderson and Fickas 1989), conventional requirements analysis

techniques used to suppress conflict, making any resolution untraceable and adding to the communication problems. Only years later research in requirements engineering (Easterbrook 1991; Easterbrook 1994; Robinson 1990; McDermid 1994) recognizes that dealing with conflicts and not suppressing them is beneficial to the quality of the meeting outcome; it is also proposed that conflicts (e.g. cognitive conflicts or of interests) that emerge in requirements engineering are to be resolved through a consideration, exploration and discussion of multiple viewpoints, in a word, negotiation.

In general terms, requirements negotiation can be seen as an iterative process through which stakeholders make tradeoffs between requested system functions, the capabilities of existing or envisioned technology, the delivery schedule and the cost (Curtis et al 1988).

The number of approaches developed to address requirements negotiation is far smaller than of approaches to requirements elicitation. Little progress has been made towards understanding how conflict may be handled in requirements engineering and this makes it a worthwhile area for further investigation. All approaches that have been proposed to deal with conflicting requirements take into consideration frameworks of multiple perspectives.

Boehm and colleagues develop the Win-Win negotiation model and its supporting tool the WInWin System (Boehm and Egyed 1998) for stakeholders' collaboration and negotiation of win-win requirements. They are both based on four artifact types: Win conditions, Issues, Options and Agreements. Win conditions capture the stakeholder goals and concerns with respect to the new system. If a Win condition is non-controversial, it is adopted by an Agreement. Otherwise, an Issue artifact is created to record the resulting conflict among Win Conditions. Options allow stakeholders to suggest alternative solutions, which address Issues. Finally Agreements may be used to adopt an Option, which resolves the Issue.

The approach developed by Easterbrook (1994) uses the viewpoint concept (Nuseibeh and Finkelstein 1992) and proposes a model for integrating conflicting domain descriptions based on a representation of multiple viewpoints. The conflict negotiation model consists of three phases: exploration of the participants' viewpoints; generation of suggestions for resolving the conflict and the evaluation of these suggestions. In fact, it is argued that the resolution is not necessarily the most important product of the negotiation process, but the extra information elicited during the process, and the participant new understanding of one another's viewpoints may be far more valuable.

The work of Robinson (1990) is concerned with the development of automated aids to promote negotiation during specification. The Oz tool (Robinson 1990) provides ways to represent conflicting perspectives and automated means to produce resolutions. Oz semi-automates a multi-perspective specification design method, in which a perspective is created to represent the desires of each type of person associated with a proposed system; then, specifications are created expressing ideal systems from the view of each perspective. Conflicts which arise during integration are resolved using negotiation techniques.

4 RESEARCH METHOD FOR INVESTIGATING GROUPWARE SUPPORT TO REQUIREMENTS NEGOTIATION

This section describes a detailed paradigm for investigating groupware support for requirements negotiation. As mentioned before, it is a worthwhile area for investigation as little understanding of how to deal with conflict has been made since conflict had been recognized as integral part of requirements engineering.

Requirements engineering is an area of software engineering where the use of advanced technology for collaboration at distance is becoming not only a necessity, but also a reality, the customers and developers increasingly defining software requirements from different parts of the world. Supporting requirements engineering at distance is not only possible nowadays, the use of cutting-edge systems being part of the new organization, but also important for the new business environments which are considering teleconferencing as an alternative to travel for face-to-face meetings.

However, the traditional face-to-face meetings are more often characterized by more intense and richer interaction, and thus the distributed setting is intuitively perceived as not effective for negotiation situations as part of requirements engineering. Moreover, the idea of facilitating requirements negotiation meetings at distance has been challenged and almost rejected by trained facilitators (personal communication with a panel of experienced facilitators at the International Conference of Requirements Engineering 1998).

The fact that the social issues at the root of many difficulties in requirements engineering cannot be modeled by the usual technical methods makes us believe that the current research of requirements negotiation offers little insight into groupware-mediated requirements negotiation: the survey of literature on requirements negotiation revealed that most research investigated computer support for conflict management (Easterbrook 1994, Robinson 1990; Boehm and Eyged 1998) with little attention to the behavioral and performance aspects of the interacting group. However, given that negotiation situations in general are most sensitive to change in communication medium (Short et al 1976), what is the impact of technology on such distance separated meetings? Do people interact differently when they use teleconferencing systems for solving equivocal tasks? How is their behavior and performance affected?

Last but not the least, such an investigation is worth pursuing given that acquiring and negotiating requirements is an integral part not only of the software development life cycle, but of any business process. Thus, the findings of such a study would help managers better evaluate the potential impacts of computer-mediated environments for dispersed settings and assist them in choosing what communication technology ought to be used in enabling effective communication for distributed requirements negotiations, not only in software enterprises but also in other business processes.

To conduct this investigation, I propose to do a study of differences between face-to-face and groupware-mediated communication in requirements negotiation situations, in terms of group behavior and performance. I propose to do this study in the following steps:

I. Study the major theories on small group interaction and performance, with an emphasis on tasks such as negotiation. Theory is essential in such studies, as a framework to guide the collection and interpretation of data for the study (McGrath 1984). Such a study is perceived as essential in understanding relevant issues to group interaction and performance.

II. Study the theories of social psychology of telecommunications and media effects. Become familiar and be critical of existing studies of variation of communication medium in similar situations. These theories may be used as a starting point in making predictions of media effects on requirements negotiation.

III. Based on the knowledge of traditional requirements negotiations and on the capabilities of existing communication technology, decide what groupware is relevant for this study. At this stage I propose to use available multimedia systems that integrate capabilities of both audio/video conferencing, synchronous access to shared whiteboards and shared files. Examples of such advanced meeting technology include Microsoft's NetMeeting and Intel's ProShare systems. I propose to use the term groupware to refer to such meeting technologies from now on.

IV. Design and conduct an empirical investigation of the difference in behavior and performance of groups negotiating requirements in two conditions: face-to-face and groupware-mediated communication. This will include a controlled laboratory experiment in which: (1) the communication medium is the independent variable and is varied from face-to-face to groupware-mediated setting and (2) group behavior and performance are the dependent variables. I propose to do this as follows:

    A. Conduct an exploratory phase, consisting of: observational sessions, interviews with experienced industry software engineers to identify relevant aspects of behavior and performance of groups negotiating requirements in traditional situations and pilot studies of the formal laboratory experiment; it is believed that this phase will provide more insight into the experimental conditions and measures of dependent variables.

    B. Conduct the formal laboratory experiment. The group size and task type/complexity will be controlled.

    C. Collect data and analyze the findings; discuss the limitations of such a laboratory experiment.

    D. Conduct field validations, in the form of discussions with experienced industry software engineers on the degree to which the lab findings align with what would be found in real work environments.

V. Analyze, interpret and discuss the findings form the entire empirical study. While significant, the lack of realism exhibited by the laboratory experiment is not necessarily fatal in this situation. The reasons of having a laboratory setting include: (1) it permits a precise manipulation of the desired independent

variable, while holding constant other variables that would normally be associated with it in field studies; (2) it allows careful observation of the intimate details of the negotiation process, which is often inaccessible in field settings and (3) it permits trying out novel conditions and strategies in a safe, exploratory environment before implementing them in the real world. The question here thus is not whether there are differences between laboratory and field settings - there always are - but whether these differences have an impact on the relationships found between variables, whether there is sufficient continuity between the laboratory and a field setting to which one wishes to generalize for a particular effect to be found in both arenas (Short et al 1976).

It is important to note that, if the software company to which this report is presented is willing to do such a study, the participants for the study could be randomly selected among the employees of the company to which this exposition is targeted Otherwise, if this is not a possibility, university students that took software engineering courses in the past could be recruited. Although a detailed experimental design is not possible at this stage (the observational sessions and pilot studies playing an important role), it is important that one of the participants in each condition play the role of the facilitator for the group. The issue here then is that trained facilitators should be invited to participate in the study. Therefore, the approach taken in analyzing the results cannot be specified for now. Although the study appears to be quantitative, it may turn in fact to be a qualitative one. Depending on the number of trained facilitators that will respond to the invitation, the number of groups in each condition will be determined. If this number is high enough (generally a number of 8 groups in each condition is considered to give statistical power to the study), a statistical analysis of the results (quantifiable measures of performance and behavior) will be carried out. Otherwise, this might not be appropriate and a qualitative description of the results would present the study findings.

5 SUMMARY

This paper presented requirements engineering as a difficult task, and discussed a number of its critical problems: lack of a systematic process, poor communication between people, lack of appropriate knowledge and shared understanding, inappropriate, incomplete or inaccurate documentation, poor management of people or resources in the context of continuously changing requirements, and psycho-socio-technical problems. A conceptual framework for requirements engineering that characterized the requirements activities and their characteristic aspects has been constructed. Then, the author focused on requirements elicitation and negotiation and the state of the art of the research in each has been described. The last part of the paper discussed the need for support of distributed requirements engineering, paralleled by developments in computer-supported collaborative work. The paper concluded with a detailed proposal for an investigation of groupware support to distance requirements negotiation.

6 REFERENCES

Al-Rawas, A. and Easterbrook, S. (1996). Communication problems in requirements engineering: a field study, **Proc. of the First Westminster Conference on Professional Awareness in Software Engineering**, London

Anderson. J.S. and Fickas, S. (1989). A proposed perspective shift: Viewing specification design as a planning problem, **Proc. IEEE International Workshop on Software specification and design**

Boehm, B. and Egyed, A. (1998): Software Requirements Negotiation: Some Lessons Learned**, ICSE'98**

Brooks, F. (1987): No silver bullet: essence and accidents of software engineering**, Computer, Apr.,** 10-19

Chatzoglu, P. and Macaulay, L.A. (1995): Requirements capture and analysis: the project manager's dilemma, **International Journal of Computer Applications in Technology 8**(3/4).

Checkland, P. (1981): **Systems Thinking, Systems Practice**. Chichester, UK, Wiley

Christel M. G. and Kang K. C. (1992): Issues in Requirements Elicitation**,** Technical Report CMU/SEI-92-TR-12 ESC-TR-92-012.

Christel M. G. , Wood D. P. and Stevens S. M. (1993) AMORE: Advanced multimedia organizer for requirements elicitation. Pittsburgh: Software Engineering Institute, Carnegie-Mellon University. CMU/SEI-93-TR-12.

Crawford, A. (1994). **Advancing business concepts in a JAD workshop setting**, Yourdon Press

Curtis, B., Krasner, H and Iscoe, N. (1988). A field study of the software design process for large systems**, Comm. Of the ACM, 31**(11)

Dobson J. E., Blyth A. J. C., Chudge J. and Strens R. (1994): The ORDIT approach to organizational requirements in Jirotka, M. and Goguen, J. (Eds.), **Requirements Engineering: Social and Technical Issues**, Academic Press, pp. 87-106

Easterbrook, S.M. (1991). **Elicitation of requirements from multiple perspectives**, PhD Thesis, Imperial College, University of London

Easterbrook, S. (1994): Resolving requirements conflicts with computer-supported negotiation, in Jirotka, M. and Goguen, J. (Eds.), **Requirements Engineering: Social and Technical Issues**, Academic Press, pp. 41-65

Faulk, S. (1997): Software requirements: a tutorial, in Thayer, R.H. and Dorfman, M. (eds), **Software Requirements Engineering**, 2nd edition, IEEE Inc., pp. 128-149

Gaines, B.R. and Shaw, M.L.G. (1996): Requirements acquisition, appeared in **Software Engineering Journal,** http://ksi.cpsc.ucalgary.ca/articles/SE/ReqAcq/

Goguen, J.A. (1994): Requirements Engineering as the reconciliation of social and technical issues, in Jirotka, M. and Goguen, J. (Eds.), **Requirements Engineering: Social and Technical Issues**, Academic Press, 165-200

Goguen J. A. and Linde C. (1997), Techniques for Requirements Elicitation, in Thayer, R.H. and Dorfman, M. (eds), **Software Requirements Engineering**, 2nd edition, IEEE Inc., pp. 110-122.

IEEE Guide to Software Requirements Specification (1984), IEEE Std 830 - 1984. IEEE Inc., 345 East 47th St., New York, NY 10017, USA

Jarke, M. and Pohl, K. (1994). Requirements Engineering in the Year 2001: On (Virtually) Managing a Changing Reality. **Software Engineering Journal**

Jarke, M., Pohl, K., Doemges, R., Jacobs, S. and Nissen, H. (1994). Requirements Information Management: The NATURE Approach**. Ingenerie des Systemes d'Informations (Special Issue on Requirements Engineering) 2**(6)

Kaindl H. (1995), An Integration of Scenarios with their Purposes in Task Modeling, **DIS'95**, Ann Arbor, Michigan USA, August 23-25.

Krasner, H. Requirements dynamics in large software projects, a perspective on new directions in the software engineering process, **Proc. IFIP**

Macaulay L. (1996): **Requirements Engineering**, Springer-Verlag London Limited

McDermid, J.A. (1994). Requirements analysis: orthodoxy, fundamentalism and heresy, in Jirotka, M. and Goguen, J. (Eds.), **Requirements Engineering: Social and Technical Issues**, Academic Press

McGrath, J.E. (1984): **Groups: Interaction and Performance**, Prentice-Hall

Mulllery (1987). 'CORE - A method for controlled requirements expression', in R. H. Thayer and M. Dorfman (eds), **Software Requirements Engineering**, pp. 304-13. Los Alamitos, Cal: IEEE Computer Society Press.

Nuseibeh B. and Finkelstein A. C. W. (1992): ViewPoints: A vehicle for method and tool integration. **Proceedings of the IEEE International Workshop on Computer-Aided Software Engineering** (CASE'92), Montreal, Canada.

Potts, C., Takahashi, K., Anton, A. (1994): Inquiry-based requirements analysis, **IEEE Software**

Robinson W. N. (1990). Negotiation behavior during requirement specification, IEEE

Short, J., Williams, E. and Christie, B. (1976): **The Social Psychology of Telecommunications**. Wiley

Sommerville, I. and Rodden, T. (1994). Requirements Engineering for Cooperative Systems. University of Lancaster. **CSCW/1/1994**.

Sommerville, I. and Sawyer, P. (1997): **Requirements Engineering: A good practice guide,** Wiley

The Standish Group Report (1995): Few IS projects come in on time, on budget, **ComputerWorld, 12**, original source: The Standish Group, “The High Cost of Chaos”, http://www.standishgroup.com.