

The Role of Asynchronous Discussions in Increasing the Effectiveness of Remote Synchronous Requirements Negotiations

Daniela Damian
University of Victoria
Engineering Department
BC, Canada
danielad@cs.uvic.ca

Filippo Lanubile
University of Bari
Dipartimento di Informatica
Bari, Italy
lanubile@di.uniba.it

Teresa Mallardo
University of Bari
Dipartimento di Informatica
Bari, Italy
mallardo@di.uniba.it

ABSTRACT

Important and yet very difficult process in software development, requirements engineering is plagued with additional challenges in the emergent dynamics of geographically distributed software teams. Our hypothesis is that a mix of lean and rich communication media are needed towards increasing the effectiveness of meetings in reaching mutual agreement when stakeholders are geographically dispersed.

We studied tool-supported remote inspections in six educational global project teams in a multicultural software development environment. In this paper we present the preliminary results from comparing the effectiveness of the requirements negotiations when preceded by the asynchronous discussions to those negotiations with no prior asynchronous discussions.

Categories and Subject Descriptors

D.2.1 [Software Engineering]: Requirements/Specifications – *methodologies, tools*; K.4.3 [Computers and Society]: Organizational Impacts – *computer-supported collaborative work*.

General Terms

Experimentation, Theory, Measurement.

Keywords

Software inspection, requirements negotiation, asynchronous communication, common ground, computer-mediated communication.

1. THE PROBLEM OF COMMON GROUND IN SOFTWARE DEVELOPMENT

Software development is a collaborative problem-solving activity where success depends on the ability to create, share, and integrate information [19]. Defining and specifying the large set of functional and non-functional system properties is an intense collaborative process that requires the development of a common understanding of the problem and solution spaces. The process of achieving an agreement with respect to system requirements is however as difficult as important in software engineering. Research in requirements engineering reports on the communication gap between the different groups of stakeholders [12].

Common ground is the knowledge that participants have in common when communicating and the awareness of it [4]. Decision making tasks, prevalent in software development and particularly in requirements engineering, are more efficient when participants share common grounds because finding or establishing common ground between people facilitates interpersonal relationships and mutual understanding. Common ground is not necessarily based on previous knowledge but it is usually dynamically established while interpreting cues obtained during social interaction.

One cannot assume common ground at the inception of a software project, given the differences in professional backgrounds of software clients and developers, as well as goals and priorities for the software system. Software inspections and requirements workshops are among the methods recommended for establishing common ground in software development. They aim at removing misunderstandings and clarifying the meaning of requirements early in software development, and both rely on collaborative activities between different stakeholders. While requirements inspections [10], [18], [15] are validation techniques that aim at assuring that the system requirements capture the stakeholders' intent, requirements workshops are means of including the relevant stakeholders in focused meetings where requirements are elicited, modeled and negotiated [14].

2. CHALLENGES OF GLOBAL SOFTWARE DEVELOPMENT AND THE TASK-TECHNOLOGY FIT THEORY

The problem is, however, that traditionally these methods largely rely on same-time, same-place interactions as people get together in a meeting room. As companies turn to outsourcing as a business model, there is a dramatically increasing trend towards distributed software development and co-located meetings are becoming problematic. The access to skilled technical staff at remote locations, lowered costs as well as possibilities for around the clock development increase the demand for distributed software engineering efforts. However, focused investigations of collaboration and requirements management processes in multi-site organizations reveal significant challenges faced by distributed teams [7]. Specifically, the distributed communication between clients and developers is problematic because of differences in software processes, professional backgrounds, and culture across sites. Failure to achieve a common understanding

of system features, combined with reduced trust and the inability to effectively resolve conflicts results in budget and schedule overruns and, ultimately, in damaged client-supplier relationships.

Computer-mediated tools however, promise to provide support to distributed development groups. Studies on media capacity theories [8] and its application to software development tasks [2] reveal some patterns of tool usage to overcome the challenges of distance. While asynchronous communication such as email was found better suited for conveyance of information (as when communicating project information that related to planning activities), real-time communication media such as teleconferencing or videoconferencing sessions were found as more appropriate for convergence to decisions (as when negotiating common understanding with respect to project scope and implementation).

Whenever engaged in group work, the communication medium which we use to exchange information must be able to reduce uncertainty as well as equivocality [2]. The former is reduced by eliminating information omission while the latter by removing information ambiguity. Translating this theory of media richness more closely to the area of computer-mediated communication, research [2] suggests that uncertainty reduction is better addressed by lean media (i.e., email and more generally asynchronous communication tools) that focus on factual information rather than emotional cues, while equivocality reduction is better handled by rich media (videoconferencing, F2F meetings and more generally synchronous tools) that provide immediate feedback. With respect to grounding processes, a number of studies show a positive relationship between the richness of communication technology and the ability to establish common ground (e.g., [16]).

The task-technology fit theory thus proves appropriate and potentially critical in understanding how computer-mediated communication supports collaborative tasks of software engineering and in particular geographically distributed software projects. To explore this further, the global aspect of distributed software teams adds an interesting yet largely overlooked dimension that may influence the perception of “task-technology” fit in software development: Culture. Empirical studies of joint decision-making provide evidence that cultural factors have an impact on the ability to support construction of common ground ([11], [16]). In [11], participants from high-context cultures (e.g., Asian) showed a preference for text-based asynchronous communication tools because they had more time to compose and revise messages before sent, while participants from individualistic low-context cultures (e.g., American) preferred talking because they could control the conversation. In [16] asynchronous text-based communication was replaced by instant messaging and the effects of culture were reduced but not removed. Further, culture plays an important role in requirements management activities and how remote stakeholders used communication tools in trying to reach mutual understanding of features and project constraints [6].

Reaching common ground and shared understanding in software requirements is a complex process that involves both uncertainty and equivocality reduction, and our position is that the task technology fit theories are useful in studying the design of improved processes and tools that support multicultural and distributed computer-mediated software teams.

3. ASYNCHRONOUS COMMUNICATION IN SUPPORT OF MORE EFFECTIVE SYNCHRONOUS REQUIREMENTS NEGOTIATIONS

Communicating and agreeing on requirements involves a constant interplay between uncertainty reduction and ambiguity reduction in requirements. Activities of requirements gathering are typically followed by modeling and inspection activities, which trigger the need to further collect information about requirements and their context (reducing uncertainty) and resolve misunderstandings or conflicts (reducing equivocality). Agreeing on requirements however is a fundamental problem in RE [12] as the result of the many (and often conflicting) goals and priorities that stakeholders have. Process models that support collaborative software engineering such as WinWin [3] include iterative cycles of identification of stakeholders’ win conditions followed by an exploration and negotiation of mutual agreements.

Our hypothesis is that multiple channels of communication (a mix of lean and rich media) are needed to improve requirements engineering activities when stakeholders are geographically dispersed. While synchronous meetings are needed for requirements negotiation sessions, asynchronous tools are valuable in providing a mechanism to structure the discussion of requirement issues before synchronous negotiations.

In the area of synchronous communication, while a large body of literature debates the usefulness of videoconferencing over audio conferencing for distributed group work, there is some consensus that the addition of video is valuable in negotiation situations and relationship building (e.g. [1]). A study of distributed requirements negotiations found videoconferencing-supported meetings as conducive to win-win situations as face-to-face meetings [5]. The comparison of group performance and behavior patterns in face-to-face and videoconferencing negotiation sessions indicated not only the feasibility of computer-mediated interaction when a rich enough medium is provided, but that certain elements in the less rich medium (videoconferencing) facilitated a more rational approach in equivocality reduction processes.

Despite the ability to create the useful rich communication medium for negotiation meetings, synchronous interaction and in particular videoconferencing sessions come with additional overhead. The necessary infrastructure is expensive to setup and maintain at remote sites, and its coordination across organizational boundaries is often problematic. We thus posit that while there has to exist support for rich synchronous requirements negotiations in distributed teams, research is needed towards increasing the effectiveness of synchronous meetings in reaching mutual agreement. We believe that most of the uncertainties can be reduced prior to the expensive synchronous communication; synchronous negotiation meetings will thus be more effectively used to reduce ambiguities in requirements. The design of a tool set to support effective requirements negotiations in distributed software teams needs to include computer-supported asynchronous communication tools that structure the discussion of requirements issues (and enable groups to develop some common ground) in addition to synchronous communication tools that allow groups to converge on issues that require equivocality reduction and were not resolved asynchronously.

4. EMPIRICAL INVESTIGATION

To validate the usefulness of asynchronous discussions in support of more productive synchronous requirements negotiation meetings, we conducted a first case study that investigated the use of an internet-based tool for asynchronous discussion of requirements issues prior to synchronous requirements negotiations.

IBIS [9] is a web-based inspection tool that supports dispersed inspection teams through sequential stages of requirements issue Discovery, Collection, and Discrimination. During the Discovery stage, inspectors review individually the document with the help of checklists or scenarios, and records issues. In the Collection stage the inspection leader or the document's author collate recorded issues and eliminate duplicates. In the Discrimination stage the inspection team makes decisions about collated issues. The Discrimination stage is designed as a structured asynchronous discussion with two mechanisms: posting of messages for each issue under discussion and voting as to whether an issue is a true issue or not (false positives).

The research setting was an instructional distributed environment that involved students in an authentic global development task, with the inherent characteristics of geographical distance and multiculturalism. The GSD experience was offered as part of a software engineering course, held in Spring 2005, in collaboration of three universities: Univ. of Victoria (Canada), Univ. of Technology, Sydney (Australia), and Univ. of Bari (Italy). The students in all three sites were assigned to six international project teams, each involving two countries. The projects were structured as outsourcing projects in which work was allocated to a geographically distributed software group in a different organization. The project outcome was a software requirements specification (SRS) as a negotiated software contract between the software group and the outsourcing company. The project finished at the point where the developer group would start writing the code for the system called for by the project.

Teamwork was critical in completing the software project as the software group had to interact with a group of clients from another country to understand the required software features. This shared understanding was developed through a series of scheduled activities of requirements elicitation, analysis, inspection, negotiation, prototype design and evaluation. Weekly videoconference meetings that also supported shared access to applications were scheduled on a weekly basis for requirements elicitation, requirements negotiation and prototype demonstration.

The requirements inspection was performed after the developers delivered a first draft of the SRS. The clients had a week to identify gaps in understanding of requirements. This inspection was entirely performed online through the use of IBIS. With the developer team considered as the authors of the requirements document, the inspection was carried out by the client team. Each member of the client team, individually, participated in the Discovery stage and read the SRS available in the system and recorded issues. The issue information contained a description of the issue found, as well as a number of issue attributes such as type severity. A course assistant collected all issues and merged duplicate issues, found by more than one client, into a list of collated issues.

For three projects out of six, the individual discovery of issues was followed by the scheduled negotiation meeting, with the

entire list of collated issues in the agenda. For the other three projects, we planned a process variant which consisted of a four-day asynchronous discussion before the synchronous negotiation meeting. The entire project team, clients and developers, participated in this discussion using IBIS (i.e. in the Discrimination stage). The purpose of the asynchronous discussion was to reach an understanding of each issue and identify those issues that could be closed online (i.e. where resolution could be reached without further negotiation) or remained open issues (everything else, and which had to be further negotiated in real-time discussion). The process of closing issues used two mechanisms in IBIS: a discussion thread consisting of messages with respect to a certain issue was created, and voting as to whether it is still an open issue or is resolved and thus could be closed. Only those issues that could not be resolved during the asynchronous discussion in IBIS (i.e. and which remained open issues) were then discussed during the requirements negotiation, which was held in a videoconference meeting between the remote developers and clients.

To assess the impact of asynchronous discussions on the synchronous negotiation meetings, we looked at the variation across projects in terms of the number of open issues resolved during the asynchronous discussions, as well as during the synchronous negotiation. We assessed the effectiveness of the synchronous negotiation sessions as the ration of closed issues during negotiation relative to the total open issues at the beginning of the negotiation. Here, we present the results from a preliminary analysis of the quantitative data we collected from the IBIS database.

Figure 1 shows the trajectory of open issues throughout the three stages in each of the six projects, as an indication of how asynchronous discussions (AD) improved the effectiveness of remote requirements negotiations. The three teams that performed AD in IBIS were: Australian-Canadian, Australian-Canadian and Italian-Canadian. It can be seen that all three dotted lines, representing projects with AD, finished below the three continuous lines (which correspond to projects with no AD). To note is project B2 which started with the highest number of collated issues (112) but which ends with a significantly lower number of open issues (3). We have obtained preliminary feedback from the participants that this was due to the asynchronous discussion.

To obtain a more comprehensive understanding of the processes of achieving common ground in requirements and the support given by the asynchronous discussions prior to the synchronous negotiations, our current efforts are in a more in-depth analysis of the group communication throughout the projects under study. We are applying content analysis methods to the asynchronous discussion records in IBIS, as well as qualitative analysis of the videotaped videoconferencing negotiation sessions. We are interested in identifying evidence of common ground established as a result of asynchronous discussions and which directly contributed to solving issues during the synchronous requirements negotiations. Information on the severity of each requirements issue, the depth of analysis for each issue, the number of discussants in each issue discussion, as well as the terms/language used in resolving discrepancies in requirements will inform our understanding of the extent to which the mix of lean and rich media, in this case synchronous negotiations preceded by asynchronous discussions was beneficial in a team separated by

language, time zones and cultural/professional backgrounds and, more importantly, different initial understanding of the problem at hand.

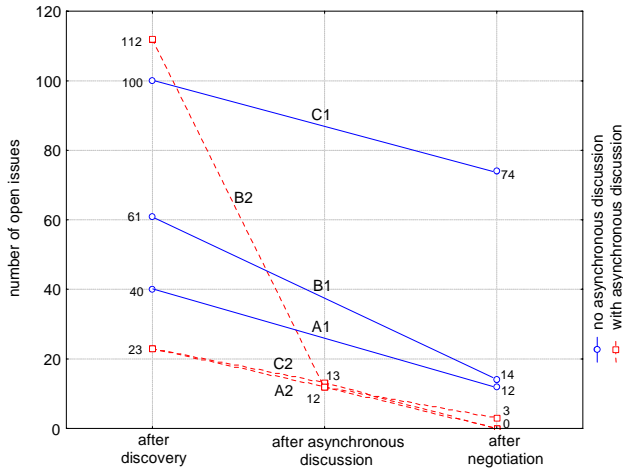


Figure 1. Open issues at the end of three process stages

5. SIGNIFICANCE OF RESULTS

Developing common ground is one of the most challenging problems of distributed teams, due to inherent communication gaps among multi-cultural geographically distributed stakeholders. The usefulness of asynchronous discussions prior to requirements negotiations in focusing the synchronous negotiation meeting on the issues that could not be resolved during the asynchronous discussion and the preliminary results in our research confirm this thesis. The task-technology fit provides us with tools to understand the appropriateness of asynchronous media for reducing missing information in requirements, as well as of synchronous media for negotiating mutual agreement on those issues that require more than uncertainty reduction.

Today's synchronous technologies for distributed groups are still limited in the quality of communication and ease of use. Even if forthcoming advances in technologies will increase the chances of successful adoption of synchronous communication such as videoconferencing and shared access to applications, yet coordinating meetings and technology infrastructures across multiple sites in distributed development is likely to stay problematic. Working towards increasing the effectiveness of synchronous requirements negotiations is thus an endeavour with long term impact.

An understanding of factors that contributed to synchronous requirements negotiations to be more effective when preceded by the asynchronous discussions in IBIS has the potential to inform tool as well as process design for distributed requirements management activities. This addresses an emerging need of multicultural and distributed software teams: the ability to use a computer mediated environment that integrates asynchronous and synchronous capabilities in a manner that fosters effective requirements negotiations. This is the focus of our research and we hope to report on specific details of such tool and process design in the near future.

6. ACKNOWLEDGMENTS

We thank to Dr. Ban Al-Ani, Luis Izquierdo, Fabio Calefato and to all the students who participated to the remote projects.

7. REFERENCES

- [1] Abel, M. Experiences in an exploratory distributed organization, in Galegher et al. (eds.). *Intellectual Teamwork*, (1990), 489-501.
- [2] Andres, H.P. A Comparison of Face-to-Face and Virtual Software Development Teams. *Team Performance Management: An International Journal*, 8, 1/2 (2002), 39-48.
- [3] Boehm, B., Bose, P., Horowitz, E., and Lee, M.J. Software Requirements as Negotiated Win Conditions. In *Proc. of the ICRE*, (Colorado, 1994).
- [4] Clark, H.H., and Brennan, S.E. Grounding in Communication. In L.B. Resnick, R.M. Levine and S.D. Teasley (eds.). *Perspectives on Socially Shared Cognition*. American Psychological Association, Washington, DC. 1991.
- [5] Damian, D.E., Eberlein, A., Shaw, M.L.G., and Gaines, B.R. The effects of communication media on group performance in requirements engineering. *IEEE Software*, (2000), 28-36.
- [6] Damian, D., and Zowghi, D. An insight into the interplay between culture, conflict and distance in globally distributed requirements negotiations, In *Proc. of the HICSS*, (January 2003).
- [7] Damian, D., and Zowghi, D. Requirements Engineering challenges in multi-site software development organizations. *Requirements Engineering Journal*, 8 (2003), 149-160.
- [8] Dennis, A.R., and Valacich, J.S. Rethinking Media Richness: Towards a Theory of Media Synchronicity. In *Proc. of the HICSS*, (January 1999).
- [9] Lanubile, F., Mallardo, T., and Calefato, F. Tool Support for Geographically Dispersed Inspection Teams. *Software Process: Improvement and Practice*, 8, 4 (2003), 217-231.
- [10] Lanubile, F., Shull, F., and Basili, V.R. Experimenting with Error Abstraction in Requirements Documents, In *Proc. of the METRICS*, (March 1998), 114-121.
- [11] Massey, A.P., Hung, Y.C., Montoya-Weiss, M., and Ramesh, V. When culture and style aren't about clothes: perceptions of task-technology "fit" in global virtual teams. In *Proc. of the GROUP conference*, (September 2001).
- [12] Nuseibeh, B., and Easterbrook, S. Requirements engineering: a roadmap. In *Proc. of the ICSE*, (May 2000), 35-46.
- [13] Olson, G.M., and Olson, J.S. Distance matters. *Human-Computer Interaction*, 15 (2000), 139-179.
- [14] Macaulay, L. *Requirements Engineering*, Springer, 1996.
- [15] Parnas, D.L., and Lawford, M. Guest eds. Introduction: The Role of Inspection in Software Quality Assurance. *IEEE Transactions on Software Engineering*, 29, 8 (2003), 674-676.
- [16] Setlock, L.D., Fussell, S.R., and Neuwirth, C. Taking it out of context: collaborating within and across cultures in face-to-face settings and via instant messaging. In *Proc. of the CSCW*, (November 2004).
- [17] Short, J., Williams, E., and Christie, B. *The social psychology of telecommunications*, Wiley&Sons, New York, 1976.
- [18] Shull, F., Rus, I., and Basili, V.R. How Perspective-Based Reading Can Improve Requirements Inspections. *Computer*, 33, 7 (2000), 73-79.
- [19] Walz, D.B., Elam, J.J., and Curtis, B. Inside a software design team: knowledge acquisition, sharing, and integration. *Communications of the ACM* 36, 10 (1993), 63-77.