Learning Human Aspects of Collaborative Software Development

Irit Hadar
Department of Management Information Systems
University of Haifa
Haifa, 31905, Israel
hadari@mis.haifa.ac.il

Sofia Sherman
Orit Hazzan
Department of Education in Technology and Science
Technion – Israel Institute of Technology
Haifa, 32000, Israel
sonyashe@techunix.technion.ac.il oritha@techunix.technion.ac.il

ABSTRACT

Collaboration has become increasingly widespread in the software industry as systems have become larger and more complex, adding human complexity to the technological complexity already involved in developing software systems. To deal with this complexity, human-centric software development methods, such as Extreme Programming and other agile methods, have been developed and implemented. Aiming to prepare future software developers for today's software industry, this paper presents a framework for developing collaborative learning tools and activities, and examples that were developed for the course "Human Aspects of Software Engineering" in order to assist students in learning collaborative software development. The learning processes and knowledge construction undergone by the students in the study were examined empirically, both in general and with respect to collaboration in particular. Results indicate that, based on their individual and group in-class experiences and reflections, students developed skills and constructed both practical and theoretical knowledge relating to successful collaborative software development.

Keywords: Software Engineering, Software Development, Software Engineering Education, Collaborative Learning.

1. INTRODUCTION

The course "Human Aspects of Software Engineering" (Tomayko and Hazzan, 2004), offered to seniors at the Management Information Systems (MIS) department at the University of Haifa, opened with the question: What are the human aspects of software engineering? Students' initial responses surrounded one central issue: collaboration. These responses included themes such as teamwork and cooperation, mutual trust, the challenge of integrating contributions from different people, multiple perspectives of a single project, work allocation between team members, and so on. While collaboration in itself is an important part of the human aspects involved in software engineering (SE), other issues are also important, such as motivation, cognitive processes, work experience and professional skills. These issues were raised by the students only at a later stage of the discussion, and only after the instructor dropped them some hints. The students' responses suggest that collaboration is perceived as a central and very challenging issue in software development processes.

This perception is quite closely tied to reality. While the SE industry deals with the ever-increasing complexity of its products, collaboration among different people participating in the same development project is essential and has already been considered as an everyday part of professional software development (DeMarco and Lister, 1999; Humphrey, 2000; Izquierdo et al., 2007; Sharp and Robinson, 2007; Venolia et al., 2005). Multi-participant collaboration adds to the already high technological complexity as well as to the many challenges related to human aspects created or affected by such collaboration. Today, many software development teams demonstrate collaborative work by using special tools and methods, such as Extreme Programming (Cf. Beck, 2000) and other agile software development methods (Cockburn, 2001; Highsmith, 2002), as well as internet-based multi-site cooperation tools that support remote (sometimes even international) collaborative software development (Herring and Rees, 2001).

These changes in industry call for an adaptation of learning tools and environments in order to prepare future generations of software engineers. Thus, our research objective is to find ways, based on existing theories and principles of effective collaborative learning, for teaching collaborative software development. The research questions derived from this objective are:

1. What are the characteristics of effective collaborative learning?
2. How can the characteristics identified in (1) be supported by teaching tools/activities in the context SE?
3. How do students exhibit collaboration when learning collaborative software development?
4. What are the learning processes and knowledge construction taking place?

One approach that seems appropriate here, and which has attracted much attention in the literature, is *collaborative learning*. Collaborative learning is usually aimed at constructing students' knowledge on a given topic of interest. Not much attention, however, has been paid to actually learning *how to collaborate* (Burton et al., 1997). It is our view that collaboration itself should also be taught, and particularly in the context of SE. Moreover, we believe that the most effective way to prepare students for collaborative work in the software industry is to expose them, in class, to an active collaborative work experience, followed by a reflection process and analysis of knowledge construction. Specifically, the guideline we followed was *to use collaborative learning for the learning of collaborative software development*.

In order to answer our research questions according to the above guideline, we conducted our research in two phases. The first phase included building a theoretical framework comprised of existing and emerging theories and principles of effective collaborative learning. This framework was later used for the development of respective activities and a tool so that students could actively practice collaboration in software development processes. In the second phase, we explored and identified learning processes and knowledge construction that took place with respect to collaborative software development when facilitating these activities and using the tool. To this end, we conducted a qualitative empirical study within the course "Human Aspects of Software Engineering" (Tomayko and Hazzan, 2004) offered to seniors at the MIS department at the University of Haifa.

The remainder of this paper is organized as follows: Section 2 reviews the collaborative learning principles and experience, on which we based the theoretical framework for developing collaborative learning tools and activities. Section 3 presents the research methodology and setting and, in particular, the course in which the research was conducted. In Section 4 we present several illustrative collaborative learning activities and a collaborative tool as well as examples of observations obtained during their application. Section 5 discusses the suggested framework in light of both theory and practice. We conclude in Section 6.

2. THEORETICAL FRAMEWORK

Roschelle and Teasley (1995) define *collaboration* as "the mutual engagement of participants in a coordinated effort to solve a problem together" or as a "coordinated, synchronous activity that is the result of a continued attempt to construct and maintain a shared conception of a problem". While the first definition is more process-oriented, the second one focuses on its artifact.

The *collaborative learning* approach implements the principles of collaboration for the purpose of effective learning. According to Alavi (1994), three attributes of effective learning processes can be identified in the literature in the area of cognitive learning theory: (1) Active learning and construction of knowledge; (2) Cooperation and teamwork in learning; and (3) Learning via problem solving.

Active learning is accomplished by engaging students in the construction of knowledge by acquiring, generating, analyzing, manipulating, and structuring information. *Cooperation and teamwork* lead to social processes that occur more effectively through interpersonal interactions in a cooperative (versus competitive) context. This enables team members to monitor individual thinking, to provide feedback for clarification and change in perception (i.e. learning), and to be exposed to alternative points of view. *Learning via problem solving* is supported by the view that learning is a process of building and transforming mental models. In problem-solving situations, mental models are tested, extended, and refined until they are effective and reliable in solving the said problem (Alavi, 1994).

According to Shuell (1986), learning strategies that encompass these three attributes of effective learning have been promoted more than traditional strategies that involve passive (versus active), competitive (versus cooperative), and individualistic (versus group-oriented) learning.

An additional attribute that is acknowledged as very effective in learning processes is *Reflection* (see Schön, 1983, 1987) for a general discussion about reflection as a professional practice and Hazzan (2002) for a discussion on reflection in SE processes). The reflection's objective is to consciously analyze different elements and aspects that took place during the learning activity. The main role of the reflection process in our context is to raise the students' awareness to the other three characteristics so as to enhance the learning processes involved. We apply this forth attribute as part of the collaborative learning framework we suggest.

Nunamaker et al. (1991) describe the gains and losses of collaborative learning, as follows:

Group Process Gains:

- A group as a whole generates more information and alternatives compared with a single average group member;
- Groups are more effective and objective when performing evaluation and error detection tasks;
- Working in a group can motivate the individual member to perform better;
- Interactions among group members lead to synergy.

Group Process Losses:

- Participation of members in the group process is fragmented (i.e., group members should take turns speaking);
- One or a few individual members might dominate group discussions and monopolize the group's time;

- Fear of negative evaluation (evaluation apprehension) can cause members to withdraw and avoid participating in group discussions;
- Higher volumes of information generated during the group process create information overload for individual members.

Upon examination of these gains and losses, Alavi (1994) found that the effectiveness of collaborative learning can be further enhanced by applying computer and communication-based capabilities in the form of group-decision support systems.

Trying to benefit from such systems, Brush et al. (2002) found that while online discussions hold great potential for extending in-class discussions beyond the classroom door, integrating these two discussion types is challenging, since they were found to compete with each other. Rather than serving as a starting point for in-class discussions, participants in the online discussions seemed uninterested in addressing the same issues again in class. Thus, we believe that a better approach for integrating these two types of discussions is by starting the discussion in class, and leaving some questions unanswered, to be dealt with during the online discussion. This principle is applied to all of the activities we designed (as will be further elaborated in Section 4).

Guzdial et al. (2001) found that engineering and math students are less willing to collaborate than their peers in other fields, sometimes actively avoiding collaboration, despite the friendly tools and mandatory course assignments they receive. Guzdial et al. presented three explanations for what they called "active resistance to collaboration":

(1) *Competition and single-answer assignments.* Engineering and mathematics students tend to see their homework assignments as having only one correct answer (even when that is not the case). Hence, they are not willing to collaborate, believing this will cause them to lose their relative advantage. In contrast, classes in which collaboration is most successful (e.g., English composition, architecture, object-oriented design) are classes with a heavy emphasis on design, in which there are many plausible correct answers to a given task.

(2) *The challenge of seeking help.* Students who are confused or have little confidence in their solution refuse to collaborate, wishing to conceal their errors or to avoid admitting their confusion. The paradox here is that these students are those who need help the most. This tendency is heightened in cases in which a competitive class atmosphere is observed.

(3) *Faculty attitudes and models of collaboration.* If instructors do not support collaboration, they might not convey to their students what collaboration is about or how and why they should practice it. In cases in which collaboration is seen to succeed, classes are organized around discussions; in classes in which no discussion or collaboration takes place, students, and sometimes even faculty, simply do not know how to collaborate.

The conclusion from the above-presented studies is that collaboration should be used in the correct context in order for it to succeed. Namely, it should be integrated with in-class learning rather than compete with it; assignments should be designed to complement the collaborative learning method, for which many opinions are in place and there can be more than only one correct answer. Finally, an open, non-competitive class atmosphere and collaborative work should be encouraged by the instructors who should use in-class discussions and collaborations and demonstrate the process and outcomes of collaborative learning.

The theoretical framework we suggest (Figure 1) encompasses our view of the overall components and attributes for achieving effective collaborative learning. The basic component is the collaborative activity (the inner frame) which is based on the 4 attributes presented above for effective learning. Moreover, the purpose of further enhancing the effectiveness of collaborative learning, namely increase gains and reduce losses of this approach, can be met by applying computer and communication-based capabilities. For this aim, a collaboration tool of this type is required as a supportive infrastructure. Lastly, a proper learning environment, with an atmosphere encouraging collaboration, should be created and maintained.

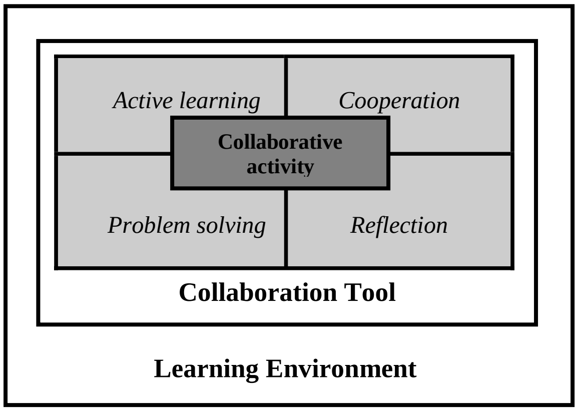

Figure 1: A Framework for Effective Collaborative Learning

3. METHODOLOGY AND SETTING

Our goal was to develop a teaching method for collaborative software development, based on the theoretical framework presented, and explore the learning processes and knowledge construction taking place when it is applied. The teaching method, including the collaborative learning activities and tools, was designed and used within the advanced elective course "Human Aspects of Software Engineering" offered to seniors at the MIS department at the University of Haifa and taught by the first and second authors of this paper.

3.1 The Course

The "Human Aspects of Software Engineering" is based on a course originally developed by Tomayko and Hazzan (2004, 2005). Appendix A presents the course outline.

The objective of the original course (Tomayko and Hazzan, 2004) was to discuss different human – cognitive and social – aspects of SE such as program comprehension, development methods, processes and products, teamwork, ethics, abstraction, and more. It also included specific activities in which human aspects play an important role. The original course was used as a platform for the "Human

Aspects of Software Engineering" course taught here, but several adjustments were made.

Since we wished to find ways to teach collaborative software development, two main objectives guided us when we taught this course: a) Teach students to collaborate and b) Teach students human aspects of software engineering, focusing on collaboration-related issues. For these aims, we used simulations of collaborative software development assignments in which students actively practiced collaboration followed by reflections in which students analyzed different aspects of their own experience, and ended with group discussions of the assignments' meanings and implications.

3.2 The Participants

The participants of the course were ten MIS students in the last semester of their studies who had already studied and practiced all life cycle activities of software development, namely requirements analysis, design, implementation, and testing. Further elaboration of their background is presented below.

In parallel to this course, all students took the "Yearly Project" seminar. This was very useful, since some of the discussions and students' tasks were based on their experience in the project. Note that it would also be possible to apply the suggested teaching method (as well as conducting this study) if this was not the case. This, however, would only require more time for the accomplishment of students' assignments.

In the beginning of the semester, the participants were required to fill in a questionnaire we designed in order to evaluate their background. This questionnaire included three types of questions: facts-related questions (for example, regarding their educational background and aggregated experience); knowledge related questions (for example, definitions of the different development phases); and questions regarding their attitudes towards SE-related issues (such as human aspects and their impact on software development). From this questionnaire we learned that:

- All students learned the following courses or their equivalents (we refer here only to courses relevant for our study): Introduction to Computer Science, Design and Implementation of MIS, Object Oriented Programming, and IS Analysis.
- All students developed software systems in teams (of 2 students or more) during their studies.
- While several students worked in industry, none reported programming experience gained outside the university studies.
- All students demonstrated fair knowledge with regard to basic SE concepts.
- With regard to issues of human aspects, the students referred repeatedly to issues related to collaboration, such as the helpfulness or interference of other team members, task allocation, disputes, synchronization, etc. This demonstrates once again the attitude we observed in the class discussion (described in the introduction) that collaboration is perceived by the students as central and very challenging in software development processes.

3.3 The Empirical Study

Since our aim was to explore and understand phenomena related to the proposed teaching method, we conducted an empirical study, applying qualitative research methods (Bassey, 1999; Bogdan and Biklen, 1992).

Data was gathered using the following tools:

- Online forum where students' reflections and discussions were documented (65 students' responses);
- Observations of class discussions and teamwork that were recorded and transcribed (16 hours of observations);
- Questionnaires (in the beginning, during, and end of the course) regarding perceptions of, and attitudes toward, relevant SE topics;
- Individual interviews with students at different stages of the course;
- Students' homework assignments (after each activity).

All textual data retrieved via the aforementioned research tools were analyzed by text analysis applying the inductive approach (cf. Bogdan and Biklen, 1992; Strauss and Corbin, 1990). In this analysis approach, categories emerge from the data and are validated and refined throughout the analysis process. Our aim was to identify learning processes and knowledge construction taking place when learning how to collaborate. Thus, the categories emerging from the analysis referred to learning-related phenomena. These categories are presented in the next section (Table 1), and a demonstration of the analysis process is can be found in Appendix B.

In the results section we describe the activities and the tool we designed and their expected benefits based on the collaborative learning literature. In addition, we present examples from our observations that took place when applying each of these activities. Note that it is not our intention to prove that this teaching method leads to better performance relative to some other teaching method; rather, our purpose is to identify and describe the learning processes that took place when using this teaching method.

4. ACTIVITIES AND LEARNING PROCESSES

In what follows, we present one collaborative tool and four activities designed to support the learning of collaborative software development. Note that these activities and tool are brought as examples for how to use the suggested framework for collaborative learning.

Table 1 summarizes the four activities and illustrates their collaborative aspect in the context of human aspects of SE as perceived by the students in the course. The collaborative aspect of these activities is vast; for the sake of brevity, however, for each activity we present only two observations that relate to collaborative learning. The phenomena stated in Table 1 are descriptions of categories that emerged from the text analysis. A demonstration of the analysis processes conducted in this study that led to these findings can be found in Appendix B.

In all the activities described in Table 1, the four attributes of collaborative activity were reflected as follows.

Active learning: students' experience in the activity led them to construct perceptions as to what factors may lead to, or interfere with, the success of coping with the task at hand, thus constructing knowledge and skills for coping with similar tasks in the future. *Cooperation*: two aspects of cooperation took place here. First, all activities dealt with issues related to cooperation; and, second, the students were required to handle their tasks in cooperation with their team members. *Problem Solving*: In all activities the students were required to present a solution to a defined problem. *Reflection*: The students reflected on the task they executed and their decision-making processes, individually and within the teams, using the online forum.

In parallel to the activities described in Table 1, we used a collaborative learning tool (as presented in Figure 1). This tool was an internet forum, used and managed within the course's website, and served as a platform for students to communicate and discuss different issues. The forum was used following class activities, assignments and discussions, continuing the learning process beyond the classroom doors (Brush et al., 2002) and implementing the benefits of computer- and communication-based capabilities for supporting group decisions (Alavi, 1994).

In addition, the forum serves also as a reflection tool. After each in-class activity, the students were instructed to reflect on that activity, relating both to what they had experienced individually and as a team. In each reflection,

Activity	Description	Examples of observed phenomena
Project Planning	Allocation of modules to teams and tasks to team members, and planning an overall schedule for an SE project. This involved both intra-team and inter-team planning.	(1) The perception of the students with regard to the success of the mission was closely related to the degree of their active participation in the task. (2) The students constructed perceptions as to what factors contribute to or interfere with the success of the mission.
Bonus Allocation	The students were presented with specific situations of teamwork in SE, were required to make individual, and then team, decisions regarding their preference of bonus allocation (individual versus team bonus), and to analyze the relationships between reward and cooperation in the context of software development project. (Based on Hazzan, 2003).	(3) The students tended to choose the percentage of the personal reward based on their assessment of their personal skills relative to those of the other team members (preferring high individual reward when assessing their skills to be higher then their peers' skills). (4) The students constructed new views with respect to the desired combination of skills and levels of team members and appropriate bonus allocations for different combinations.
Ethics	The students were presented with several ethical dilemmas concerning SE, focusing on human aspects and specifically on collaboration, and were required to suggest solutions. Based on these experiences, as well as their own previous ones (e.g. from their yearly project), the students were required to suggest a set of ethical rules to guide software engineers.	(5) The students successfully suggested ethical rules, many of which were similar to the ones published in: http://www.acm.org/constitution/code.html (6) Inconsistently with the suggested rules, the students' tendency to actually apply these codes of ethics depended largely on the context (e.g. passive verses active situations, physical verses virtual environments, and norms of the organization within which they act).
Teamwork	Toward the end of the semester, each pair of students was instructed to observe another pair during a project development session (of at least two hours) within the framework of their yearly project. The students observed, documented and analyzed the observations in light of the information they had acquired and knowledge they had constructed during the entire course.	(7) The approach of most students to this analysis was to compare what they observed to the way they themselves implement teamwork. It was a somewhat judgmental approach, implicitly referring to one approach (their own) as the "right" one and evaluating the other accordingly. (8) Some elements that contribute to teamwork, which were exhibited by all student pairs, and were then agreed upon in the follow-up discussion in class were: (a) the need for prior acquaintance with the other team members, specifically their specialties and limitations; (b) switching roles among team members, particularly while working on the computer; (c) listening to the other team member and considering each opinion.

Table 1: Examples of Activities Generated and Executed in the Exploratory Study

the students were asked first to express their understanding and thoughts regarding the issue at hand. After all of the students posted their reflections, a discussion took place based on these reflections. In this discussion, the students commented on their peers' statements and replied to comments made regarding their own statements.

Being both a working environment and a reflection tool that documents students' inputs, the forum was also very useful as a research tool. It provided additional data regarding the students' learning processes, contributing to data collection and triangulation.

5. DISCUSSION

The collaborative learning principles and the examples of respective activities and tool presented in this paper were aimed at supporting learning collaborative software development. It is important, however, to take into consideration that collaborative software development was not new to the students that learned this course. These students had two and a half years of collaboration experience, executing in pairs or teams their homework assignments related to different software development activities (programming, design, analysis, etc.). Thus, our aim was not to teach them collaborative software development as a new subject, but rather to develop and enhance their awareness as to how to improve this aspect of software development processes. As was illustrated in the previous section, this was done through active experience, in some cases simulating a real-life software development environment, applying different collaboration principles, as well as reflection and analysis activities.

The forum used for online discussions implemented the contributions of group decision-support systems presented by Alavi (1994), increasing the effectiveness of collaborative learning. First, it supported cooperation and teamwork among the students by facilitating information sharing and group processes; second, it facilitated evaluation and modification of student's mental models and awareness through exposure to alternative perspectives and increased and rapid feedback from group members.

Each of the activities, all of which were started in class and were followed by reflections in the forum, focused on a different aspect of collaborative software development. Based on the data analysis, we found that the students encountered many opportunities to construct firmer perceptions as to what successful collaborative software development is. In the discussions, both in-class and online, students tended to place great emphasize on the question: Which attributes contribute or damage collaboration? While perceptions of what these attributes are and how they affect collaboration varied among the students, it was apparent that the students indeed practiced reflection and analysis, becoming more aware and critical with respect to topics related to collaboration in software development processes.

Similar to Nunamaker et al.'s (1991) analysis of collaborative learning, our proposed framework for learning how to collaborate in SE situations also have gains and losses. Following are the predominant ones:

Gains:

G1. Students were motivated by the idea of simulating real-life situations they might encounter when working in industry.

G2. Throughout the course, a special atmosphere of openness and sharing developed, encouraging students to speak freely both about their in-class exercises as well as about different past experiences.

G3. Better assimilation of the subject studied (collaboration) was achieved through reflection, analysis and discussions.

G4. The fact that the activities invited students to look back on their previous collaboration experiences, presenting the opportunity to discuss, analyze and reflect on their real experiences, enabled them to develop a broad and multi-perspective understanding of the topic on the one hand, and to view and analyze past experiences from a new perspective, conceptualizing different impressions by the newly learned collaborative work concepts, on the other hand.

Losses:

L1. Simulations can only partly imitate real-life situations, leaving several aspects unattended.

L2. Since each student comes with his or her own past experience, their early perceptions might differ greatly, leading to misunderstandings or lack of focus in the discussions.

L3. Because the students come with previous collaboration experience, they are reluctant at times to replace their old habits with new behavior.

L4. Students at this stage of their studies differ from each other in their professional experience. Some already work in industry while others have not yet gained any professional experience beyond their academic tasks. In simulation situations, the latter tend to withdraw, letting the more experienced students take over. Similarly, students with lower academic achievements tend to give way to students with higher achievements. This phenomenon heightens when the more dominant students handle the discussion aggressively.

Table 2 specifies the gains and losses identified in our study for each collaborative learning activity or tool presented in this paper. In what follows, we describe how the gains and losses are attributed to the different activities.

Looking at the process gains, we find that gains (G3) and (G4) were relevant for all activities. Since all the activities concluded with reflection and discussion (G3), students could base their considerations and reasoning on their past experience and to analyze previously encountered

Tool/Task	Gains	Losses
Forum Discussion	(G2); (G3); (G4)	
Project Planning	(G1);(G3); (G4)	(L1);(L2); (L3); (L4)
Bonuses	(G1); (G2); (G3); (G4)	(L2); (L3)
Ethics	(G1); (G2); (G3); (G4)	(L2)
Teamwork Observation	(G2); (G3); (G4)	(L1); (L2)

Table 2: Gains and Losses Identified for Each Activity

phenomena in light of the new concepts they learned (G4). The first two gains were more specific: (G1) was relevant only in simulations and (G2) was relevant when open discussions were held, especially regarding the students' individual experiences.

Examining the process losses, we see that (L1) and (L3) were relevant only where simulations are concerned; (L2) and (L4) relate to different elements in students' past experience and early perceptions, and their influence needed to be examined for both simulations and class discussions. For example, in the case of the online forum discussion we found that (L2) was *not* present since the online discussion was always conducted after an in-class activity and/or discussion; hence, at this point common language and discussion focus have already been achieved. It is obvious that (L4) was not present either, since the first advantage of the online forum group discussion refer to the fact that dominant students are prevented from taking over the discussion (Alavi, 1994). Note that the fact that the forum did not suffer from any of these losses, does not mean that the online forum had no disadvantages whatsoever, but only that it lacked those losses specifically identified in the context of learning how to collaborate.

6. CONCLUSION

This paper suggests a framework for teaching collaborative software development. For this purpose, collaborative learning activities were designed and a tool was used based on features and attributes suggested by the literature for enhancing collaborative learning effectiveness. They were applied in an advanced university course in order to explore the learning processes and knowledge construction that take place when this teaching approach is applied.

We found that during the collaborative learning processes, students constructively develop a conceptual framework of collaborative software development. Each student - first individually, then in groups and finally in class forum - identifies relevant attributes and their desired values that might contribute to collaboration.

We also found that students' past experience influences both the gains and losses of collaborative learning activities (see Table 2). Accordingly, future work might examine the possibility of introducing collaborative software development at earlier stages of software development education, recruiting the students' first development experiences to learning successful collaboration.

We believe that educating software engineers about effective and fruitful collaborative software development may improve the efficiency of the software development industry and the quality of its products. Accordingly**,** another direction for future research is to apply the approach presented in this paper to the software industry, using it for augmenting collaboration in the work of software development teams. Such research could examine the initial perceptions of team members regarding different human aspects of collaboration in SE, explore how the approach and relevant tools affect them, and take further steps to enhance the practitioners' knowledge and skills in collaborative software development.

REFERENCES

Alavi, M. (1994) "Computer-Mediated Collaborative Learning: An Empirical Investigation", MIS Quarterly, Vol. 18, No. 2, pp. 159-174.

Bassey, M. (1999) Chapter 7: Methods of Enquiry and the Conduct of Case Study Research, in Case Study Research in Educational Settings, Buckingham, UK: Open University Press, pp. 65-91.

Beck, K. (2000) Extreme Programming Explained: Embrace Change, Addison-Wesley.

Bogdan, R. and Biklen S. K. (1992) Qualitative Research for Education: An Introduction to Theory and Methods (2nd edition), Toutledge, London**.**

Burton, M., Brna, P. and Pilkington, R. (1997) "Splitting the Collaborative Atom: How to Support Learning About Collaboration", in B. du Boulay and R. Mizguchi (Eds.), Artificial Intelligence in Education: Knowledge and Media in Learning Systems, Amsterdam, The Netherlands: IOS Press, pp. 135-142.

Brush, A., Bargeron, D., Grudin, J., Borning, A. and Gupta, A. (2002) "Supporting Interaction Outside of Class: Anchored Discussion vs. Discussion Boards", Proceeding of Computer Support for Collaborative Learning (CSCL) Conference, 2002.

Cockburn, A. (2001) Agile Software Development, Addison-Wesley.

DeMarco, T. and Lister, T. (1999). Peopleware: Productive Projects and Teams, Dorset House Publishing Company.

Guzdial, M., Ludovice, P., Realff, M., Morley, T., Carroll, K. and Ladak, A. (2001) "The Challenge of Collaborative Learning in Engineering and Math", Proccedings of IEEE/ASEE Fronitiers in Education (FIE) 2001 Conference, IEEE, Reno, NV.

Hazzan, O. (2002) "The Reflective Practitioner Perspective in Software Engineering Education", The Journal of Systems and Software, Vol. 63, No. 3, pp. 161-171.

Hazzan, O. (2003) "Computer Science Students’ Conception of the Relationship Between Reward (Grade) and Cooperation", Proceedings of the Eighth Annual Conference on Innovation and Technology in Computer Science Education (ITiCSE 2003), Hazzan, O. and Tomayko, J. (to appear). Tasks in software engineering education: "The case of a human aspects of software engineering course", Software Engineering: Effective Teaching and Learning Approaches and Practices (Ellis, H. J. C., Demurjian, S. A. and Naveda, J. F. – eds.), Idea Group.

Herring, R., and Rees M. (2001) "Internet-Based Collaborative Software Development Using Microsoft Tools", Proceedings of the 5th World Multiconference on Systemics, Cybernetics and Informatics. Orlando, Florida. July 2001: 22-25.

Highsmith, J. (2002) Agile Software developments Ecosystems, Addison-Wesley.

Humphrey, W. S. (2000) Introduction to the Team Software Process, SEI Series in Software Engineering.

Izquierdo, L, Damian, D., Singer, J., Kwan, I. (2007). "Awareness in the Wild: Why Communication Breakdowns Occur", ICGSE 2007.

Nunamaker, J.F., Dennis, A.R., Valacich, J.S., Vogel, D.R. and George, J.F. (1991) "Electronic Meeting Systems to

Support Group Work", Communications of the ACM, Vol. 34, No. 7, pp. 41-60.

Roschelle, J. and Teasley, S. (1995) "The Construction of Shared Knowledge in Collaborative Problem Solving", in O'Malley, C.E., (Eds.), Computer Supported Collaborative Learning. Berlin: Springer-Verlag, pp. 69-97.

Schön, D. A. (1983) The Reflective Practitioner, BasicBooks,

Schön, D.A. (1987) Educating the Reflective Practitioner: Towards a New Design for Teaching and Learning in The Profession, San Francisco: Jossey-Bass.

Sharp, H. and Robinson, H. (2007) "Collaboration and Co-ordination in mature eXtreme Programming teams", International Journal of Human-Computer Studies, doi:10.1016/j.ijhcs.2007.10.004.

Shuell, T.J. (1986) "Cognitive conceptions of learning", Review of Educational Research, Vol. 56, No. 4, pp. 411-436.

Strauss, A. and Corbin, J. (1990) Basics of Qualitative Research, Sage, Newbury Park.

Tomayko, J. and Hazzan, O. (2004) Human Aspects of Software Engineering, Charles River Media.

Tomayko, J. and Hazzan, O. (2005) "Teaching human aspects of Software Engineering", Proceedings of the 27th International Conference on Software Engineering (ICSE), St. Louis, Missouri, USA, pp. 647-648.

Venolia, G., DeLine, R., and LaToza, T. (2005). "Software Development at Microsoft Observed: It's about people... working together", Microsoft Research Technical Report, MSR-TR-2005-140. http://research.microsoft.com/research/pubs/view.aspx?type=TechnicalReport&id=994.

AUTHOR BIOGRAPHIES

Irit Hadar is a Lecturer at the department of Management Information Systems at the University of Haifa. Her research focuses on cognitive processes of software development; the role and influence of visual models in requirement engineering and software design; and human factors' influence on software quality. She has her Ph.D. from the Department of Education in Technology and Science of the Technion – Israel Institute of Technology. Her Master and Bachelor degrees are from the Faculty of Industrial Engineering and Management of the Technion.

Sofia Sherman is an M.Sc. student at the Department of Education in Technology and Science of the Technion – Israel Institute of Technology. In her research she studies student's perceptions of human aspects of software engineering. Her Bachelor degree is from the Department of Education in Technology and Science of the Technion.

Orit Hazzan is an Associate Professor at the Department of Education in Technology and Science of the Technion – Israel Institute of Technology. Her main research topics are human aspects of software engineering and computer science and software engineering education. In May 2004 she published her book *Human Aspects of Software Engineering*, co-authored with the late Jim Tomayko. Her second book – *Agile Software Engineering* – co-authored with Yael Dubinsky, will be published by Springer in Spring 2008.

APPENDIX A
HUMAN ASPECTS OF SOFTWARE ENGINNERING -- COURSE TOPICS

(Hazzan and Tomayko, to appear)

Lesson no.	Topic
Lesson 1	The Nature of Software Engineering
Lesson 2	Software Engineering Methods
Lesson 3	Working in Software Teams
Lesson 4	**Software as a Product**
Lesson 5	**Software Engineering Code of Ethics**
Lesson 6	**International and Cultural Perspectives on Software Engineering**
Lesson 7	**Different Perspectives on Software Engineering**
Lesson 8	**The History of Software Engineering**
Lesson 9	**Program Comprehension, Code Inspections, and Refactoring**
Lesson 10	**Learning Processes in Software Engineering**
Lesson 11	**Heuristics of Software Development**
Lesson 12	**Software as a Business**
Lesson 13	**Case Studies in Software Engineering**
Lesson 14	**Students’ Summary Projects and Presentations**

Table 3: The Course's Lessons

APPENDIX B
AN EXAMPLE OF THE ANALYSIS PROCESS

Phenomenon no.1:
In the project planning activity, the perception of the students with regard to the success of the mission was closely related to the degree of their active participation in the task.

Analysis process:

In this part of the research, we focused on research questions no. 3 and 4, which required tracing students' exhibition of collaboration and learning processes regarding collaborative software development.

At the first iteration of the project planning, material was collected via the online forum (including students' reflections and discussions). We found that the students used the words *success*, *successful* or *unsuccessful* many times. At the second iteration, we marked all the sentences that included these words, and focused on them in the third iteration of data analysis.

Content analysis of these sentences led us to define the following category for learning processes: *students are constantly looking for factors increasing/interfering with the success of the mission.* (This category emerged in students' responses to other activities as well).

In parallel, when analyzing the observations of teamwork and then a class work on the assignment, we found differences in the amount of contribution of each student. We defined another category *students' involvement* and divided the students to three levels we identified: (a) *high involvement*, (b) *low involvement*, and (c) *no involvement*. When looking at all categories emerging from the project planning activity, we noticed connections between the level of student involvement and their perception of the activity success. We then checked this new hypothesis by mapping for each student his/her perceptions of success to their level of involvement, finding that: the three students who acted as team leader in the team discussion (highly involved) expressed a general opinion that the discussion was quite successful; the four students categorized as less involved expressed many reservations regarding the way the discussion was held, indicating many flaws that they identified in the process, while none of them stated an explicit evaluation of the process outcome; the three students who contributed nothing to the discussion all stated that the discussion was a failure.

Table 4: An Example of the Analysis Process

iscap

Information Systems & Computing
Academic Professionals

EDSIG

Serving Information Systems Educators

STATEMENT OF PEER REVIEW INTEGRITY

All papers published in the Journal of Information Systems Education have undergone rigorous peer review. This includes an initial editor screening and double-blind refereeing by three or more expert referees.

ISSN 1055-3096