The key problems in team software projects

Jouni Ikonen, Antti Knutas, YongYi Wu, Isaac Agudo

Abstract: *Teamwork is an essential part of schoolwork. However, the experiences are not always positive for the members of the team, they face many problems that can frustrate a successful result. In this paper we concentrate on investigating problems in team projects, particularly those including programming tasks, and analyze the differences between successful and less successful projects (as part of e-learning). We have defined and distributed a questionnaire among different European countries. The main result is a list of the most common problems, of which the most common turned out to be communication problems.*

Key words: *questionnaire, collaborative work, programming, software, teamwork*

INTRODUCTION

Collaborative learning, or students working cooperatively to reach a shared learning goal [2], is one of the common teaching methods used for teaching programming in computer science education [1]. It has been shown to increase student motivation [9], critical thinking [4] and to teach teamwork skills needed in the industry [1]. Team projects (or group projects) have many pitfalls [6, 12]. Teams that look like dream teams, composed of skilled people, have fallen miserably due to a variety of problems. Many of the problems may be because of to the lack of training on teamwork. Sangrà, et. al [10] came to general conclusion that e-learning is part of the new dynamic that characterizes educational systems at the start of the 21st century, resulting from the merge of different disciplines, such as computer science, communication technology, and pedagogy. These take into consideration the four main categories in which authors conceptualize e-learning are technology, delivery systems, communication, and educational paradigms. Our initial objective in this work is to look e-learning from the perspective of distributed project work applying technological solutions for collaborative learning.

Our objective in this paper is to investigate what are the most common problems in programming related team projects. For this purpose we made a questionnaire to explore experiences on programming team projects and especially in student team projects. We generally assume that there are differences between student and professional software projects [5]. Based on our previous research our assumptions of the differences are:

- Learning is the stated main objective in student projects.
- There is no legacy of how software project "has always been done" in student projects.
- The participants of a student project are more or less voluntarily participating to the project, which is likely to limit the tools that project manager can use to coerce the workers.
- Students might have different goals, as for someone the project might be extra credit and for others it might be way to produce software demo to get a job.
- Also, the personality of every student is different, which influences their performance and their relationships with their peers.

Our main research questions are:

- What kinds of problems are present in programming team projects?
- What kinds of differences are between successful (best) and less successful (worst) projects?

The paper is organized in two main parts. The first part introduces our research methodology and research setup. The second part presents some of our findings as results. The paper is finished with conclusion, which aims to help the reader to identify our main findings.

RESEARCH METHODOLOGY

Apart from investigating the problems in team projects, we also wanted to identify the differences between those projects considered successful by the members of the team and those where something went wrong. We asked participant to think about the project which they felt performed best but also about their worst experience. For this purpose we developed a questionnaire to have deeper understanding how project participants feel and what issues they encountered.

Research setup

The initial questionnaire was formulated by a group of three people (consisting of a doctor of IT, a doctoral student in computer aided collaboration and a computer science master thesis worker). During the formulation it was revised multiple times according to feedback from user tests. User tests were performed in three stages in a Nordic country and a South European country to avoid cultural bias.

The questionnaire was then distributed via participating researchers' personal connections, social media and via mailing list of a European project [3] with participants from more than 30 countries. Questionnaire was open for 2 weeks in the end of May 2015. Respondents were offered a possibility to enter a raffle for a chocolate bar in exchange of using 5 to 15 minutes of their time for our research.

The questionnaire

The questionnaire asked some basic information about the users, such as whether they have participated in projects that involve programming, how long they have been programming, country and age group. We also asked about general attitude towards group projects based on prior experiences, what kind of problems they have experienced (based on the list of problems provided by Pournaghshband [8], with an additional "other, please explain" category), and how many group projects they have participated in.

Users were asked to think about their best and worst group projects. If they had participated only in one project, they were asked to answer only on the best or the worst project category according how they felt the project went. For *the best project* and *the worst project* the following questions were asked:

- How many persons the project had?
- How long the project was?
- Did you know what you were expected as member of group? (Likert scale)
- Was the workload divided equally? (Likert scale)
- Did you contribute more than your fair share to the project? (Likert scale)
- How much group members worked together? (Likert scale)
- How frequent the group interaction was? (Likert scale)
- How often group members shared information about their progress? (Likert scale)
- Did you have any way to see other group members' progress?

After this we asked their opinion about the following issues: How often group progress should be tracked, and how project process has been tracked in previous projects. We also asked them if they would like to see how each person contributes to the project progress and whether they have used version control software or distributed version control software in their projects.

RESEARCH RESULTS

Audience Demographics

The questionnaire was answered by 81 persons from 22 different countries, based on the provided country information, which was a voluntary field. Most of the respondents were young adults at age group of 19-22 years (32.5%), none of the respondents were under 18 years and 26.25% of respondents were 30 or older. About 65.38% of respondents were male. Questionnaire concentrates on programming group work, so we asked how many years they have being programming. Five persons answered that they have no programming experience, 12 persons identified to have less than one year of experience, 26 had from two to five years, and 38 more than five years of experience. We also asked about educational background on programming and 21 identified having professional skills, with more than 2 years of experience on the field, 34 had a Computer Science (CS) or related degree with less than 2 years of experience on the field. Additionally, there were 16 persons that took some programming courses, 8 hobbyists and 2 without programming skills. Given that the questionnaire was circulated on academic mailing lists and social media groups, we have to assume that many of the persons answering the questionnaire are students of involved with academic institutions.

Most common problems

Our warm up question was "Based on your previous experiences on group projects, how do you feel about your group projects in general?" and asked to rate their experience on a 5 step likert scale, where one end is *dislike* and other end is *enjoy*. The purpose of this question was to focus the respondent's attention on those projects they have been involved and identify general attitudes. The average response was 3.8 (σ = 0.94), which indicated that respondents are generally quite happy with their group project experiences.

We wanted to identify what kinds of problems our respondents had in their teamwork experiences. In the formulation of this question we used the list of problems provided by [8] and added the "other, please specify". While different problems were identified by most of the respondents, it turned out that 15 respondents didn´t have any problem. Table 1 lists the problems. Clearly the most common problem was "poor communication among members", with 39 occurrences. Other common problems were procrastination (delaying the problem), conflict in personal schedule, poor leadership and lack of cooperation.

Table 1. Common problems identified in programming team projects

Problem	# of occurrences
Poor communications among members	39
Procrastination problem (delaying or postponing problem)	29
Conflicts in students' schedule	25
Lack of cooperation	23
Poor leadership	23
Members' personal problems	19
Integration testing problem	17
Lack of confidence (to begin the project)	16
Haven't experienced problems	15
Failure to compromise (couldn't meet each other halfway)	14
Other problems	6

Poor communication and procrastination are problems that can be due to bad team culture and which an experienced project manager can ease. There are also tools and practices to help overcome these problems, however tools are useless if some members are not committed to the project. Student assignments due dates are often at the end of the course, which might be months away. It is then easy to postpone starting project or personal responsibilities in the project. Teamwork requires number of different things to take care of, e.g. scheduling, people management and quality management, for which there exist a number of different tools [7, 11]. Software projects have their own specialties, such as source code management, and there exist a vast number of applications supporting software projects [1].

Project management tools could facilitate automatic communication - up to a degree. In particular, they can show everybody's personal contributions, e.g. how much they have contributed to programming assignment in forms of code, comments, tests or documentation during last 24 hours. We have noticed in many cases that being aware of fellow students progress might encourage other persons to start working too.

Differences between the best and the worst projects

We also investigated on how team members feel about workload division. Respondents' workload is much less fairly divided in their worst projects, giving it an average grade of 2.6 (5 marking *fairly divided*), while in best project they gave average of 3.45. We are also interested about how team members feel about their personal contribution compared to others (Table 2). The best and worst projects gave similar average results (3.45 vs. 3.42), which means that in average people consider contributing more than their share to the project. When we look at the answers distribution (Table 2) we see that there is much more spread in worst project. This indicates that some respondents felt that they did extremely a lot of work and some did hardly any.

Table 2. Personal opinion on "Did you contribute more than your own share of the work?" compared in the best and the worst project

	1 **(Much less)**	2	3	4	5 **(Extremely much more)**	Answers	Average
Best project	2.67%	5.33%	41.33%	45.33%	5.33%	75	3.45
Worst project	8.00%	12.00%	28.00%	25.33%	21.33%	71	3.42

Role of team members is not always clear. We asked about understanding personal role as group member. In best projects participants reported that they had clear idea (vs. no idea of role) with average of 4.18 as in worst projects this was 3.45.

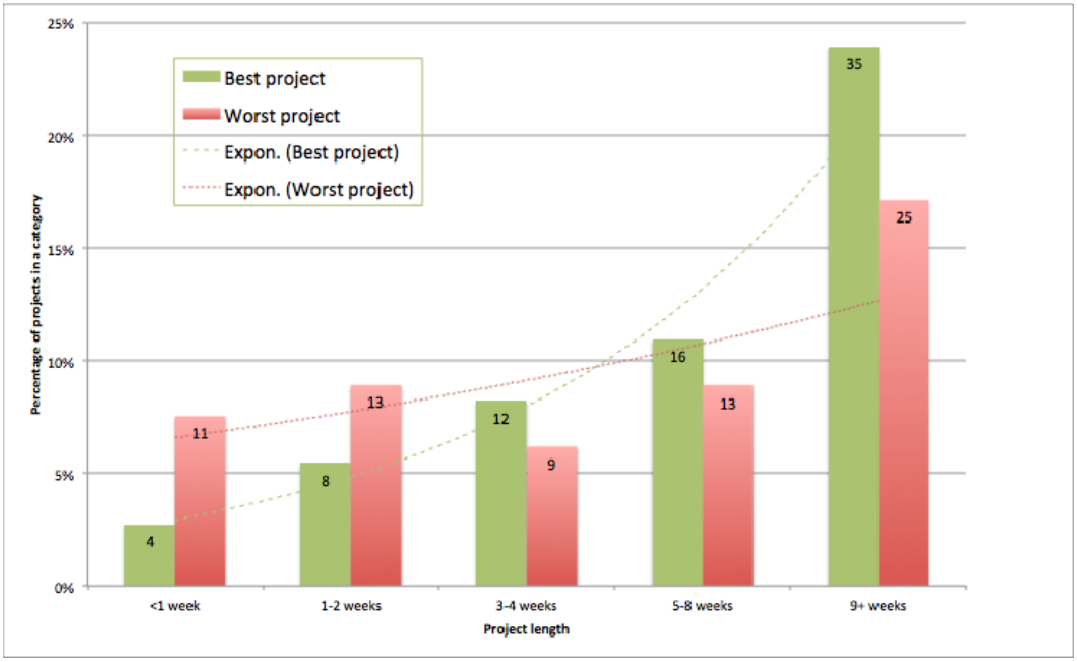

Figure 1. Comparison of project length for best and worst project experiences

We analyzed if the project length would correlate to best and worst project in some way. This is illustrated in Figure 1, where we can see that there are many more short projects (of all projects) classified as worst projects than in long projects. Bar length in the figure presents the percentage of the selected category of all the answers. Numbers marked in bars are absolute numbers of the answers in the category. As projects are getting longer, relative number of best projects is much higher than worst projects and trend lines in the figure illustrate this. The query does not reveal the reason for this, but we can theorize that less experienced persons do shorter projects and/or people are not willing to use project management tools for short projects.

CONCLUSION AND FUTURE WORK

Objective of this work was to establish what are the most common problems in team projects, which include programming. The presented questionnaire indicated that poor communication between team members is the most common problem in the respondent's projects. We also identified some aspects where best and worst projects differed from each other. In our results worst projects tend to be shorter, participants feel that work is not divided equally and members' understanding of their own role is unclear.

The work presented will be used as a base research to identify what kind of issues we should issue in our teaching and if there are need for additional tools. As the future work we will continue analyzing the results together with investigating how visualizing the contributions of each team member can affect a project. This could be achieved with some kind of project dashboard to give general view for team members about everyone's ongoing work, contributions and current issues. It would also be interesting to see if motivational techniques (mainly gamification) could be applied to distributed version control repositories in order to motivate a more active participation.

REFERENCES

[1] Coccoli, M., Stanganelli, L. and Maresca, P. 2011. Computer Supported Collaborative Learning in software engineering. *2011 IEEE Global Engineering Education Conference (EDUCON)* (2011), 990–995.

[2] Dillenbourg, P. 1999. What do you mean by collaborative learning? *Collaborative-learning: Cognitive and computational approaches.* (1999), 1–19.

[3] Fetch - Future Education and Training in Computing: How to support learning at anytime anywhere, EU project 539461-LLP-1-2013-1-BG-ERASMUS-ENW, http://fetch.ecs.uni-ruse.bg/ (visited 28.6.2015)

[4] Gokhale, A.A. 1995. Collaborative Learning Enhances Critical Thinking. *Journal of Technology Education*. 7, 1 (1995).

[5] Kasurinen, J., Mirzaeifar, S. and Nikula, U. 2013. Computer Science Students Making Games: A Study on Skill Gaps and Requirement. *Proceedings of the 13th Koli Calling International Conference on Computing Education Research* (New York, NY, USA, 2013), 33–41.

[6] Knutas, A., Ikonen, J. and Porras, J. 2013. Communication Patterns in Collaborative Software Engineering Courses: A Case for Computer-supported Collaboration. *Proceedings of the 13th Koli Calling International Conference on Computing Education Research* (2013), 169–177.

[7] Mäkiaho, P., Poranen, T. and Seppi, A. 2014. Version Control Usage in Students' Software Development Projects. *Proceedings of the 15th International Conference on Computer Systems and Technologies* (2014), 452–459.

[8] Pournaghshband, H. 1990. The Students' Problems in Courses with Team Projects. *Proceedings of the Twenty-first SIGCSE Technical Symposium on Computer Science Education* (New York, NY, USA, 1990), 44–47.

[9] Resta, P. and Laferrière, T. 2007. Technology in support of collaborative learning. *Educational Psychology Review*. 19, 1 (2007), 65–83.

[10] Sangrà, A., Vlachopoulos, D. and Cabrera, N. 2012. Building an inclusive definition of e-learning: An approach to the conceptual framework. *The International Review of Research in Open and Distributed Learning*. 13, 2 (Feb. 2012), 145–159.

[11] Serçe, F.C., Swigger, K., Alpaslan, F.N., Brazile, R., Dafoulas, G. and Lopez, V. 2011. Online collaboration: Collaborative behavior patterns and factors affecting globally distributed team performance. *Computers in Human Behavior*. 27, 1 (Jan. 2011), 490–503.

[12] Williams, S. and Roberts, T.S. 2002. Computer-supported collaborative learning: strengths and weaknesses. *Computers in Education, 2002. Proceedings. International Conference on* (2002), 328–331.

ABOUT THE AUTHORS

Associate Professor Jouni Ikonen, D.Sc., Innovation and Software, Lappeenranta University of Technology, Phone: +358 294 462 111, E-mail: jouni.ikonen@lut.fi

Antti Knutas, M.Sc., Lappeenranta University of Technology, Phone: +358 294 462 111, E-mail: antti.knutas@lut.fi

YongYi Wu, MSc. student, Innovation and Software, Lappeenranta University of Technology, Phone: +358 294 462 111, E-mail: yongyi.wu@lut.fi

Associate Professor Isaac Agudo, Computer Science Department, University of Malaga, Phone +34952136315, E-mail: isaac@lcc.uma.es

The paper has been reviewed.