Success Factors in Making your Capstone in Software Engineering Productive and Appealing

Carsten Kleiner[1], Arne Koschel[1]
[1]*University of Applied Sciences & Arts Hannover, Germany*

The capstone project in software engineering is a quite important class for the students in order to enter a successful professional career. Therefore it is essential to choose an interesting topic for such a project, both from the student's as well as the future employer's point of view. In addition capstone projects may be an important foundation for future research projects for the instructor. In the last couple of years we have experimented with several different types of such projects with different levels of success achieved. In this paper we try to summarize our experiences by deriving important influential parameters that have a high potential of leading to a successful and appealing project for students. While some of the parameters seem reasonable as a general success factor, others may also be specific to a certain project or team. We feel we have a sufficient base of experience to differentiate those types, even though results are not statistically significant due to small numbers. We will also evaluate how the different parameters act together and derive some interdependencies.

Corresponding Author: Carsten Kleiner, carsten.kleiner@fh-hannover.de

Introduction

The main goal of a capstone software engineering project in an undergraduate CS program is to prepare students well for the beginning of their professional career after graduation. Another important goal is to offer an interesting project for the students in order to increase their motivation. Also a capstone might be an important foundation for future research projects and publications by the instructors. To meet these goals one is faced with the challenge of choosing from several different parameters to set up a capstone.

Therefore we tried to determine relevant success factors for capstones in software engineering. We did that by reflecting our projects offered within the last 5 years with about 70 participating students overall. Basis for the evaluation are student questionnaires (two per student per project), feedback from internal and external project partners as well as personal perception of the project results and mechanics. Note that the influential parameters described later in this paper have been derived from the evaluation. There has not been an up-front analysis leading to specific project offerings. This will change in the future based on the conclusions presented in this paper.

Setting for our Projects

All projects that have been used for the evaluation in this paper have been offered in the final year of our bachelor's program in *Applied Computer Science*. The curriculum contains a one-year project, which is typically a software development project. Project team size varies between 6 and 14 students. They are expected to spend about 1 day per week on this project on average. Students may pick from several different capstones offerings in parallel, so an appealing topic is important in order to attract the better students.

In Germany all bachelor programs are subject to accreditation by one of two national agencies. They expect schools to produce well employable graduates. Thus a fair amount of the curriculum has to be devoted to *applied* classes. Thus we try to offer capstones in close cooperation with external partners from industry or non-profit organizations whenever possible.

Such cooperation is a little difficult since the outcome of a student project is almost non-predictable. Whereas a very motivated student team may be able to produce a really good and helpful piece of software, a poorly motivated and/or capable team may not be able to produce a usable result. Nevertheless our experiences with external projects are pretty good: students seem to be better motivated when producing software for external partners. On the other hand external partners are very satisfied with the results. It is important to let them know right from the start that there is no guarantee of a usable product at the end and no support is offered.

From a diversity point of view we typically have much more homogeneous groups of students than in other countries. Still there are two important issues:

- The number of female students has been steadily decreasing over the last couple of years and is well below 10% right now. In order to reverse this trend appropriate measure must be applied, particularly in the capstones.

- There are a fair number of students originating from parts of the former Soviet Union who have German ancestors and migrated to Germany. Their number is particularly large in the *Universities of Applied Sciences*. They seem to prefer the applied approach as opposed to the more theoretically oriented classical universities. They tend to operate in fixed ethnic groups within the student body based on their mother language. This situation poses significant challenges on the organization of a project team as cultural subgroups are formed which are difficult to break up.

We think that the aforementioned basic setting for our capstones is very similar to many other institutions. Therefore the conclusions in the following sections should be easily transferable.

After a brief review of related publications we will present the most important influential parameters for capstone success from our evaluations. A detailed and structured analysis of our projects against the influential parameters remains is omitted due to space constraints. We will finally summarize the experiences and present our conclusions for future capstone offerings.

Related Work

There have been many interesting publications describing experiences with capstone projects in software engineering which cannot all be referenced due to space constraints. We will just discuss a few which are based on particularly large numbers of examples or have a more conceptual approach.

The great importance of an external customer to provide a real-world framework for the project has been proven by several colleagues already[1,4,5]. Also some papers discuss important success factors, but more from a conceptual view[3] or differently structured[6,7] than in this paper. Finally the issues with diversity have already been raised[8], and an extensive experience based discussion of process models[2] may be found that evaluate certain of the influential parameters in detail.

Influential Parameters

When trying to determine the success factors for a capstone in software engineering one has to look at different categories. Among the parameters which can be influenced by the instructor are technological details of the task offered as well as the organizational setting. Therefore we will look into the details of these categories at first. But it is important to note that there are other parameters which are largely beyond the instructor's power. Even those may be interesting to look at in detail because identifying these factors early might be helpful to choose appropriate countermeasures. We will call such parameters social factors in this paper.

We will mainly focus on parameters that are specific to capstone projects in a university setting. We will not look at factors that impact any professional software development project as well. Those have already been investigated by industry and research and led to the modern software engineering models which are beyond the scope of this paper.

Technological parameters

(T1) Legacy vs. cutting-edge technology: the latter typically seems more appealing to students in the first place. Nevertheless after graduation they will often be confronted with legacy systems rather than new technologies. Therefore the former may be wiser from the perspective of employability. Also students will get a better insight into their future workplace and more opportunities for industry cooperation are there when using legacy.

(T2) Well-taught vs. new additional technology: students typically feel more comfortable when a technology is used that they already know well from previous classes (e.g. Java as programming language at our institution). On the other hand a capstone offers a perfect setting to start with a completely different technology (such as C# at our institution). There is sufficient time for a thorough learning of something new in case the overall problem is downsized accordingly. Thus it may be one of the best ways to learn something new while using it to produce something meaningful right away.

(T3) New standalone application vs. adding a component to an existing system: On one hand the idea of adding a component to an existing system seems more appealing for a student project as students can start with something already running and *just* extend it (e.g. choose some open source software and add functionality to it). There is already a given project structure and code organization that may used. The downside to this approach is the potentially large amount of work to be spent before the existing code is understood to a degree where extensions are possible. This may also lead to significant loss in student motivation if the existing software is not well-documented or does not function properly (which is frequent in open-source software). So the benefit of the existing setup may be used up soon.

(T4) Technological challenges vs. application-specific challenges: to keep a project manageable the major challenges should be from either the technological area or from the application domain. If it is challenging in both aspects you will either need a very talented project team or a team that provides experts for both aspects. This is probably the factor where student preferences vary the most. We see a fair amount of students who prefer the computer science

specific technological challenges while there are also a number of students who prefer to apply well-known technology to an unknown domain. Since there are both types of job profiles for CS graduates, either choice may be good depending on your team. If you have the luxury of having both types of students in your team then you can really solve big problems.

Organizational parameters

(O1) Traditional project organization vs. agile development: we have gathered experiences with more traditional types of project organization (such as requirements specifications and waterfall model) as well as with agile methods (extreme programming, Scrum). The students typically favor modern approaches. Unfortunately natural restrictions of projects at universities (e.g. time and space constraints, distributed working) make it necessary to adjust these methods accordingly. Such adjustments are possible but also restrict the use of the agile models to some extent.

(O2) Project management by either instructor, mentors from a master's class or members of the team itself: There are different people who can serve as project managers. Apart from the natural choices of instructor and students from the project team we have also experimented with picking master students from a project and quality management class. Experiences are very promising, but unfortunately this is only a one semester class, so that project management has to be changed midway through the project. Adjusting durations of both classes would be very beneficial. Also such master students served very well as quality managers because their opinion and advice was very well absorbed by the project team.

(O3) Distributed development vs. dedicated project lab: students increasingly prefer to work at their homes instead of labs on campus. Problems incurred by this behavior are very closely related to problems with distributed software development in a professional setting. We feel that this issue deserves special attention in a university context, because we think it is extremely important to assign a dedicated project lab. The team should get the chance to make this room as cozy as possible, thus students are more likely to work together in the lab. This is particularly important in initial and final phases of a project, but is beneficial throughout.

(O4) External cooperation vs. internal topic: The origin of the topic for the capstone is of huge importance. Our evaluation includes both: topics raised by cooperating companies and organizations as well as more research oriented internal topics. In general we observed a great student motivation when they worked on externally assigned tasks. But there are also problems: we found that it is almost impossible to solve an interesting external problem in a capstone project and at the same time get production-quality software. This is due to the time constraints of a capstone: approximately 8-10 students working 9 months for 1 day a week resulting in 320 to 450 person days total. Given that this includes time for initial preparation, setting up the project infrastructure and learning the details about the topic, it is very difficult. Thus choosing an interesting external topic should be tied to letting the partner know what not to expect out of the project. With these restrictions industry cooperation is advised. Nevertheless, we also had internal topics that have been liked a lot by students. This is particularly true for very challenging previously unknown cutting-edge technology that had to be used.

Social parameters

(S1) Involvement of female students: since the number of female students is very small we have not found any reliable idea on how to attract them to a particular project. We found that female students tend to form fixed small groups and choose a capstone subject mostly based on the potential project team rather than the content. Sometimes a practical value of the outcome of the project with respect to an application domain seems to be important. Pure technologically challenging projects without immediate practical value seem to be of lesser interest to them.

(S2) Involvement of culturally different subgroups: as discussed in the introduction we deal with a fair amount of students with culturally different background. These tend to form their own groups and act in small subgroups within the project team. Typically these students are technologically well skilled and thus may greatly contribute to project success. But they need more detailed guidance and very specifically formulated tasks to work on in order to achieve good results. Also they typically prefer to work technically and do not like to do application-domain oriented or conceptual work as much. Maximum benefit might be reached if a flexible internal project organization is used and these culturally different students are teamed with other students. In the beginning this will slow down the project due to friction loss. But over the full course of a project it will likely pay off, so overcoming initial resistance is advised.

(S3) Technological competence of students in project team: this is a parameter that is almost impossible to influence. Students may choose their capstone projects from the ones offered. We discovered that typically the more competent students form clusters choosing the same project as do the less competent ones. There is likely no way of choosing a capstone project that the less competent team will master as well as the more competent one. A difference in individual productivity up to factor 10 is reality in our experiences. It may nevertheless be helpful to estimate average competency

of your team in advance. Thus there is a chance to adjust the details of the project accordingly. This might produce a task perfectly suited for the specific skills of your project team.

(S4) Motivation of students in project team: potentially the most important success factor in a capstone project besides the participant's skills is their motivation. Part of it seems to be an initial basic motivation that is present or not; this can be influenced only to a certain degree by any of the other success factors by the instructor. Nevertheless there is also a significant part of motivation that may be influenced. This is similar to a professional setting with the difference that grades have to replace money and career opportunities. It is a great motivation boost if an external partner closely interacts with the project team and expresses interest in the project results. For internal topics the plan to write a research publication on the project results leads to some students spending extra effort. Moreover project management by fellow students or master students seems to increase motivation as opposed to the instructor doing the project management. However, while this might reduce some effort for the instructor, it needs the instructors will 'to let the project go' regardless of maybe less perfect results.

Evaluation and Conclusion

A successful project with external cooperation (O4) has shown that factor S4 can be significantly improved. This might even remedy missing technological competence (S3) in the team, at least partially. Nevertheless a small part of the team felt discouraged by the external problem due to excessive demands.

On the other hand even internal subjects (O4) led to very good student motivation (S4) in another case. This may have been because of a particularly well-skilled project team in this case (S3). So in summary it seems that especially for weaker project teams a practical application is important whereas stronger teams may be equally successful and motivated on internal tasks. This is also important because internal tasks may be used as foundation for research projects and thus attract well-skilled students to the master's program.

Regarding O1 we discovered that the less skilled the project team is, the more successful the project becomes with a more traditional process model. Agile methods seem to be particularly successful with well-skilled students. Obviously they make the most out of the more open character of such processes. In our experiences neither T1 nor T3 determine general rules for project success. But it seems that well-known technology is favored by weakly-skilled project teams, because they feel safer and are less likely to be overstrained.

Similarly attractiveness to women (S1) can be significantly improved by using known technologies and putting an emphasis on application-specific challenges as opposed to technological ones (T4). Parameter S2 is of great importance as has been shown by a less successful project where involvement could not be achieved by mixing students in sub teams because almost the whole project team consisted of such a culturally different group.

So far we have tried to leave it to the students to find their preferred mix of on-site and off-site working, but we provided a designated project lab. In the future the demand for distributed working (O3) is increasing, but our impression is that it lessens motivation and is particularly bad for agile development. Aligning these expectations will be a major challenge. Also we would like assign project management to master students (O2) because they seem well respected by the participants and lead to increased manpower for the project itself.

Topics are perfectly suited for capstones in software engineering if many of the choices of the influential parameters may be made after project team members and their background is known. This is due to the interdependencies between social parameters and the other parameter groups. An appropriate choice of O and T parameters according to the specific team are most likely to produce an appealing and successful project.

References

1. Ville Isomöttönen, Tommi Kärkkäinen: The Value of a Real Customer in a Capstone Project. Proc. of 21st Conference on Software Engineering Education and Training, p. 85-92, IEEE Press, 2008.
2. David Umphress, Dean Hendrix, J. Cross: Software Process in the Classroom: The Capstone Project Experience. IEEE Software, 09/10-2002, p. 78-85.
3. Janet E. Burge, Gerald C. Gannod: Dimensions for Categorizing Capstone Projects. Proc. of 22nd Conference on Software Engineering Education and Training, p. 166-173, IEEE Press, 2009.
4. S. Karunasekera, K. Bedse: Preparing Software Engineering Graduates for an Industry Career. Proc. of 20th Conference on Software Engineering Education and Training, p. 97-106, IEEE, 2007.
5. Bruce R. Maxim, Kiumi Akingbehin: Experiences in Teaching Senior Design Using Real-World Clients. Proc. of 36th ASEE/IEEE Frontiers in Education Conference, San Diego, 2006.
6. A.T. Chamillard, K.A. Braun: The software engineering capstone: structure and tradeoffs. In Proc. of SIGCSE '02. ACM, New York, 227-231.
7. R.E. Beasley: Conducting a successful senior capstone course in computing. Journal for Computing Small Coll. 19, 1 (Oct. 2003), 122-131.
8. T.E. Carpenter, A. Dingle, D. Joslin: Ensuring capstone project success for a diverse student body. Journal Comp. Small Colleges 20, 2 (2004), 86-93.