# Ownership of Artifacts and Intellectual Property for Software-Intensive Capstone Design Projects

James Vallino
*Rochester Institute of Technology*

The undergraduate software engineering program at the Rochester Institute of Technology has had a capstone project in its curriculum since its inception in 1996. These software-intensive projects, typically composed entirely of software elements, bring a heightened need to have solutions for the issue of ownership of artifacts and intellectual property generated during the project. This comes about primarily because on a software-intensive project, it is easier to create artifacts that represent notable value to the project sponsor and have embodiments of intellectual property that clearly have been reduced to practice. The approach used to handle ownership of project artifacts and intellectual property created within the context of capstone design projects is an important consideration that runs through many aspects of the project from solicitation of project proposals through to expectations on deliverables from the project team. The question of artifact ownership is one of the first ones that potential project sponsors ask. Our approach defines four project types that vary with regard to assignment of ownership of project artifacts and intellectual property. The paper also discusses adjustments made in our institution's policies to allow project agreements that streamlined the process of starting new projects.

Keywords: software-intensive capstone projects, intellectual property, capstone project agreements

*Corresponding Author: James Vallino, J.Vallino@se.rit.edu*

## Introduction

The Rochester Institute of Technology (RIT) started the first undergraduate degree program in software engineering[1,2] in 1996. The program is the largest in the United States with over 400 undergraduate students currently enrolled. The curriculum is distinct from a traditional computer science program in many respects. Instead of emphasizing the boundaries of computing technology areas, our software engineering program[3] concentrates on four elements of engineering practice: engineering design, software product development, teamwork, and communications. A capstone design project, which we call senior project, was in the program since its inception.

Our original design intent for the program was that the software engineering content would be divided between courses emphasizing the principles that undergird the design of software, and the practices that a software development group needs to follow to deliver a quality software product at the end of a project. While there are some principles that are specific to particular domains, the design principles are mostly applicable to any domain in which software is developed, including web-based systems, desktop software, embedded safety-critical systems, or mobile applications. The product development practices include development methodology, process and product measurement for software quality assurance, and project management.

From the start, key characteristics of our program included heavy emphases on teamwork and communication skills. Software engineering programs that followed (there are now 48 programs) have tended to follow that lead. With respect to teaming, the typical software engineering course in our program will have 3 team-based projects through the term with 40 to 50% of the student's final grade based on team project activity. Each team will be 3 or 4 students demonstrating their grasp of the course concepts through their team activities. Each of these course projects lasts three to five weeks. The first team-oriented software engineering course, Introduction to Software Engineering, has a term long project, and senior project is one project for an external sponsor with 4 or 5 students working for an entire academic year to deliver a software solution for the sponsor's problem.

### Software-Intensive Capstone Projects

Software-intensive capstone projects done as part of a software engineering program can have a different perspective on the course itself and the expectations for what a student team can accomplish. There are a number of aspects to this. First, because the students have already been more heavily involved with team activities, the capstone design project will not be the students' introduction to significant team project activity. A student in our program takes a minimum of nine software engineering courses prior to entering senior project, which provides the student with an experience base of working on more than 20 team projects. This allows us to have an expectation that the teams will not need to be taught about teaming at this point. While teams will have the normal team formation issues, the students have seen many of these issues before, and we can expect fewer problems getting a project started.

Earlier in their program, the students have studied elements of project management and product development. This means that there will not be a need to spend time during senior project to discuss the need to elicit customer requirements, the steps for project initiation, metrics to measure and track the progress of both project and product, or any aspect of the generation of appropriate engineering project documentation.

The final element that affects the expectations placed on a team working on a software-intensive project is the nature of software projects themselves. It is a lot easier to deliver artifacts that represent notable value to the project sponsor and have embodiments of intellectual property that clearly have been reduced to practice.

With these characteristics of our program and what curriculum content needs to be in senior project, we designed our senior project courses different than many other capstone design courses. Our senior project courses do not have any lecture or curricular material that is covered during class time. The class periods are used by the teams for meeting with their faculty coach and project sponsor, or as guaranteed team meeting times. This allows us to have higher expectations on a team in senior project. We expect that a senior project team will deliver a software product taken at least to the working prototype stage, if not all the way to a fully-functional product.

## Project Artifact Ownership Considerations

With the expectation that the team will deliver a working system at the end of the project, project sponsors are very interested in knowing if they can use the artifacts developed by the team. For the remainder of this paper, I will use the term project artifact to cover anything that a team produces as part of their senior project work. Where the phrase intellectual property usually refers to patentable ideas, the ownership and use concerns here are much broader. The reality of our senior projects is that very few generate any new intellectual property, but with a working prototype expected, though not guaranteed, at the end of the project, there is notable value in the artifacts produced.

In anecdotal conversations with others connected with capstone design projects, I have often heard it mentioned that this is not a concern for their projects because, "All the sponsor gets is a report." If the project sponsor wanted to have the report created by a consulting team, the sponsor would have had to pay for that work which indicates that there is value in the information in that report. Additionally, it is not uncommon for capstone design projects to be proposed by faculty members looking for a student team to perform work a one of their research projects. I have never seen one of these projects where the faculty member did not have an expectation of full use of all project artifacts after the project is completed. This is ample justification for the need to make an explicit statement regarding the ownership of project artifacts. A discussion of ownership, copyright, and intellectual property rights can be made part of course content. Of paramount importance in all of this is protecting the students' rights.

## Institutional Policy

For many years, we ran senior projects in a grey area without directly addressing issues of ownership of project artifacts. To enable us to attract project sponsors, we wanted a streamlined proposal process that included the minimum number of approval steps for individual projects. We also wanted the sponsors to have use of the project artifacts that a team generated. As our program grew and senior project built a positive reputation with our project sponsors, it became clear that our policy of not addressing ownership issues was not a sustainable policy. Too many sponsors were asking questions where a wink-wink, handshake answer was not sufficient. After a few protracted rounds of negotiation over non-disclosure agreements, it also became clear that we needed to streamline how we addressed the disclosure of proprietary information. This is not a direct topic of this paper, but is mentioned in our later discussion of project agreements.

As with many administrative matters, the place we started investigating how we would handle project artifacts was with our institution's policies. Most institutions will have a policy on intellectual property. These can range from the student owns everything to the institution owning everything[4]. When we first undertook getting clarification on project artifact ownership, RIT was updating its Intellectual Property Policy[5]. The policy current at the time stated that students owned anything they created as part of their coursework, which would include all artifacts created in senior project courses. If we wanted to adhere to RIT policy, and allow project sponsors to use the project artifacts, we clearly had a need for an explicit assignment of the students' rights to the project sponsor.

There was a lengthy discussion at the Deans' level about requiring a student to assign rights to his or her work to a project sponsor. Making this a requirement was not in the interest of protecting the students' rights, but the policy as written, did not allow for the assignment to be requested either. We raised two points when arguing for a change, or addition, to the policy:

- Project sponsors would be much less willing to spend the time working with a capstone design team, if they would have no rights to the project artifacts at the end of the projects.
- The students would suffer because they would lose the experience of working with a completely new sponsor who may not be very technical, and the projects created by faculty who are mostly not active in commercial level software development would not be as interesting or challenging.

The end result was that the following language was added to the new Intellectual Property Policy so that students could be asked to assign their rights to course project work to a project sponsor:

> *Students may be requested to grant rights or ownership in Student-developed IP to RIT or others as a condition of access to certain class projects, independent research projects, or other programs at RIT. Students who choose not to grant rights or ownership in Student-developed IP shall not participate in these class projects, independent research projects, or other programs, but shall be given the opportunity to participate, without penalty, in alternative projects which do not require the granting of rights or ownership in Student-developed IP.*

## Project Types

The new policy statement allowed us to request the assignment of project artifact ownership to the project sponsor. The next step was creating project agreements that would cover the various types of projects that we envisioned were possible.

Our experience with software engineering capstone design teams is that production of a software artifact that is close to production quality is a distinct possibility. We have had projects that went live during the project period, or were incorporated into sponsor products with just a few months of additional engineering effort. The open community eco-system was initiated with free and open-source software projects, and some sponsors are interested in jump-starting a community of users around one of their products by making open-source tools or add-ons available to the software community. This led us to consider four types of projects:

- *Full-rights*. Students assign all rights to the project sponsor.
- *Limited use*. Students assign limited rights for internal use and the ability to extend and modify the software to the project sponsor. The students maintain all commercial and intellectual property rights.
- *Open source*. The project will be released into an open-source community using an open-source licensing agreement that is agreed to by the team and project sponsor. The sponsor will access the project artifacts through the open-source repository.
- *No assignment*. The student will keep all rights to the project artifacts, and not assign any to a project sponsor.

Working with RIT's legal counsel, we created project agreements for the first two projects types[6,7] that the team and sponsor sign during the first week of the project. These agreements also include a simple non-disclosure agreement that specifies that all information about the project is public except for material that the project sponsor explicitly identifies as proprietary. We specify a guideline that no more than 25% of a project's scope should be under non-disclosure coverage. Each project has a faculty coach assigned to it. The faculty coach does not assign any rights to the sponsor because he or she will not materially contribute to the technical activity of the project. The coach will, however, sign a non-disclosure agreement[8] covering any proprietary information provided to the project team. The project agreements are executed between the individual students and the faculty coach, and the company sponsoring the project. This is different than other NDAs which must be signed by the VP of Research and not an individual faculty member. There is no project agreement for the open source projects because the team agrees to make all project artifacts available to the open source community, and the sponsor will not disclose any proprietary information to the team. Finally, a student may opt to not work on any project that requires any assignment of rights to project artifacts. In this case, a faculty member serves the role of project sponsor, and creates an expanded course project for the student to work on. Over the seven years that we have identified these project types, only one student has opted to not assign rights.

In the project proposals[9], which the students see when stating their project preferences, the project sponsors identify the project type. For example, the description for a full-rights project in the proposal is:

*Assignment of Full Rights*

> *If a team is assigned to this project, all students on the team will sign a standard Student Course Project Intellectual Property and Non-Disclosure Agreement. This agreement assigns the rights to the team's project work to the*

> *sponsor, and describes the process whereby the project sponsor can reveal proprietary information to the team. For non-RIT projects, the faculty coach will sign a standard Faculty Course Project Non-Disclosure Agreement which describes the same process for revealing proprietary information.*

We have also found it necessary to include a checkbox for the sponsor to indicate that they have obtained corporate and legal approval of the project agreement, if any, which will be used. This was motivated by several recent projects where the sponsor did not seek approval before the project began, and then returned with an objection from their legal counsel, and a requirement for the team to sign their standard corporate agreement. It was this type of individual project negotiation that our standard project agreements were meant to eliminate. Since we have started using them, it has been much easier to explain ownership of project artifacts to sponsors. Since we are in the fortunate position to always receive more project proposals than we have teams, it is also easy to tell a potential sponsor that we will accept a proposal only if the project agreements are acceptable unmodified.

## Discussion

This paper described the approach we use for ownership of project artifacts and intellectual property associated with our capstone design course in the software engineering program at Rochester Institute of Technology. Because of the nature of these software-intensive projects and the expectations of creating working project artifacts that go well past proof-of-concept or prototype stages, the paper argued that these issues are heightened with software-intensive projects. Despite that claim the discussion in this paper is relevant to all capstone projects since issues of ownership of project artifacts do exist across the board.

The approach we have taken of defining several project types follow our local institutional policy, is transparent to the students, and accommodates students who want to retain all rights to their project work. For capstone courses that teach some required course material, this also provides an opportunity for the discussion of ownership and intellectual property rights. In our software engineering curriculum, having the students consider the implications of each project type brings the students full-circle back to topics that we discussed during Software Engineering Freshman Seminar. Project types similar to the ones we have defined can be used for projects that are not software-intensive. An open source project type is even available with the initiation of an open hardware movement[10].

Even though the development of our approach was driven by questions from external project sponsors, about one third of our projects are for internal RIT sponsors. These primarily business units within RIT are also interested in making use of the project artifacts. In this case, the students sign agreements that the Provost then signs as the RIT representative. This also includes the few projects sponsored by an individual faculty member who now has a clear indication of what he or she can do with the project artifacts after the project completes.

For information about other aspects of our software engineering senior projects, please refer to our senior projects webpage[11], the page for project sponsors[12], and the senior project information page[13].

## References

1. Lutz, M. J. and Naveda, J. F., “The Road Less Traveled: A Baccalaureate Degree in Software Engineering”, Proceedings 28th SIGCSE Technical Symposium on Computer Science Education, 1997.
2. http://www.se.rit.edu/pagefiles/documents/VSEN%20Flowchart%20v6.6%202013-09-08.pdf
3. F. Naveda, etal., "The Road We’ve Traveled: 12 Years of Undergraduate Software Engineering at the Rochester Institute of Technology," Proceedings of International Conference on Information Technology: Next Generations, Las Vegas, NV, April 2009.
4. L. Werner, etal, "Software Engineering Education via the use of Corporate-Sponsored Projects: A Panel Discussion of the Approaches, Benefits, and Challenges for Industry - Academic Collaboration," Proceedings of 2013 Conference on Software Engineering Education and Training, San Francisco, CA. May 2013.
5. RIT Intellectual Property Policy C3.0., http://www.rit.edu/academicaffairs/policiesmanual/sectionC/C3.html.
6. http://www.se.rit.edu/~swen-561/CourseInformation/StudentCourseProjectAgreement.doc
7. http://www.se.rit.edu/~swen-561/CourseInformation/StudentCourseProjectLimitedAgreement.doc
8. http://www.se.rit.edu/~swen-561/CourseInformation/FacultyCourseProjectAgreement.doc
9. http://www.se.rit.edu/~swen-561/CourseInformation/Senior%20Project%20-%20Proposal.docx
10. http://www.oshwa.org/
11. http://www.se.rit.edu/node/senior_projects
12. http://www.se.rit.edu/content/senior-project-sponsorship
13. http://www.se.rit.edu/~swen-561/Information.html