UNDERSTANDING COMPLEX SYSTEMS: THE NEED FOR AN OPEN SOURCE BASED HOMEWORK ASSIGNMENT IN CS 3140 AT UVA

HOW OPEN SOURCE SOFTWARE USED IN SOFTWARE DEVELOPMENT EDUCATION AFFECTS STUDENTS, PROFESSORS, AND THE NETWORKS THEY JOIN

A Thesis Prospectus
In STS 4500
Presented to
The Faculty of the
School of Engineering and Applied Science
University of Virginia
In Partial Fulfillment of the Requirements for the Degree
Bachelor of Science in Computer Science

By
Johnathan Eftink

November 11, 2024

Technical Team Members:
None

On my honor as a University student, I have neither given nor received unauthorized aid on this assignment as defined by the Honor Guidelines for Thesis-Related Assignments.

ADVISORS

Kent Wayland, Department of Engineering and Society

Rosanne Vrugtman, Computer Science

Introduction

How Can FOSS be Leveraged to Increase Understanding of Software Development?

Free and Open Source Software (FOSS) is an underutilized educational tool within college courses, despite its potential to enrich student learning. When incorporated into software development education, FOSS enables students to grow outside of purely academic code and understand what real-world software development looks like. This makes it a valuable addition to the computer science course CS 3140 (Software Development Essentials) at UVA. Integrating FOSS into the course will expose students to the world of FOSS while also increasing their understanding of software development practices.

Integrating FOSS into CS 3140 would involve a homework assignment in which students create a Unified Modeling Language (UML) class diagram based on an open source project. A UML class diagram visually describes how different portions of code interact with each other. The technical portion of my project involves designing this assignment to align with the learning objectives of CS 3140 and ensure it is manageable for students to complete.

Beyond enhancing their resumes, working with FOSS has been shown to increase students' confidence in their coding abilities (Salerno et al. 2023), among other benefits. It is important, however, to realize that there are also potential negatives that come with trying new homework designs. These will both be discussed more thoroughly in my STS paper in which I investigate how FOSS has been used in software development curriculums, the networks created by using FOSS in curriculums, and how the networks affect students, professors, and the existing contributors. For instance, students working with FOSS systems can join the open-source community, which can allow them to build on their networking skills and understand how professional interactions work. The overarching connection between the technical paper and the STS paper lies in exploring how FOSS can be integrated into software development education and the implications this has on students, professors, and contributors throughout the process.

Technical Project

Understanding Complex Systems: The Need for an Open Source Based Homework Assignment in CS 3140 at UVA

In the real world, most computer-based projects are not a few java classes, but a large collection of interleaving systems that require a deep knowledge of the entire code base. Unfortunately, many CS students at UVA are not adequately prepared for this complexity. The current Software Development Essentials course, CS 3140, at UVA has six homework assignments that range from starting a project from scratch to working on a project that is only missing some information and are all written by a single professor. While this variety of code base size does help prepare the students for the corporate world, none of these assignments currently provide a substantial code base written by multiple people. To address these limitations, a new homework assignment focused on analyzing large, real-world code bases from "Free and Open Source Software" (FOSS) projects can be introduced.

Adding a seventh homework assignment focused on analyzing a large code base will teach students how to read and understand FOSS code and navigate documentation. This assignment will help students practice breaking down complex code into understandable portions. Additionally, students will gain insight into how software development practices are applied in real-world projects. While these practices are taught in class, students often lack exposure to their application in larger systems. By analyzing FOSS code, students will see how these practices are used in non-academic code, deepening their understanding of their practical value and improving their ability to navigate large, real-world code bases.

The FOSS based assignment, however, should be limited in scope. Students must complete the assignment within a two-week time frame, so having the students contribute to the FOSS project would likely be too large of an undertaking. The assignment would also have to fit within the learning objectives of the course, which could present a challenge. To address these issues, students read, understand, and create a UML diagram to explain the structure of the FOSS project. The assignment would ask the students to diagram the connections between different portions of the code base and how they interact with each other.

Since students will have learned how to comprehend complex code, they will be able to walk into their jobs with the ability to be onboarded faster. This skill will not only set them apart by demonstrating their readiness to contribute meaningfully from the first day, but also build confidence in navigating unfamiliar codebases. Gathering feedback from students and graduates through polls about whether this skill distinguishes them from other interns/professionals will allow the department to improve the homework over time. This iterative process will allow professors to align the assignment more closely with both industry demands and student needs. Overall, this additional assignment will better prepare students for the corporate world by improving their understanding of large code bases and increasing students' confidence in their

abilities.

STS Research Problem

How Open Source Software Used in Software Development Education Affects Students, Professors,, and the Community They Join

Background of FOSS in Education

Free and Open Source Software (FOSS), while it may be used widely in the corporate world, is underutilized in education. Importantly, FOSS in education, in this context, refers to students collaborating and adding to an open source project, not using an open source learning management system such as "Moodle" (Lakhan, Jhunjhunwala 2008). An example of a researcher using FOSS in education in which students are contributing to is by Pinto et al. (2019). In this study, they have students contribute to a project, and then poll the students to see their opinions. This shows that contributions to FOSS projects have been used in education and will continue to be used.

Theoretical Frameworks

Actor Network Theory (ANT) is key in understanding the relationship between people and objects within a FOSS development system. The network of these relationships consists of "actors" that interact with each other. The human actors at play are professors, students, regular users, and contributors to the FOSS code. Professors worry about how good of an education they are providing the students. This means that homework assignments must meet department requirements and learning objectives. Students have the goal of learning the material and gaining experience that will be beneficial to them (e.g. helping them secure a job or perform better in academics). Though the contributors to the FOSS project do not directly interact with professors and students, they do engage with an actor: the project. The FOSS project itself affects students, professors, and contributors as all of those groups look to make their desired changes.

Another concept that will be analyzed is the "Community of Practice." A community of practice is a group of people that share a common interest that come together to fulfill both individual and group goals (ERLC). The community of practice that will be focused on is the FOSS community within software development. This community has developed their own set of standards and

common practices. Students entering this community must learn and understand these practices as well as making sure their goals, at least partially, match the group goals. I will analyze how students undertake joining the FOSS software development community of practice and the impact this involvement has on them.

Literature Review

To understand the role of Free and Open Source Software (FOSS) in education, it is essential to first define what FOSS is. FOSS, as described by Yang and Wang, is software that has many contributors to a project, and contributions can be done by anyone (Yang, Wang 2008). The value of FOSS, however, extends beyond the software itself as it has a network of actors- students, professors, contributors, and even the code itself. This concept is referred to as Actor Network Theory (ANT), that Warf describes as a social theory that is focused on how actors mobilize rules, resources, and power to accomplish tasks (2015). ANT, when used in a FOSS ecosystem, allows us to understand how individuals and objects in this network interact and influence one another. This theory provides a lens in which we can examine the relationships between various participants in a FOSS based educational environment.

With this understanding of FOSS, we can then start to understand how FOSS can be used in education, and how it affects the actor network. Recent research has begun to highlight the psychological and practical impacts of FOSS on student participants. Research by Wen takes a psychological standpoint and focuses on participants' perception of their learning within a FOSS system (2018). This study shows how contributors work in an educational environment and how they feel about their experiences. Salerno et al. investigated how students' view of themselves changes when they are the contributors (2023). This again shows that students experience positive psychological effects while working on open source projects. Pinto et al. expands on this by examining students as contributors, exploring how these groups communicated and developed skills within the FOSS ecosystem that they participated in (2019). This shows the practical skills students developed through their participation in a FOSS system.

It is also important to understand the role of the professor within this system. Research by Silva et al. and Pinto et al. highlights the difficulties professors face with FOSS-based homeworks, its impact on their classes, and the potential drawbacks of using FOSS (2019, 2017). These are important as it explains how professors work and react within a FOSS system as well as how actualizable a FOSS based class or homework may function. While the experiences of professors and students, as well as actor networks in software development have been studied, there is a research gap in understanding how students, professors, contributors, and the code itself

collectively develop an actor network.

These students and professors all enroll into an existing actor network in the software project. These existing actor networks will have developers, IT, managers, users, and the code itself (along with any infrastructure they use for this code). Linde et al. describe the different actors that play a role in IT-projects (2003). While Linde et al. describes actor networks for corporate projects, this will be similar to the actor network in FOSS projects since it is still developers enrolling into a network. Establishing a foundation of what a network looks like, we can further examine how students and professors integrate into this network.

Methods

To understand the actor network and how it affects the human actors in particular, it is crucial to break the network and the communities into multiple partitions. First, it is important to analyze the existing actor networks. This will be through research discussing how actor networks form inside of software projects, how they interact as time goes on, and how this affects contributors to the software projects. Understanding how the network affects initial contributors will shed light on how it affects students. Sources about the initial enrollment of people and objects into the actor network will also provide information about how new actors can be enrolled. Which leads to the second key partition, professors.

Professors chose to enroll students into certain existing actor networks. Why these decisions are made will give more insights into how students will eventually enroll into FOSS actor networks. Professors will also now interact with the FOSS actor network. Sources describing how these networks affect professors will help understand the actor network more thoroughly and shed light on the third key partition, students.

Students, when learning software development in a course, enroll in a specific project due to their professor. While their enrollment into a specific project might not be voluntary, it is still important to understand the opinion of students throughout this process. Sources that describe students' emotions and frustrations during this process will show how the initial enrollment stage affects students, and in turn will affect professors and contributors. From there, students will begin to exert greater influence on the network, which, in turn, will have an evolving impact on them. Sources that describe the continued opinions of students will help shed light on the newly changed actor network and the effects felt by all.

Conclusion

Free and Open Source Software (FOSS) offers benefits to students and educators and should be used in the UVA computer science program. Integrating FOSS into the curriculum of CS 3140 at UVA offers a powerful opportunity to enhance students' understanding of real-world software development practices. Exposing the students to FOSS will not only deepen their technical skills,

but also install a sense of confidence and readiness, giving them the tools they need to succeed in the professional world. Using the STS concepts of Actor Network Theory and Communities of Practice, we can analyze how students and professors are affected by the usage of FOSS in education. Specifically, the study aims to uncover how students and professors join a FOSS project, the network of relationships that are created from their participation, and the impact this has on the project's contributors. Analyzing how students join the community of practice can provide valuable insights into how they interact within a FOSS project. Specifically, it will show how students will be able to affect other contributors, as well as how other contributors affect students. With this understanding, an actor network that not only shows how contributors and students affect the non-human project, but also how the different contributing actors interact and impact one another. Overall, FOSS has many effects on students' education, networking, and confidence. By including a FOSS homework assignment in UVA CS 3140, students will be able to gain valuable benefits that will be useful for them in their career.

Works Cited

Edmonton Regional Learning Consortium. (n.d.). *What is a community of practice?*. Creating Communities of Practice. Accessed on 11/02/2024. https://www.communityofpractice.ca/background/what-is-a-community-of-practice/

Linde, A., & Linderoth, H. C. (2006). An actor network theory perspective on it projects. *Making Projects Critical*, 155–170. https://doi.org/10.1007/978-0-230-20929-9_8

Pinto, G. H., Filho, F. F., Steinmacher, I., & Gerosa, M. A. (2017). Training software engineers using open-source software: The Professors' Perspective. *2017 IEEE 30th Conference on Software Engineering Education and Training (CSEE&T)*. https://doi.org/10.1109/cseet.2017.27

Pinto, G., Ferreira, C., Souza, C., Steinmacher, I., & Meirelles, P. (2019). Training software engineers using open-source software: The students' perspective. *2019 IEEE/ACM 41st International Conference on Software Engineering: Software Engineering Education and Training (ICSE-SEET)*. https://doi.org/10.1109/icse-seet.2019.00024

Salerno, L., de França Tonhão, S., Steinmacher, I., & Treude, C. (2023). Barriers and self-efficacy: A large-scale study on the impact of OSS courses on student perceptions. *Proceedings of the 2023 Conference on Innovation and Technology in Computer Science Education V. 1*, 320–326. https://doi.org/10.1145/3587102.3588789

Shaheen E. Lakhan and Kavita Jhunjhunwala. (2008, May 5). *Open source software in education*. EDUCAUSE Review. https://er.educause.edu/articles/2008/5/open-source-software-in-education

Silva, F. G., Brito, M. S., Tavares, J. V., & Chavez, C. von. (2019). Floss in software engineering education. *Proceedings of the XXXIII Brazilian Symposium on Software Engineering*, *18*, 234–243. https://doi.org/10.1145/3350768.3353815

Yang, J., & Wang, J. (2008). Review on free and Open source software. *2008 IEEE International Conference on Service Operations and Logistics, and Informatics*, *1*, 1044–1049. https://doi.org/10.1109/soli.2008.4686552

Warf, B. (2015). Networks, geography of. *International Encyclopedia of the Social & Behavioral Sciences*, 567–571. https://doi.org/10.1016/b978-0-08-097086-8.72123-4

Wen, S.-F. (2018a). Learning secure programming in Open source software communities: a socio-technical view. *Proceedings of the 6th International Conference on Information and Education Technology*, 25–32. https://doi.org/10.1145/3178158.3178202

Evaluation for Prospectus Approval: APPROVED AS IS (No changes required)

Re-Grading Rubric for Full Prospectus

As noted before, there are two issues here. The first is the grade. I have to evaluate how much you improved this as a piece of writing. I will briefly review below the revisions you have made and then update the grade. The second is whether it passes the bar for approval as a Thesis Prospectus. That evaluation comes at the bottom.

Writing

On writing, note that this evaluation is still based on the presumption that everyone can write effectively. The only question is whether you have gotten to an A-quality paper yet, or whether it would need another iteration to get there. Note that this is my assessment; I am not suggesting that you revise this yet again merely for the quality of writing.

Introduction/Big Picture

Intro Grade: 1.9 out of 2 points (A)

Comments—Nicely done!

Technical Topic

Technical Project (Writing) Grade: 3.8 points out of 4 (A)

Comments—Looks good!

STS Topic

The STS Question

Background, Literature, Concepts

Comments—One of the more thorough applications of ANT I've had.

Methods & Evidence

Comments—Fantastic job describing the kinds of evidence that you want to collect, but could use some discussion of how you will get it.

Overall STS Grade: 14.2 out of 15 points (A)

Conclusion

Conclusion Grade: 0.95 out of 1 Point (A)

Reference List & Overall Formatting

References & Formatting Grade: 2 out of 2 points (A+)

Clarity & Coherence of Writing

Revised Grade: 22.9 out of 24 points (A)

Overall Comments: Looks good!

STS PORTION of the Prospectus Approved! No Corrections Required

I will be happy to accept this as a completed Prospectus. Note that I do not need to approve any revisions that you make in response to my comments above (if there are any). Make the changes that you deem appropriate, get the Prospectus into a form that you are comfortable with (being preserved in the library and available to all …), make sure it is formatted appropriately, and then save it as a pdf to be uploaded at the end of next semester. You should then upload it to Canvas under the "Upload Final Approved Prospectus" assignment.

Congratulations on finishing!