2023

Theorizing a Low Entry-Cost, Interdisciplinary Undergraduate Curriculum for Open Source Project-Based Learning

Jeffrey D. Wall

Theorizing a Low Entry-Cost, Interdisciplinary Undergraduate Curriculum for Open Source Project-Based Learning

Jeffrey D. Wall
College of Business, Michigan Technological University
jdwall@mtu.edu

ABSTRACT

As information systems undergraduate students enter the workforce, they often work with business professionals, computer scientists, and IT specialists to analyze business problems to design and implement fitting technological solutions. Yet, many MIS courses fail to expose students to interdisciplinary project-based work to prepare them for the eventualities they will experience in industry. MIS, CS, and IT curricula may include project-based work, but often in isolation. We propose the design of a project-based curriculum that utilizes free and open source software (FOSS) ecosystems to bring together students from across business and computing disciplines to solve real-world problems. Drawing on issues currently experienced in programs that utilize FOSS ecosystems for student learning, we propose methods that may offer lower entry-costs for faculty and students in terms of human and financial resources.

Keywords

Interdisciplinary education, open source software for education, systems analysis and design, project-based learning

INTRODUCTION

Providing project-based experiences in academic courses can help undergraduate students prepare for the rigors they will encounter in the workforce (Müller, Schindler and Slany 2019; Pinto, Ferreira, Souza, Steinmacher and Meirelles 2019). Many different project experiences can be implemented in coursework. For example, faculty can arrange projects with a particular external client. Faculty may also ask students to identify and develop solutions for problems that exist in an industry without a specific client. Further, faculty may engage students in project work with free and open source software (FOSS) ecosystems. These FOSS projects provide students with an experience similar to working with a client to make real contributions to a project (Papadopoulos, Stamelos and Meiszneer 2013). The scope of this paper explores the use of FOSS ecosystems to support project-based learning in MIS courses.

Many studies have identified the benefits of utilizing FOSS ecosystems to support project-based learning in the classroom over other project-based learning models. However, most of these efforts are studied in computer science and software engineering programs. Few studies have explored how FOSS project-based learning can draw on the strengths of multiple business and computing disciplines, except in unsystematic ways (Chacon and Bornstein 2014). Although FOSS project-based learning shows a great deal of promise, the literature identifies a series of issues that make it difficult to implement. Strong FOSS project-based learning can require administrative overhead that create barriers to entry for faculty and programs (Ellis, Hislop, Pulimood, Morgan and Coleman 2015). In fact, some FOSS ecosystems will only work with students from elite programs to ensure that the time ecosystem members spend with students is productive (Chacon and Bornstein 2014). We propose a model for FOSS project-based learning in which MIS, CS, IT, and other computing and business disciplines can collaborate to better mimic the interdisciplinary analysis, design, implementation, and maintenance work that takes place in industry while minimizing administrative overhead.

The remainder of this paper continues as follows. First, a brief literature review is provided to present the current state of FOSS project-based learning for undergraduate students. Second, an interdisciplinary model for FOSS project-based learning that is designed to minimize administrative overhead is presented. Third, a discussion of the potential benefits and limitations of the model is given.

EXISTING MODELS FOR FOSS CURRICULUM

FOSS project-based curriculum can be implemented in a variety of ways. FOSS projects can focus on a variety of different student contributions to an ecosystem depending on the desired learning outcomes of a course. For example, students can work on ***forward engineering projects*** that involve the creation of new features; ***reengineering projects*** that involve the improvement and refactoring of existing features; ***corrective projects*** that focus on identifying and fixing software bugs; and ***management projects*** that involve a variety of managerial concerns for the ecosystem, such as identifying better project management and DevOps processes, improving documentation within the ecosystem, improving the onboarding processes for new members, etc. (Pinto et al. 2019). Although most of these activities have been studied in the context of computer science and software engineering degrees, these activities are highly relevant to MIS students as well.

Curricular models can also differ in their complexity. I***ndividualistic student-led model*s** rely on simple procedures in which each student selects a FOSS ecosystem to contribute to. Students largely manage the interactions with the ecosystem by themselves (Papadopoulos et al. 2013; Pinto et al. 2019). A slight alteration to such a model is a ***pair-led or group-led model*** in which pairs or groups of students choose an ecosystem and contribute to it. A more complex model includes a ***standing ecosystem relationship*** in which faculty develop a strong relationship with one or a few FOSS ecosystems that students contribute to, requiring some administrative overhead (Müller et al. 2019). More complicated models include ***multi-institution models*** in which multiple universities (regionally or globally) collaborate with a small set of FOSS ecosystems (Ellis et al. 2015). These multi-institution models are designed for elite or mature programs and require substantial administrative overhead, limiting new entrants (Chacon and Bornstein 2014).

A LOWER ENTRY-COST, INTERDISCIPLINARY CURRICULAR MODEL FOR FOSS PROJECT-BASED LEARNING

We propose a theoretical model for the design of FOSS project-based learning that can draw on the interdisciplinary strengths of various business and computing disciplines, while providing a lower entry cost for faculty and students than some of the models described previously. We focus the model around the careful selection of ecosystems that support interdisciplinary work and developing incremental capabilities for learning to manage administrative overhead.

Ecosystem Selection

To maximize the benefit that students from computing and business disciplines can obtain from a FOSS project, we suggest that FOSS ecosystems should be selected for their focus on managerial concerns, degree of relatability to students, potential for user interactions, and capability for contributing stand-alone plugins.

Find Ecosystems with a Managerial Focus

FOSS projects exist around a variety of problems that relate well to computer science students, but may not always be applicable to business students. To maximize the interdisciplinary value of FOSS projects, we suggest that projects should selected based on their relation to managerial problems. Many FOSS tools support management, such as open source enterprise resource planning (ERP) systems and modules, open source project management tools, open source ecommerce tools, etc.

Find Ecosystems with a Relatable Problem

FOSS projects are diverse with topics spanning from biology, to forestry, to accounting and finance. Not all such projects may be relatable nor interesting to students. Although identifying systems with a managerial focus may assist in increasing the relatability of projects, further selectivity could support engagement and minimize entry costs to learn new business systems. For example, undergraduate students are integrated into a university business system surrounded by information systems. Several management systems exist specifically for universities, including FOSS learning management systems (LMS) like the Canvas LMS project (https://github.com/instructure/canvas-lms) or ERP systems designed for university management like the OpenEduCat ERP project (https://github.com/openeducat). These tools are designed to support managerial concerns that are highly relatable and relevant to the lives of students. Students can produce products that are directly related to problems they experience.

Find Ecosystems with Potential for User Interactions

MIS and other business students would benefit greatly by engaging in FOSS projects with easy access to potential system users. Not all ecosystem will provide this easy access. With access to potential users, students can engage in requirements gathering and customer discovery activities to help identify and prioritize features and engage in regular Agile testing and feedback gathering activities. Again, FOSS ecosystems built around university management systems may provide for easier interaction

with users. For example, students themselves possess knowledge of problems, but also have easy access to other students, faculty, and administrators who possess direct knowledge of management issues.

Find Ecosystems with Plugin/API Capabilities

Developing strong relationships with members of a FOSS ecosystem is one of the major administrative overheads in starting and maintaining FOSS project-based learning. Not all FOSS ecosystems are open to student engagement and others are only interested in working with elite programs. To avoid this, we suggest selecting open source projects that provide more ways to integrate with the software than simply by contributing to the official code-base. Finding FOSS projects that possess a well-documented application programming interface (API) provides different ways to interact with the FOSS ecosystem. For such ecosystems, faculty and students can either make contributions directly to the FOSS code-base or develop their own plugins using the API that can be shared through a public repository. By allowing this flexibility, programs that don't possess the time, resources, or ability to develop close relationships with the FOSS community can still engage indirectly with the ecosystem. This may gradually lead to closer interactions with the FOSS ecosystem through integration of student plugins into the core system. Canvas LMS and OpenEduCat provide examples of managerially focused systems that are relatable to students, allow for easy interaction with system users, and provide APIs for plugin creation (https://canvas.instructure.com/doc/api/ and https://openeducat.org/feature-lms).

Developing Incremental Capabilities

Beyond selecting appropriate projects for interdisciplinary work, we suggest that interdisciplinary FOSS project-based learning will benefit from a patient process in developing mature capabilities. Programs do not need to rush into large scale FOSS projects. We suggest that students and even FOSS ecosystems can benefit from thinking about onboarding materials and by carefully planning interactions between courses and FOSS ecosystems and between courses across disciplines.

Developing Infrastructure/API/DevOps Tooling Trainings

Many FOSS ecosystems provide poor documentation of their code-base and management processes, making it difficult for new members to onboard quickly (Pinto et al. 2019). The complexities of FOSS ecosystems coupled with poor documentation may create a barrier to entry particularly for undergraduate students. However, if taken with an improvement and managerial mindset, these barriers present an interesting opportunity for both technically- and managerially-oriented students. We suggest that some of the first projects faculty and students should consider when selecting a new FOSS ecosystem to work within should include developing onboarding materials and documentation for the FOSS project. This may include reviewing and modeling the architecture of the system, designing procedural documents about how to contribute to the ecosystem, providing trainings on how to use some of the DevOps tools used to contribute to the ecosystem, developing and documentation for specific sections of the project. These project deliverables can benefit both the FOSS ecosystem and future courses that engage with the ecosystem.

Delaying Interactions with the FOSS Community

Although some of the most rewarding and beneficial interactions with FOSS ecosystems may come from close interactions between students and mentors from the FOSS community, this may not be feasible for programs as they begin to implement such learning experiences. We suggest that if FOSS projects are carefully selected using criteria mentioned previously, students will be able to gain exposure to working with users and build something that potentially useful to the larger ecosystem. Over time, these plugins can act as a signal to the FOSS ecosystem that a course within an academic program might provide more direct value to the ecosystem. In some cases, programs may wish to avoid direct FOSS ecosystems indefinitely to avoid administrative overhead and grant faculty greater control of the learning process.

Delaying Interdisciplinary Collaboration

Although the purpose of this model is to develop an interdisciplinary model for FOSS project-based learning, we suggest that programs new to this style of learning should avoid jumping into interdisciplinary collaborations too early. It might be useful for faculty across disciplines to discuss early in the process the FOSS projects that could maximize interdisciplinary collaboration using the criteria presented earlier. However, it might not be as useful to begin scheduling courses across disciplines for interdisciplinary interaction or other related planning. For example, to minimize the overhead of planning during early adoption, CS students could select tasks within a FOSS ecosystem relevant to their course of study (e.g., documenting the core architecture of the project), IT students could do the same (e.g., document the DevOps tools and operating environments of the software), and MIS students could engage in managerial tasks (e.g., customer discovery, improving FOSS onboarding processes, documenting project management and development processes, etc.). In this way, each group could act

independently on projects most pertinent to their disciplines without the need for direct collaboration. As the programs mature in their use of FOSS project-based learning, close collaborations between interdisciplinary programs can be gradually integrated. For example, programs might schedule project courses at the same time to allow for interaction, cross-list project courses across disciplines to allow students to gain interdisciplinary exposure, develop planning committees, etc.

DISCUSSION AND CONCLUSION

FOSS project-based learning presents a beneficial tool to prepare CS, IT, and MIS students for the work they will do when they enter the workforce. These projects can provide students with exposure to requirements gathering and customer discovery, systems architecture, DevOps processes and tooling, and interdisciplinary work environments.

Most existing literature is designed to support CS degree programs in their engagement with FOSS ecosystems. To support interdisciplinary collaboration on FOSS projects that better accommodate MIS programs, we provide a set of theoretical guidelines for FOSS project selection and implementation. Projects should be carefully selected to provide exposure to managerial topics (e.g., ERP systems) around a system familiar and relevant to students (e.g., university learning and management systems) that grant students access to potential system users and flexible ways to contribute directly or indirectly.

While much of the literature suggests that the best FOSS experiences should include close interactions between the academic program and members of the FOSS ecosystem (Chacon and Bornstein 2014; Ellis et al. 2015), this isn't immediately feasible for many programs. Programs starting with FOSS project-based learning must build capabilities over time. We suggest important activities that are sometimes ignored in the existing literature. After selecting an ecosystem to work in, MIS programs can begin to develop capabilities by analyzing and documenting management processes, architectures, customer needs, etc. for the ecosystem. Even if such documents aren't used by the ecosystem itself, they can act as learning tools for subsequent classes.

FOSS project-based learning provides ample opportunities for growth. It can feel daunting for faculty and students to engage in such projects. However, with careful selection of projects and a patient and improvement-oriented mindset, MIS programs may develop meaningful capabilities in FOSS project-based learning.

REFERENCES

1. Chacon, S., and Bornstein, J. (2014) Modernizing CS education with open source, *Open Source Convention 2014*, Portland, Oregon.
2. Ellis, H. J., Hislop, G. W., Pulimood, S. M., Morgan, B., and Coleman, B. (2015) Software engineering learning in HFOSS: A multi-institutional study, *112nd ASEE Annual Conference & Exposition*, Seattle, WA: ASEE.
3. Müller, M., Schindler, C., and Slany, W. (2019) Engaging students in open source: Establishing FOSS development at a university, *52nd Annual Hawaii International Conference on System Sciences*, Wailea, HI.
4. Papadopoulos, P. M., Stamelos, I. G., and Meiszneer, A. (2013) Enhancing software engineering education through open source projects: Four years of students' perspectives, *Education and Information Technologies,* 18, 381–397.
5. Pinto, G., Ferreira, C., Souza, C., Steinmacher, I., and 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*, Montreal, QC, Canada: IEEE.