Engagement Models in Education-Oriented H/FOSS Projects

Grant Braught
Dickinson College
Carlisle, PA, USA
braught@dickinson.edu

Steven Huss-Lederman
Open Energy Dashboard
USA
StevenHL@OpenEnergyDashboard.org

Stoney Jackson
Western New England University
Springfield, MA, USA
hjackson@wne.edu

Wes Turner
Rensselaer Polytechnic Institute
Troy, NY, USA
turnew2@rpi.edu

Karl R. Wurst
Worcester State University
Worcester, MA, USA
kwurst@worcester.edu

ABSTRACT

Engaging students in free and open source (FOSS) projects can provide significant curricular benefits but is known to be challenging for both students and faculty. This paper reports on our efforts to mitigate these challenges through the creation and use of *Education-Oriented H/FOSS (Humanitarian FOSS or FOSS) projects* — authentic open source projects consciously designed and managed to facilitate student and faculty engagement. We describe four active Education-Oriented H/FOSS projects and introduce a framework for illustrating different models of H/FOSS engagement. The framework is used to structure a discussion of the considerations and trade-offs of different engagement models, and highlights particular models that have been used to engage students and faculty in our four Education-Oriented H/FOSS projects. The framework positions projects along dimensions of professor involvement, responsibility for project hosting/management, mode of student knowledge and skill acquisition, and the curricular engagement goals. In doing so it broadly captures trade-offs that exist between the level of institutional resources used and the level of student independence required. It is anticipated this framework and the discussion that it organizes will be useful to faculty a) in evaluating the appropriateness of particular H/FOSS projects for use in their courses and curriculum and b) as guidance to those considering the creation of new Education-Oriented H/FOSS projects.

CCS CONCEPTS

• **Social and professional topics** → **Computer science education**; • **Software and its engineering** → *Open source model*.

KEYWORDS

Curricula, Open Source, Humanitarian, Engagement Models

ACM Reference Format:
Grant Braught, Steven Huss-Lederman, Stoney Jackson, Wes Turner, and Karl R. Wurst. 2023. Engagement Models in Education-Oriented H/FOSS Projects.

SIGCSE 2023, March 15–18, 2023, Toronto, ON, Canada

ACM ISBN 978-1-4503-9431-4/23/03…$15.00
https://doi.org/10.1145/3545945.3569835

In *Proceedings of the 54th ACM Technical Symposium on Computing Science Education V. 1 (SIGCSE 2023), March 15–18, 2023, Toronto, ON, Canada.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3545945.3569835

1 INTRODUCTION

Free and Open Source Software (FOSS) is software developed by communities committed to a common cause, and where the source code produced is open to all for study, modification, redistribution and adaptation to new purposes. Beyond the openness of the source code, these communities also aspire to make their designs, discussions, feedback, and decision making open, transparent and publicly available. This openness, the argument goes, has many benefits, spurring innovation, supporting both collaboration and competition and producing better, more secure software [1, 24, 38, 41]. Many FOSS tools, products and processes are now central to modern software development and FOSS skills and experience are in high demand [16].

As computing educators, the openness of FOSS presents exceptional curricular opportunities for us to foster our students' technical, professional and personal development. Students working with FOSS are exposed to a full range of large, complex, real-world software artifacts and development processes, giving them experiences that cannot be achieved with class-size projects. When engaging with a project's community, students observe and practice professional skills including communication, collaboration, critical thinking, question asking, and technical writing. The almost limitless breadth of FOSS projects can inspire and motivate students as well. For some students the authenticity of the technical work and the resume building aspect is motivating. For others, seeing how their computing skill set can empower them to impact causes that they care about is inspirational. More complete discussions of these and other benefits can be found in the literature [8, 23, 25, 36]. A review of a wide variety of ways that educators have been incorporating FOSS into curricula can be found in Braught et al. [3].

Incorporating FOSS into computing curricula may also have broader individual, programmatic and institutional benefits. This is particularly true when the FOSS projects involved have connections to needs on campus, in the local community and/or have wider humanitarian goals. Collectively, these types of projects are referred to as Humanitarian FOSS (HFOSS)[1]. Beyond the advantages above, the integration of HFOSS into the computing curricula has

[1]We use FOSS to when generally referring to Free and Open Source Software, HFOSS when referring to Humanitarian FOSS, H/FOSS when the effort may or may not be Humanitarian, and OSS for Open Source Software.

been argued to help counter the "computing-is-coding myth" and draw the campus and local communities closer together [27, 39]. Murphy et al. [29] cast HFOSS efforts explicitly in terms of community engagement that resonate with more general calls for community and civically engaged computing [18, 32] and with related efforts on computing for social good [20]. HFOSS engagement holds the promise of broadening participation in computing by presenting computing, not just as tech, but as a powerful means for students to positively impact causes and communities that they care about [2, 11]. Computing students, particularly those with less programming experience, view computing majors and careers more favorably after taking an HFOSS engaged course [22]. When given the freedom to engage with H/FOSS projects in a capstone course, women were more likely to select projects with humanitarian goals than were men [4]. These results are consistent with recent findings that suggest that computing courses that emphasize people over things attract more women students [21].

While there are the above advantages of incorporating H/FOSS experiences into the curriculum, the literature also identifies a number of significant challenges for both faculty and students [7, 8, 12]. Engaging with H/FOSS projects can impose a significant learning curve on both students and faculty. Students must learn to navigate substantially larger and more complex code bases. With H/FOSS projects that are external to the curriculum, students may also need to learn new programming languages, libraries and tools. For faculty, having students working on external projects requires managing a classroom environment where they are no longer the authoritative expert on the project [8]. To contribute to an H/FOSS project, students (and faculty) will need to learn modern workflows and processes typically using git and GitHub or GitLab.

Interacting with H/FOSS communities within an academic environment also presents a number of cultural challenges. The work in H/FOSS communities is less structured, even chaotic [33], as compared to more traditional academic assignments. The pace of work in a H/FOSS community may also not align well with course timing or assignments. Community and maintainer responses to student inquiries, questions and contributions may be slow, inaccessible, or off-putting in a way that affects the students' progress. Projects can also be unpredictable, affecting student work and progress by changing underneath them (e.g. changing feature requirements, rolling out a new version, upgrading packages, shifting tools midsemester or having a fix/bug completed by others). There has also been concern that combinations of these factors may cause students to focus on smaller tasks, resulting in a narrower technical experiences than they might have in more controlled academic assignments [35]. Some studies have reported challenges such as these may cause student confidence in their abilities to decrease during H/FOSS engagement [9, 23]. They however, also note that this may be a natural correction due to students having initially overestimated their skill set.

Given the potential benefits of incorporating H/FOSS into the curriculum, significant effort has been invested in investigating ways to help mitigate the highlighted challenges. Instructors have created processes to help students select H/FOSS projects that have appropriate complexity and supportive communities [12, 28]. Other processes have focused on helping faculty select projects of uniform complexity so that students within a course have comparable experiences [19, 34]. Ellis et al. [10] describe a model that attempts to mitigate challenges with faculty expertise and project culture by having the instructors engage in the H/FOSS project community to the level of becoming project maintainers prior to engaging their students in the project. Morgan and Jensen [28] found that students were less overwhelmed by the initial learning curve when working in smaller projects where a project member agreed to serve as a mentor for the students. They note, however, that this comes with the cost of less freedom of project choice, and Gehringer [17] cautions that this approach can require up to 3-months lead time for project managers to prepare for the students. Tucker [40] advocates a CO-FOSS approach that mitigates the learning curve by having faculty define and scope a short-term H/FOSS project in collaboration with a local non-profit. Buffardi [5] proposes a localized H/FOSS (LFOSS) model that can mitigate learning curve and faculty expertise challenges by having students collaborate with local software professionals to contribute to a H/FOSS project.

With the above background and context, the remainder of this paper discusses our experiences using Education-Oriented H/FOSS projects. We begin by positioning Education-Oriented H/FOSS projects as one part of a larger effort (Section 2) to facilitate H/FOSS engagement while mitigating some of the challenges that it presents. We then introduce a framework (Section 3), based on our experiences, that provides a structure for thinking about and discussing some of the important design considerations and trade-offs that are involved in designing H/FOSS engagements. Four active Education-Oriented HFOSS projects are used to illustrate the framework (Section 4) and provide concrete examples of the many considerations and trade-offs involved in joining or starting such a project. The projects are compared and contrasted to provide examples of where Education-Oriented H/FOSS projects have been used in courses/curricula (Section 5), how faculty and institutions can engage with them (Section 6), how projects are organized (Section 7) and issues around onboarding students (Section 8). All of this discussion is ultimately intended to aid interested faculty in evaluating existing or starting new Education-Oriented H/FOSS projects.

2 OUR APPROACH

The NSF grant *Broadening Participation in Computing through Authentic, Collaborative Engagement with Computing for the Greater Good* (OpenPACE) supports a scaffolded approach to enabling undergraduate student participation in HFOSS projects. Previous work concentrated on the two ends of the process - at the beginning introducing classroom activities for familiarizing students with open source communities, tools, and processes; and at the end helping faculty prepare for having their students participate in existing "in the wild" open source communities and projects. However, we realized a large gap existed between these two ends so many faculty had difficulties in making the leap from the classroom activities to students participating in existing projects. Current work involves two intermediate steps - *HFOSS Kits*, which are HFOSS projects, frozen at a particular point in time, along with classroom activities that have students work within the full project; and, the focus of this paper, *Education-Oriented H/FOSS Projects*, which are ongoing H/FOSS projects with real clients but managed by educators with development work done primarily by undergraduate students.

3 AN ENGAGEMENT-MODEL FRAMEWORK

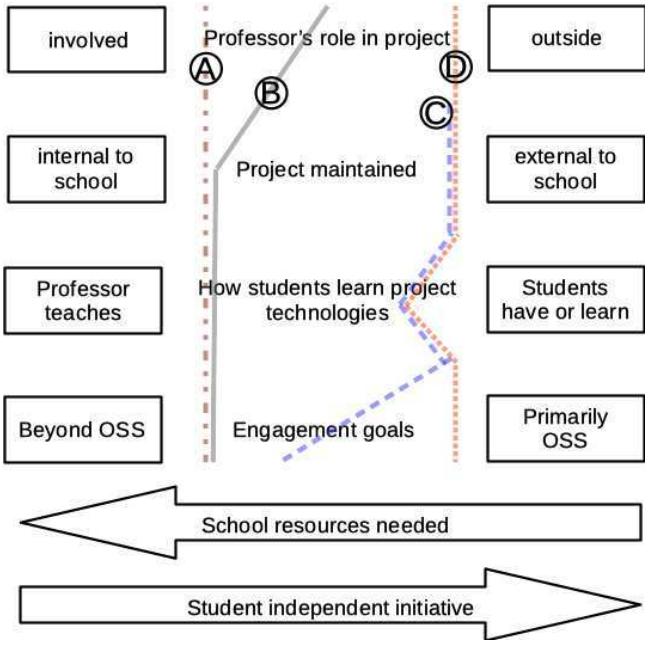

Figure 1: A Framework for Engagement-Models relating design considerations to school resources and student initiative.

Section 1 discussed the benefits and challenges of incorporating H/FOSS into computing education. This section lays out a framework for discussing and comparing different models of H/FOSS engagement. Figure 1 shows the following four design considerations that, based on our experience, seem significant:

What is the professor's role in project? Is the professor using the H/FOSS project involved in that project as a developer or maintainer, or do they play no/minimal role and so are outside the project?

How the project is maintained? Is the H/FOSS project developed and maintained internal to the school using the project or is it external? If multiple schools maintain the project then it is considered internal to each of them.

How do students learn project technologies? Does the professor educate students on H/FOSS project technologies or do students learn them on their own, possibly with project resources?

What are the engagement goals? Is the goal of the student engagement beyond open source, e.g., general software engineering, or is the focus primarily on open source?

These design considerations lie on a continuum. In general, being farther to the left side of the figure requires more school resources (professor's time, department or institutional resources) whereas being to the right side normally requires greater student initiative (more agency, independence). A possible exception is that the level of student initiative required may be independent of the engagement goals. For example, whether a course is focused on software engineering or FOSS may or may not impact the level of independence required of students.

While it is possible to have any combination of the four considerations, certain combinations seem more likely. The ones that our schools have utilized are represented by the vertical lines in the figure:

A. brown, dotted/dashed line The project resides with the professor/school so the course teaches about the project but has other goals. Often a capstone or software engineering course, but can be others, e.g., mobile applications or open source course. Example schools: Beloit College (BC), Dickinson College (DC), Nassau Community College (NCC), Western New England University (WNE), Worcester State University (WSU).

B. solid, gray line Similar to A but the professor may not be as heavily involved in the project. This can happen if there are multiple professors teaching the course where some are not leads on the project. Example school: DC (also A since one professor maintains the project).

C. blue, dashed line Here the students are either working on a special project or internship that may be external to the school. Thus, the professor/school does not have a role in the project and the student learns independently. These students often have multiple goals including OSS experience, learning new skills and resume building. Examples: All schools discussed and others not explicitly mentioned.

D. orange, dotted line The project resides outside the professor/school and the primary goal is the project experience. In this scenario, students may have experience with the project technologies but usually need to independently augment this knowledge. Examples include OSS or capstone courses where students spend the majority of their time engaged with the project. The project may be chosen by the professor or picked by the student(s). Example schools: California State University Monterrey Bay (CSUMB, project not required to be FOSS) and Rensselaer Polytechnic Institute RCOS (RCOS).

It is important to note that a school's engagement model can shift. For example, a school may start with model D on the right side of Figure 1 with minimum resources needed and then transition toward the left side as the professor(s)/school get more experience and become more invested/comfortable in open source. We also note that not all viable combinations are listed as we have not utilized them. One example is where a professor can be heavily involved but the project is outside the school.

4 THE PROJECTS

The previous section illustrated the H/FOSS engagement models used by several different schools. This sections introduces four Education-Oriented H/FOSS projects and considers how their designs facilitate (or inhibit) their use in different models (and thus by different schools). All of the projects discussed are active and engaged somewhere between 60-200 students. These projects also share a combination of characteristics that we believe are important to Education-Oriented H/FOSS projects:

- Ongoing projects with real clients, not single-semester or one-off projects.
- Working with students is central to the project. They all devote resources to this effort and understand that this focus may slow project development.
- Willingness to work with adopting faculty to support additional curricular goals. They all encourage interested faculty

to contact them and will work with those faculty to help fit the project into their course(s).
- Avoid making changes that could negatively affect courses in progress.
- Project maintainers provide regular, ongoing and high quality mentoring of students.
- Onboarding is as easy as possible including extensive documentation, containerized installations and personalized help.
- An emphasis is placed on documentation, issues and tasks that will work for students in both complexity and within the required time frame.
- Project application areas are designed to engage students. They may be humanitarian and connect to causes or communities that aligned with institutional mission, or they may connect with students' educational or residential experiences.
- Modern, real-world, complex technologies are used.

A brief introduction to each project and a description of how it fits into the framework in Figure 1 is now given. Additional detail is provided in the subsequent sections.

FarmData2 (FD2) provides support for the operation and certification requirements of small organic farming operations [13]. It is organized at Dickinson College and began, in its current form, in 2020. So far it has followed frameworks A and B with a few students doing C.

LibreFoodPantry (LFP) is a community of clients, users, and developers, spanning multiple academic institutions, who believe in developing and maintaining quality and adaptable HFOSS projects to support local food pantries [26]. It is organized by Nassau Community College, Western New England University, and Worcester State University beginning in 2019. So far it has followed framework model A with some students doing C.

Open Circuits is an efficient, cross-platform and user-friendly digital circuit designer, esp. for academic use [30]. It has been hosted at RPI/RCOS since 2018. It utilizes framework D.

Open Energy Dashboard (OED) provides software to acquire, analyze, store and visualize resource usage at an institution [6]. It is an independent project engaged with multiple schools (nine to date) that began in 2016. It originally followed framework model A but now utilizes C and D.

Note, a project can shift models or simultaneously use multiple models. For example, several of our projects use model C along with another model and OED shifted from model A to model D but is looking to reengage with model A.

5 CURRICULAR GOALS AND POSITIONING

H/FOSS engagement has been integrated into the curricula in a variety of ways and with a variety of goals. This section considers four interrelated questions that illustrate some of the choices that have been made by different engagement models and projects:

What potential roles does a H/FOSS project play in the curriculum? Is the goal to engage an individual student, a small team, a course or to integrate H/FOSS more generally? The H/FOSS project experience may be tightly coupled to a course as shown in framework models A and B. One option is where the project is an integral part of the course materials being taught. An example is LFP where the project is fully integrated into a course and used by all the students and class discussions (NCC, WNE, WSU). A related but different option is where a significant engagement with H/FOSS projects serves as a vehicle for achieving some course objectives but the project is not referenced directly during class instruction. Often different students or teams are working on different projects within the course. Examples are schools using OED (CSUMB, RCOS) and Open Circuits (RCOS). In another model, the H/FOSS experience can be central to multiple courses and the curriculum which is generally in framework models A and B. An example is FD2 that is used in two courses and heavily integrated into course materials to utilize H/FOSS to teach multiple learning objectives over time (DC). In another model, an individual student or team work on a project as shown by framework model C. All of our projects and many schools have done this to varying degrees. These students might be doing an independent project for academic credit, a summer internship, receiving pay for working on the project or doing it independent of their school during the academic year. While this engagement can be part of a curriculum where many/all students do this, it is often an extracurricular for specific students. Several of the schools involved in this paper and OED have had students engage in this fashion either before or after the course(s) involved. This can either prepare them for the H/FOSS course experience or deepen it after the course. From a project standpoint, students who continue to engage with the H/FOSS project generally produce higher quality commits and are more valuable to the project. Other models exist, e.g., a quick bug fix, but are not the focus here. As Figure 1 indicates, these different types of curricular engagement use different levels of school resources and student initiative. As a rule, the more tightly coupled to a course or curriculum, the greater the investment by the school.

Where is the H/FOSS experience integrated into the curriculum? Very often the H/FOSS experience is done in the later parts of the curriculum such as a capstone or advanced course (e.g., software engineering) [3]. Since students at this level should have more initiative, technical skills and experience, this can mitigate some of the issues discussed in Section 1 and thereby require less investment during the course. It is also possible to introduce students to H/FOSS at earlier stages. This can occur in the first two years at a community college (e.g, NCC), or by direct integration into second-year courses (see DC above) or with more limited experiences in introductory courses. Except possibly in the case of very limited experiences, these models generally require greater school resources to support students who have yet to develop a high level of independence.

How can a H/FOSS project actually be integrated? Is learning specific technologies an objective or are the technologies ones that students already know? If a specific technology is important, then project selection is narrowed. Examples might be the project language (Java, SQL) and/or underlying technologies (application architecture, UI framework, testing methodology). As a rule, the longer students engage with a project, the more time they can invest in learning the necessary technologies as part of the engagement. A second approach is to have many possible projects so students can better match their technical background and learning goals to a project. In the RCOS course, students talk to projects to find a mutually agreeable match. In the CSUMB course, student teams

get project descriptions, rank choices and are then assigned by the professor. In all these cases, students can work individually on tasks or as a team to meet course goals. Teams can allow students to help each other with challenges they face. Some schools make the decision partly based on the curricular goal, e.g., if teamwork or software engineering processes are important learning goals.

How to assess students? Assessment in a H/FOSS project offers unique challenges. These occur because tasks can be amorphous, the work is reviewed by others and the complexity of what was done can be difficult to gauge. While no suggestions are given here, we offer some ideas on what has been done. Professors have tried to get a sense of the scope and quality of work done by analyzing the repository where work is committed (issues, commit/merge record and/or the actual code) to the H/FOSS project and/or project communication channels. Another technique is self assessment and, when multiple students are involved, peer evaluation. It is also possible to have some form of progress reports that provide a record of work done, time spent, issues faced and items learned. We note that education-oriented projects may be better suited to getting the needed information either from the professor being directly involved and/or from the project leads. Finally, when it is difficult to get a precise assessment, coarse-grained grades can be used, e.g., P/F. All of these techniques have been used in varying combinations by the schools outlined in this paper.

6 FACULTY AND INSTITUTION ENGAGEMENT MODELS

Faculty wishing to engage with existing Education-Oriented H/FOSS projects can chose to do so in a number of ways. They can use a project as an object of study in their own class. They can fork it and have their students make modifications, and choose to offer those changes back to the original project (or not). They can request elevated (merge approval) privileges, or participate in project governance. They can also advocate for the use of the project at their institution to support the institution's operations and/or mission.

The faculty/maintainers for Education-Oriented H/FOSS projects serve in a number of different roles in their project. They all teach (or have taught) courses using their projects, so they have experience in course design, pedagogy and student assessment and can tailor their projects to support the student experience. In addition, they have the experience to help other faculty integrate the project into their course(s). Finally, they serve as product managers, architects, and technical leads for the project, setting the direction for their projects.

Class size can have an impact on how the faculty member chooses to have students interact with the project. For example, a faculty member may wish to have the entire class engage in a single project which reduces the needed knowledge. This may present some challenges for larger classes though, as it is important to be sure that the project and its maintainers can accommodate the number of students. However, most projects are designed in a modular fashion and teams of students can be assigned to different modules, especially if teamwork is a desired outcome of the course. Also, if the faculty member is willing to act as project manager for the students in their class, they can help assure timely response and reviews for their students' work without overwhelming the project maintainers. Naturally, this requires more faculty knowledge of, engagement with, and privileges from the project maintainers, but many projects are willing to provide this in exchange for increased community participation.

If a faculty member does not wish to be as heavily involved in the project and/or wishes to allow the students more choice in the project that they work on, multiple projects can be offered to the students as options as outlined in Section 5. This will require additional effort on the part of the faculty member to provide a list of appropriate projects and ideally to engage with multiple project maintainers in advance about the curricular goals of their course. Beyond this, the interaction with the projects may be more "hands-off" as it will not be as possible for the faculty member to be as familiar with each project's specifics.

Engagement with a project can also occur at the program or institution level. The project may be one that can be utilized in a course (e.g. Open Circuits), or by the institution in service of its operations and/or mission (e.g. FD2, LFP, OED). This will require greater time and resource commitments from the faculty member and institution. For example, funding, servers, installation support, customization, and interaction with administration. But, as a benefit, we have observed that students have higher interest in projects that are used at their institution or align with its mission.

7 PROJECT ORGANIZATION CONSIDERATIONS

Education-Oriented H/FOSS projects can be organized in various ways to support different engagement models. For example, some projects, such as OED and to a lesser extent FD2, are designed to accept individual students and up to entire classes of students where the faculty member at the student's institution help set goals for the engagement. Students work as individuals or teams contributing to the project. For OED the focus of the experience is the development of code to move the project forward in its goals set by the project maintainer. The project maintainer interacts directly with individual students as needed to clarify the work needed, to review their contributions and provide feedback, and to incorporate the students' code into the project.

On the other hand, LibreFoodPantry (LFP) expects that faculty members will bring full classes, and that the faculty member will be fully involved in managing their class and its work within the project [42]. Faculty members are encouraged to contact an LFP Coordinating Committee member to discuss their course goals and how to incorporate LFP into their course. Options include forking an LFP project and modifying it for the food pantry at the faculty member's school, participating in the development of an existing LFP project, and starting a new LFP project for their own school. Faculty who have their class participate in an existing LFP project will be given maintainer privileges so that they can approve and merge student contributions to contextualize and improve the timeliness of responses.

8 STUDENT ONBOARDING

As described in Section 1, student and faculty learning curves can present significant challenges to engaging with H/FOSS projects.

Because Education-Oriented H/FOSS projects are constantly engaging new students and faculty, often under tight time constraints, they invest heavily in simplifying the onboarding process.

Education-Oriented H/FOSS projects are thoughtful about how their architectures and the technologies they employ affect student and faculty engagement. For example, LFP has chosen a microservices architecture in part to allow different schools to use different programming languages. FD2 decided on using Vue.js because it integrates into a prior knowledge of HTML and JavaScript more easily than other frameworks. These projects are also careful to clearly articulate the prerequisite knowledge and student background that is essential for engaging in the project. This clarity makes it easier for faculty to make informed choices about which project(s) to engage with. Faculty can decide what students will already know, what will need to be taught, and what students will be expected to learn independently. Some projects (FD2) have gone as far as providing classroom-tested onboarding activities that introduce students to both the project and the technologies necessary for engagement.

Education-oriented H/FOSS projects are willing to invest in high quality onboarding materials to save time and enhance the experience for all involved – students, faculty, and project maintainers. Students typically have less technical experience than contributors to an open source software project, and have a limited time to get up to speed in the project and make a contribution. The faculty maintainers of the project may not have the time to individually help students from institutions outside their own. Faculty members at the student's own institution, depending on their engagement level in the project, may not have the time or experience to provide significant support to the students.

Our Education-Oriented H/FOSS projects provide turn-key development environments that minimize installation and setup for student developers that help get them past the typical problems with inconsistent operating systems and dependencies that can derail student developers. Such development environments, often built with containerization software such as Docker, can also provide complete suites of in-IDE and in-pipeline tools (such as static code analysis) to help students produce clean, readable, and correct code that is more likely to be accepted into the project.

9 MOVING FORWARD

While Education-Oriented H/FOSS projects are an encouraging technique, more information is needed and additional techniques should be tried. This is an ongoing effort where the following items are planned or under consideration:

Data Analysis We are beginning to collect and analyze data on using H/FOSS projects across schools and projects to better understand its impact and possible implementations. This includes the perspectives of the student, professor and project.

H/FOSS Kits As mentioned in Section 2, we are also working on H/FOSS Kits to allow for more authentic teaching of needed concepts and skills using actual projects, rather than "toy" examples, while still providing a repeatable, predictable educational experience. H/FOSS Kits produced from Education-Oriented H/FOSS projects are likely to also play an important role in further reducing the barriers to onboarding students and faculty to these projects.

Guidebooks Our group will be producing guidebooks to help professors, projects and students with effective and realistic engagement between H/FOSS projects and academic learning.

Time Help The time needed by both professors and project maintainers is still significant with Education-Oriented H/FOSS projects. One idea is to utilize student ambassadors. Students with sufficient knowledge of a H/FOSS project will be available to help both professors and students working on the project including installation issues, project resources and technologies, and coding questions. The goal is to have the student ambassador buffer the project maintainers from some of the additional effort. It also serves as a valuable experience for the ambassadors.

How can you move forward in integrating H/FOSS into your curriculum? First, know there is a community available to help including Teaching Open Source [37], The Humanitarian FOSS Project [31], and FOSS2Serve [14] which runs Professors Open Source Software Experience (POSSE) workshops [15] that trains instructors to engage students in HFOSS projects. Second, consider the framework and curricular ideas presented to help define your goals and match them to your resources. Third, know that doing this is a journey where you might start smaller, such as using an existing, Education-Oriented H/FOSS project in a single course and then gradually deepen your engagement over time. Our team and others have found the effort was well worth the rewards for our CS programs and our students.

10 CONCLUSION

We have advocated Education-Oriented H/FOSS projects as an approach to gaining significant curricular benefits while mitigating known challenges. We have presented a framework for understanding H/FOSS engagement in terms of professor involvement, placement of project, how students learn skills and engagement goals. This framework may assist faculty in evaluating the appropriateness of existing Education-Oriented H/FOSS projects for use in their courses and curriculum, or in planning a new project. We described four active projects with characteristics that we believe to be important in Education-Oriented H/FOSS projects. These projects were compared and contrasted to illustrate the framework and give concrete examples of its design considerations and trade-offs. While our work is ongoing, we believe that Education-Oriented H/FOSS projects will make the incorporation of H/FOSS into courses and curricula more approachable, enabling more students, faculty and institutions to realize the substantial benefits of engaging with H/FOSS.

ACKNOWLEDGMENTS

This work was supported under National Science Foundation Grants DUE- 1225738, 1225688, 1225708, 2012966, 2013069, 2012979, 2012999, and 2012990. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of NSF.

REFERENCES

[1] Johathan Beckett. 2021. *The Argument for Open Source.* The Startup. https://medium.com/swlh/the-argument-for-open-source-3035d5aeaddc

[2] S. Beyer. 2008. Predictors of Female and Male Computer Science Students' Grades. *Journal of Women and Minorities in Science and Engineering* 14 (2008), 377–409.

https://doi.org/10.1615/JWomenMinorScienEng.v14.i4.30

[3] Grant Braught, John Maccormick, James Bowring, Quinn Burke, Barbara Cutler, David Goldschmidt, Mukkai Krishnamoorthy, Wesley Turner, Steven Huss-Lederman, Bonnie Mackellar, and Allen Tucker. 2018. A Multi-Institutional Perspective on H/FOSS Projects in the Computing Curriculum. *ACM Trans. Comput. Educ.* 18, 2, Article 7 (July 2018), 31 pages. https://doi.org/10.1145/3145476

[4] Grant Braught and Farhan Siddiqui. 2022. Factors Affecting Project Selection in an Open Source Capstone. In *Proceedings of the 27th ACM Conference on on Innovation and Technology in Computer Science Education Vol. 1* (Dublin, Ireland) *(ITiCSE '22)*. ACM, New York, NY, USA, 358–364. https://doi.org/10.1145/3502718.3524760

[5] Kevin Buffardi, Colleen Robb, and David Rahn. 2017. Tech Startups: Realistic Software Engineering Projects with Interdisciplinary Collaboration. *J. Comput. Sci. Coll.* 32, 4 (April 2017), 93–98. http://dl.acm.org/citation.cfm?id=3055338.3055355

[6] Open Energy Dashboard. 2022. *Open Energy Dashboard*. Open Energy Dashboard. https://openenergydashboard.github.io/

[7] Heidi J. C. Ellis, Gregory W. Hislop, Mel Chua, and Sebastian Dziallas. 2011. How to Involve Students in FOSS Projects. In *Proceedings of the 2011 Frontiers in Education Conference (FIE '11)*. IEEE Computer Society, Washington, DC, USA, T1H–1–1–T1H–6. https://doi.org/10.1109/FIE.2011.6142994

[8] Heidi J. C. Ellis, Gregory W. Hislop, Stoney Jackson, and Lori Postner. 2015. Team Project Experiences in Humanitarian Free and Open Source Software (HFOSS). *Trans. Comput. Educ.* 15, 4, Article 18 (Dec. 2015), 23 pages. https://doi.org/10.1145/2684812

[9] Heidi J. C. Ellis, Gregory W Hislop, S Monisha Pulimood, Becka Morgan, and Ben Coleman. 2015. Software Engineering Learning in HFOSS: A Multi-Institutional Study. *122nd ASEE Annual Conference and Exposition* 26 (2015), 1–12.

[10] Heidi J. C. Ellis, Stoney Jackson, Darci Burdge, Lori Postner, Gregory W. Hislop, and Joanie Diggs. 2014. Learning Within a Professional Environment: Shared Ownership of an HFOSS Project. In *Proceedings of the 15th Annual Conference on Information Technology Education* (Atlanta, Georgia, USA) *(SIGITE '14)*. ACM, New York, NY, USA, 95–100. https://doi.org/10.1145/2656450.2656468

[11] Heidi J. C. Ellis, Ralph A. Morelli, Trishan R. de Lanerolle, Jonathan Damon, and Jonathan Raye. 2007. Can Humanitarian Open-source Software Development Draw New Students to CS?. In *Proceedings of the 38th SIGCSE Technical Symposium on Computer Science Education* (Covington, Kentucky, USA) *(SIGCSE '07)*. ACM, New York, NY, USA, 551–555. https://doi.org/10.1145/1227310.1227495

[12] Heidi J. C. Ellis, Michelle Purcell, and Gregory W. Hislop. 2012. An Approach for Evaluating FOSS Projects for Student Participation. In *Proceedings of the 43rd ACM Technical Symposium on Computer Science Education* (Raleigh, North Carolina, USA) *(SIGCSE '12)*. ACM, New York, NY, USA, 415–420. https://doi.org/10.1145/2157136.2157260

[13] FarmData2. 2022. *The FarmData2 Project*. FarmData2. https://github.com/DickinsonCollege/FarmData2

[14] foss2serve. 2017. *foss2serve*. foss2serve. Retrieved March 30, 2017 from http://foss2serve.org/

[15] foss2serve. 2017. *Professor's Open Source Software Experience (POSSE)*. foss2serve. Retrieved March 30, 2017 from http://foss2serve.org/index.php/POSSE

[16] The Linux Foundation. 2022. The 10th Annual Open Source Jobs Report: Critical Skills, Hiring Trends, and Education. https://linuxfoundation.org/tools/the-10th-annual-open-source-jobs-report/

[17] Edward F Gehringer. 2011. From the manager's perspective: Classroom contributions to open-source projects. In *Frontiers in Education Conference (FIE), 2011*. IEEE, IEEE Computer Society, Los Alamitos, CA, USA, F1E–1.

[18] Mark H. Goadrich, Michael Goldweber, Matthew C. Jadud, Sarah Monisha Pulimood, and Samuel A. Rebelsky. 2019. Civic Engagement Across the Computing Curriculum. In *Proceedings of the 50th ACM Technical Symposium on Computer Science Education, SIGCSE 2019, Minneapolis, MN, USA, February 27 - March 02, 2019*, Elizabeth K. Hawthorne, Manuel A. Pérez-Quiñones, Sarah Heckman, and Jian Zhang (Eds.). ACM, New York, NY, USA, 649–650. https://doi.org/10.1145/3287324.3287335

[19] Swapna S Gokhale, Thérèse Smith, and Robert McCartney. 2012. Integrating open source software into software engineering curriculum: Challenges in selecting projects. In *Proceedings of the First International Workshop on Software Engineering Education Based on Real-World Experiences*. IEEE Press, IEEE Computer Society, Los Alamitos, CA, USA, 9–12.

[20] Mikey Goldweber, Lisa Kaczmarczyk, and Richard Blumenthal. 2019. Computing for the Social Good in Education. *ACM Inroads* 10, 4 (nov 2019), 24–29. https://doi.org/10.1145/3368206

[21] Pawel Grabarczyk, Alma Freiesleben, Amanda Bastrup, and Claus Brabrand. 2022. Computing Educational Programmes with More Women Are More about People & Less about Things. In *Proceedings of the 27th ACM Conference on on Innovation and Technology in Computer Science Education Vol. 1* (Dublin, Ireland) *(ITiCSE '22)*. ACM, New York, NY, USA, 172–178. https://doi.org/10.1145/3502718.3524784

[22] Gregory W. Hislop, Heidi J. C. Ellis, and Ralph A. Morelli. 2009. Evaluating Student Experiences in Developing Software for Humanity. In *Proceedings of the 14th Annual ACM SIGCSE Conference on Innovation and Technology in Computer Science Education* (Paris, France) *(ITiCSE '09)*. ACM, New York, NY, USA, 263–267. https://doi.org/10.1145/1562877.1562959

[23] Gregory W. Hislop, Heidi J. C. Ellis, S. Monisha Pulimood, Becka Morgan, Suzanne Mello-Stark, Ben Coleman, and Cam Macdonell. 2015. A Multi-Institutional Study of Learning via Student Involvement in Humanitarian Free and Open Source Software Projects. In *Proceedings of the Eleventh Annual International Conference on International Computing Education Research* (Omaha, Nebraska, USA) *(ICER '15)*. ACM, New York, NY, USA, 199–206. https://doi.org/10.1145/2787622.2787726

[24] Jaap-Henk Hoepman and Bart Jacobs. 2007. Increased Security through Open Source. *Commun. ACM* 50, 1 (jan 2007), 79–83. https://doi.org/10.1145/1188913.1188921

[25] Fabio Kon, Paulo Meirelles, Nelson Lago, Antonio Terceiro, Christina Chavez, and Manoel Mendonca. 2011. Free and Open Source Software Development and Research: Opportunities for Software Engineering. In *2011 25th Brazilian Symposium on Software Engineering*. IEEE, Champaign, IL, USA, 82–91. https://doi.org/10.1109/SBES.2011.19

[26] LibreFoodPantry. 2022. *Libre Food Pantry*. LibreFoodPantry. https://librefoodpantry.org/

[27] Ralph Morelli, Allen Tucker, Norman Danner, Trishan R. De Lanerolle, Heidi J. C. Ellis, Ozgur Izmirli, Danny Krizanc, and Gary Parker. 2009. Revitalizing Computing Education Through Free and Open Source Software for Humanity. *Commun. ACM* 52, 8 (Aug. 2009), 67–75. https://doi.org/10.1145/1536616.1536635

[28] Becka Morgan and Carlos Jensen. 2014. Lessons Learned from Teaching Open Source Software Development. In *Open Source Software: Mobile Open Source Technologies*, Luis Corral, Alberto Sillitti, Giancarlo Succi, Jelena Vlasenko, and Anthony I. Wasserman (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 133–142.

[29] Christian Murphy, Kevin Buffardi, Josh Dehlinger, Lynn Lambert, and Nanette Veilleux. 2017. Community Engagement with Free and Open Source Software. In *Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education* (Seattle, Washington, USA) *(SIGCSE '17)*. ACM, New York, NY, USA, 669–670. https://doi.org/10.1145/3017680.3017682

[30] OpenCircuits. 2022. *OpenCircuits*. OpenCircuits. https://github.com/OpenCircuits/OpenCircuits/

[31] The Humanitarian FOSS Project. 2017. *The Humanitarian FOSS Project*. The Humanitarian FOSS Project. Retrieved Aug 2022 from http://www.hfoss.org/

[32] Sarah Monisha Pulimood, Kim Pearson, and Diane C. Bates. 2020. Encouraging CS students to compute for social good through collaborative, community-engaged projects. *SIGCAS Comput. Soc.* 49, 1 (2020), 21–22. https://doi.org/10.1145/3447892.3447900

[33] K. Shockey and P.J. Cabrera. 2005. Using open source to enhance learning. In *2005 6th International Conference on Information Technology Based Higher Education and Training*. IEEE, Champaign, IL, USA, F2A/7–F2A12. https://doi.org/10.1109/ITHET.2005.1560267

[34] Therese Mary Smith, Robert McCartney, Swapna S. Gokhale, and Lisa C. Kaczmarczyk. 2014. Selecting Open Source Software Projects to Teach Software Engineering. In *Proceedings of the 45th ACM Technical Symposium on Computer Science Education* (Atlanta, Georgia, USA) *(SIGCSE '14)*. ACM, New York, NY, USA, 397–402. https://doi.org/10.1145/2538862.2538932

[35] Sulayman K Sowe and Ioannis G Stamelos. 2007. Involving software engineering students in open source software projects: Experiences from a pilot study. *Journal of Information Systems Education* 18, 4 (2007), 425.

[36] Diomidis Spinellis. 2021. Why Computing Students Should Contribute to Open Source Software Projects. *Commun. ACM* 64, 7 (June 2021), 36–38. https://doi.org/10.1145/3437254

[37] TeachingOpenSource. 2022. *Teaching Open Source, Instructors and open source communities supporting teaching open source*. TeachingOpenSource. http://teachingopensource.org/

[38] Tidelift. 2022. *The 2022 open source software supply chain survey report*. Tidelift. https://explore.tidelift.com/survey-results/2020-survey

[39] Allen Tucker, Ralph Morelli, and Trishan de Lanerolle. 2011. The humanitarian FOSS project: Goals, activities, and outcomes. In *Global Humanitarian Technology Conference (GHTC), 2011 IEEE*. IEEE, IEEE, Champaign, IL, USA, 98–101.

[40] A. Tucker, R. Morelli, and C. de Silva. 2011. *Software Development: An Open Source Approach*. CRC Press, 9781439812907.

[41] Georg von Krogh and Eric von Hippel. 2006. The Promise of Research on Open Source Software. *Management Science* 52, 7 (2006), 975–983. https://doi.org/10.1287/mnsc.1060.0560

[42] Karl R. Wurst, Christopher Radkowski, Stoney Jackson, Heidi J. C. Ellis, Darci Burdge, and Lori Postner. 2020. *LibreFoodPantry: Developing a Multi-Institutional, Faculty-Led, Humanitarian Free and Open Source Software Community*. ACM, New York, NY, USA, 441–447. https://doi.org/10.1145/3328778.3366929