LibreFoodPantry: Developing a Multi-Institutional, Faculty-Led, Humanitarian Free and Open Source Software Community

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

Stoney Jackson
Heidi J. C. Ellis
Western New England University
Springfield, MA USA
stoney.jackson@wne.edu
heidi.ellis@wne.edu

Darci Burdge
Lori Postner
Nassau Community College
Garden City, NY USA
darci.burdge@ncc.edu
lori.postner@ncc.edu

ABSTRACT

Engaging students in humanitarian free and open source software (HFOSS) projects allows them to gain real-world software development skills while helping society. Participating in an existing HFOSS project, although ripe with learning opportunities, presents a number of hurdles for faculty and students. An alternative to joining an existing HFOSS project community is to participate in a faculty-led HFOSS project. These projects provide the instructor with more control over the learning environment, but often lack an active community outside of the classroom. This paper describes a multi-institutional effort to engage a community of developers in creating humanitarian open source projects to support their on-campus food pantries. Food insecurity on campus has become a national concern and many institutions have, or are starting, food pantries to support the student, staff, and faculty community.

Starting a faculty-led HFOSS project involves making decisions not only about the features of the project but also about community norms, tool choices, project development workflow, and inter-institution cooperation. This paper provides an overview of the creation of LibreFoodPantry, a community who is developing a suite of projects that support on-campus food pantries. It describes instances of using LibreFoodPantry's projects in various classroom settings, the lessons learned from these experiences, and the resulting discussions and decisions made by the LibreFoodPantry Coordinating Committee. This process has led to a community dedicated to easing the on-ramp for faculty who want to help their students contribute to an HFOSS project.

KEYWORDS

Open Source, Computing for Social Good, Software Development, Project Management

ACM Reference format:

Karl Wurst, Christopher Radkowski, Stoney Jackson, Heidi Ellis, Darci Burdge and Lori Postner. 2020. LibreFoodPantry: Developing a Multi-Institutional, Faculty-Led, Humanitarian Free and Open Source Software Community. In *Proceedings of 2020 ACM Technical Symposium on Computer Science Education (SIGCSE '20), March 11-14, 2020, Portland, OR, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3328778.3366929

SIGCSE '20, March 11-14, 2020, Portland, OR, USA

ACM 978-1-4503-6793-6/20/03…$15.00
https://doi.org/10.1145/3328778.3366929

1 INTRODUCTION

Over the past decade, a growing community of faculty have involved their students in existing humanitarian free and open source software (HFOSS) projects. HFOSS projects address a societal need such as open medical records systems, disaster management, microfinance, or education. The ability to help others is highly valued among all students in computing [21] and an emphasis on how computing can positively affect one's community is especially engaging for women [17]. HFOSS projects can provide learning environments that allow students to see how their technical expertise can help others.

The focus of previous HFOSS work has been two-fold: expanding the community by preparing faculty to support student learning in HFOSS [2, 14, 15] and researching the impact of student participation in HFOSS [1, 7, 12]. Outcomes of this work indicate that students report a perceived learning of software engineering, technical, and professional skills [4, 7, 8, 11].

Despite the positive impact for students, finding an appropriate existing HFOSS project to use in a class setting is one of the biggest factors that deters faculty from engaging their students with HFOSS projects [14]. Faculty who have facilitated student participation in HFOSS are often teaching senior level software engineering or capstone classes, where students have enough experience to learn new languages and frameworks as part of the course. Integrating HFOSS into lower level courses is much more challenging, as inexperienced students can be easily overwhelmed. The language, scale, platform/frameworks, domain knowledge, project community and culture, contact person, project timeline and

roadmap, etc. are all factors that influence if a specific project is appropriate for classroom adoption [5, 14, 16].

If an appropriate existing HFOSS project is found, it is likely that the project may not be prepared to receive a large influx of new contributors that intend to work as one or more teams. The instructor may wonder, "*Where do they fit in*?", "*What should they work on?*" and "*Who can answer their questions and review their work in a timely fashion?*" Except in rare cases, the instructor can't answer any of these questions. Although most open source projects provide instructions for individuals to contribute to their project, they lack instructions for how one or more teams may contribute to the project. They may assume that any team that would want to contribute to their project already knows how to work as a team using modern development practices. This is often not true of classes of students who will be engaging in these projects. To alleviate some of these hurdles, a subgroup of the HFOSS faculty community are shifting their focus to starting faculty-led HFOSS projects designed to address local, institutional needs.

Projects developed for a single client often lack a rich and robust community. But these community interactions are one of the key benefits of involving students in HFOSS, providing students and faculty with professional interactions beyond their local settings. With this in mind, LibreFoodPantry (LFP), a community who is developing a suite of HFOSS projects designed to help the volunteers at three different institutions run their on-campus food pantries, was created. Students interact with a real-world client, the food pantry, with one another across institutions, and learn about open source tools and processes, all while helping their campus communities. As a multi-institutional effort, the faculty and students involved are focused on defining LFP as a community where classes are welcome to join and leave the project as necessary based upon the academic needs of the course. The issue of how teams work is addressed by defining a clear onboarding process for a *shop* of developers. A shop mirrors the structure of a class with the instructor playing the role of a shop manager and the students playing the role of shop developers.

2 RELATED WORK

Building a faculty community around an HFOSS project leverages earlier work where an HFOSS project was shared among multiple academic institutions and open source developers [6]. In this case, the project originated in the GNOME community and the academic handoff failed due to a combination of the project technologies having a significant learning curve, a lack of instructor time, and no existing active community around the project.

Although having students involved in existing HFOSS projects provides a wealth of learning opportunities, research has demonstrated that there are a number of additional hurdles to adoption [10, 18, 19, 20] including finding the right project, time to develop curricular materials, and teaching a class where using HFOSS seems appropriate [14]. Once a faculty member is involved with a project, they may encounter problems with the academic calendar not mapping to the community timeline, need for issues that are appropriate for the students' level of education, as well as students' uncertainty about their abilities to contribute to a large HFOSS project. As an academic community, LFP is sensitive to the needs of faculty and their students. Faculty are steering the direction of the project and providing curricular support, thereby reducing the barriers to entry and adoption. Unlike the CO-FOSS model [22] where the project is developed by a class, handed over to the client at the end of the term, and then maintained by a local company, LibreFoodPantry projects are envisioned as ongoing and evolving across academic terms, having multi-institutional participation, and being used by multiple clients.

In a multi-institutional project, faculty from different institutions can support each other in project management responsibilities. Students from different institutions can help each other creating richer interactions. Projects can share clients and users at different institutions. Over time, we expect a multi-institutional effort to generate a robust community that will help sustain the project over the long term and provide students, faculty, clients, and users with a larger support network with rewarding interactions.

Having real and involved clients and users are extremely valuable to any project. Real clients and users help to establish requirements, answer questions, and provide valuable feedback on work as a product is being developed. In a class project, real clients and users also provide students with additional motivation as they realize that someone is genuinely interested in their work. Another advantage of LFP projects is that most classes can easily find an operating food pantry. Their school may have a food pantry on campus, their campus may be considering opening a food pantry, or there is likely a food pantry in their local area. In cases where a local client cannot be found, clients from other areas or institutions may serve as a client for a class from afar [22].

3 BACKGROUND

3.1 History

In November 2015, Nassau Community College (NCC), part of the State University of New York (SUNY) system, opened the NEST on-campus food pantry. Two CS professors began a multi-year conversation about how the food pantry operates, its biggest technology needs, and how a class of CS students could help.

Inspired by the effort at NCC, as part of a pre-symposium event at SIGCSE 2018, a group of faculty evaluated existing software for managing food pantries and food banks. While numerous projects exist, none were suitable for use in a class for various reasons, such as being closed source, lack of community, licensing issues, level of maturity, etc.

In Spring 2018, Western New England University (WNE) announced plans to open the BEAR Necessities Market on-campus food pantry. In Fall 2018 two CS & IT professors approached the pantry organizers to discuss having CS majors develop an HFOSS project for their pantry in a new senior capstone course.

In Fall 2018, Worcester State University (WSU) announced that it would open Thea's Pantry. A CS professor contacted the pantry organizers to offer the services of his software development capstone course to develop software to help manage the pantry.

These faculty are part of a group active in encouraging faculty to incorporate HFOSS participation into their curricula. During a research meeting in January 2019, they brainstormed the idea to support faculty-led HFOSS projects which would provide real software development projects, but in a more academically friendly environment. Realizing that a number of their institutions either had a food pantry or were about to open a food pantry, they decided to create a suite of HFOSS projects to provide food pantries with free and open source software, while providing students and faculty with a project that they could contribute to.

3.2 Pilot

During the Spring 2019 semester three institutions piloted using LFP's projects in a single course per institution. Prior to the start of the semester, all instructors agreed to use GitHub to host the LFP repositories and to use Slack to facilitate discussion within each course and across institutions. They also agreed that students should have some control of the development of LFP but did not make any decisions as to how that would work.

At NCC, LFP was used in the Mobile Application Development course, a third or fourth semester course for CS majors. Students first learn Android basics and become familiar with the development environment and application lifecycle. Then students learn about the previous work on the NEST project and its technical needs based upon meetings with the volunteers who manage the food pantry. They brainstormed ways to improve the existing project and worked both individually and in teams to implement their ideas. Students were introduced to git and learned to work collaboratively using the git workflow developed for the project.

At WNE, LFP was used in a one-semester Computer Science Senior Capstone project in the last semester of students' senior year. Five 3-4 member teams worked to design and develop a single web-based food ordering system using a Scrum-based development process. Students chose the technological platform and implemented the infrastructure for further development.

At WSU, LFP was used in a one-semester Software Development Capstone course in the last semester of students' senior year. Two 5-member teams (one per section) worked on a LFP project to design a web-based guest intake application. In addition, a cross-section sub-team developed an API service for other applications to use the USDA's FoodKeeper data, with the NCC class as a client.

3.3 Mid-Semester Discussions

The faculty met weekly throughout the semester, providing updates on the status of their classes, and discussing approaches to having their classes work within the project. By March technical issues of student communication and workflow led to discussions on the larger issues of culture, community, decision-making, governance, and welcoming new faculty and classes into the project. Decisions were made about how to make the project more inviting to outsiders, how the group could provide transparency by making meeting minutes and communications public, ways to standardize workflow and documentation, and creating a community repository to house a governance model and code of conduct.

3.4 Post-Semester Retreat

In June 2019, the LFP Coordinating Committee held a retreat to reflect on the experiences from the pilot courses, establish community norms, and develop a cohesive view of the product(s) that LFP intends to build and support. The faculty who taught the pilot courses, two additional faculty members who planned to use LFP in Fall courses, and a WSU senior researching tools to support agile and open-source software development, attended.

In an agile-style retrospective, the participants identified what worked well and was important to the success of their courses, what did not work well and needed improvement, and ideas for improving those things that did not work well. During the retreat, the participants drafted several important documents as the foundation of the LFP community and reaffirmed the open source license that all projects were using for source code. In addition to these basic documents, they began to sketch out the concept of a development "shop" and a common workflow by which development shops can contribute work as one or more teams. The retrospective is discussed in section 4.

4 CHALLENGES AND SOLUTIONS

When starting LFP the group didn't realize the number of decisions that they would have to make to start a faculty-led HFOSS community. It is difficult to make long-term decisions that allow for future participation among a larger group of academics when working across institutional settings. The following sections describe the major areas that required decisions and some of the discussion and rationale that went into them.

4.1 License

One of the first decisions made, even before the pilot courses, was which license we would use for the source code of LFP projects. Without an initial license, the work completed in the pilots would be unlicensed. This means that no one could do much of anything with the code without permission of every contributor of the project (because each contributor holds the copyrights for their individual contribution). Also, relicensing a project would be extremely difficult and becomes more difficult as more developers contribute work to the project.

A free and open source project is free and open source because of its license. Because there are many different open source licenses it is important to choose one that fits the goals, values, and necessities of the project. We wanted LFP projects to be free forever. That is, however its projects evolve in the future, everyone will forever have the same freedoms to use, redistribute, and change the projects as they do today. To achieve this, we chose to license all source code under the GNU Public License version 3 (GPLv3).

4.2 Community and Governance Structure

LFP's community and governance structure is designed to address several goals. First, create a community that is welcoming and inclusive where faculty and students of all backgrounds feel comfortable participating. A mission statement, a vision statement, a code of conduct, and contributor guidelines were drafted to reflect the community's goals. These are reflective of agile values and FOSSisms and intend to help guide community interactions and decisions in the future.

Second, quickly onboard and embed instructors, not just their students, into the LFP community. To this end we request that instructors join the Coordinating Committee and attend and participate in its weekly meetings. This gives instructors immediate and regular access to other faculty who are or have worked on an LFP project. In these meetings they can quickly orient themselves to the community's norms and practices as well as the community's current directions and how their course might contribute. This committee becomes a multi-institutional support network that the instructor can go to when they need help. In addition, the minutes of the Coordinating Committee meetings are public, as is the issue tracker they use to coordinate, giving new members a history of the project and its decisions.

Third, create a governance framework that would give instructors the power to shape the tools and the project to suit the needs of their class. In exchange for joining and participating in the LFP Coordinating Committee, we give the instructor the role of shop manager with elevated privileges over the project(s) their class/shop will be working on. This allows the Coordinating Committee to quickly gain trust in shop managers, it gives new shop managers instant access to other faculty who have used or are using LFP projects in a class, and allows shop managers to help guide the direction of the project so that it best suits their academic needs and their clients' needs.

4.3 Workflow

The workflow is the process by which an individual or a group organizes and coordinates to make contributions to a project. We needed a workflow that would allow more than one shop/course with one or more teams of developers to work on one or more projects at a time. We also wanted the workflow to be flexible enough that an instructor can customize it to fit the needs of their course. In addition, we also want it to be possible for individuals outside any shop to be able to contribute to the project as well.

LFP set up organizations on the two leading repository hosting sites, GitHub and GitLab. GitHub was the initial choice because it is widely known among students and faculty and is widely used by open source projects. During the Spring 2019 term GitHub was used by all three institutions to host code and issues. We based the workflow on GitHub Flow [9]. In GitHub Flow, an individual contributes a change by making a copy of the original upstream project in GitHub (a fork), makes changes to the fork, and then offers these changes back to the upstream project through a pull request. All the while, contributors use the issue tracker in the upstream project to coordinate their efforts. We adapted this workflow for shops as follows: each team creates a team fork of the upstream project and the shop manager is given privileges over the upstream project to review and merge pull requests from teams.

By the end of the semester, two problems became clear. First, teams had limited privileges over the issue tracker in the upstream project and therefore could not help refine issues. Also, this constrained teams' ability to coordinate. Teams either used the issue tracker in their forks, which causes confusion as there are now two issue trackers to manage; or they used some other mechanism outside the project, which the instructor could not easily monitor to assist teams. Second, with teams having separate forks, the instructor and the teams were less aware of the activities of other teams.

To address these issues, we wanted to have each shop (course) have a single shop fork that all of its teams would use. This way the instructor and the teams would be aware of all activity within the shop. Also, we wanted the shop developers to have more privileges over the issue tracker in the upstream project so that they can participate more in the refining and coordinating activities, and could use the issue tracker to help coordinate efforts within the team. GitHub could not support this model.

Based upon the course experiences from Spring 2019, summer research assistants from all institutions investigated the use of GitLab as opposed to GitHub. During this investigation of GitHub versus GitLab, three versions of these tools were compared: GitHub Free, GitLab Free, and GitLab Gold. A feature table was created that showed the similarities and differences of these platforms and how the tools directly compared to each other. Most of the features in GitHub are available in GitLab Gold, which also has a large number of additional features that could be useful in developing and maintaining the LFP projects.

In addition to the feature comparison, our proposed workflow for LFP projects was modeled and tested on each platform. We found that the basic proposed git workflow we would use for shops worked similarly on all platforms. Along with testing the workflow, other platform features were tested including project permission levels and project boards for coordinating issues and work. We found that GitLab offered better permission levels for managing our different types of users and developers. We also found that GitLab offered better project board systems for managing and coordinating work. It was ultimately decided that the suite of LFP projects would switch to using GitLab Gold (which is free to open source projects) to host our repositories.

4.4 Communication

Finding a mechanism for students and faculty to communicate within a course as well as across institutions is important to keep the history of the project for future students and faculty. Students (and faculty!) often do not default to working in the open, and

communication that does not originate on public channels rarely becomes public after the fact.

During the Spring 2019 term all institutions used Slack with channels for each individual institution as well as a general LFP channel. There was little inter-institutional communication and Slack was not widely used within courses either. Some general issues with Slack are: it isn't open source (although widely used by open source projects), it isn't accessible, potential contributors must request access to a channel (lurking is not an option), and students aren't already using Slack in their lives, thus it became another place they had to remember to check.

The Coordinating Committee and student researchers investigated alternative communications platforms. Requirements included being open source, accessible for those with vision or hearing impairments, hosted, and allowing anyone to join/lurk without approval. After looking at several options two were selected for piloting: Gitter and Discord. Neither satisfied all our requirements.

While Gitter is open source and hosted, (and accessible through an IRC bridge), its main problem is that the only way to join Gitter is through an existing GitHub, GitLab or Twitter account. Because of the lack of granularity in both GitHub and GitLab's authentication systems, Gitter requires access to all of a user's repositories. While Gitter promises not to use that access, the warnings are disturbing enough that we believe that many would decline to use Gitter.

While Discord is not open source and not accessible, it provides a number of positive features such as audio, video and screen sharing, and many students are already users. The biggest concern with Discord is that it is marketed as a gamer's tool with many references to gamer culture. This may evoke a negative response in some students. It is possible to turn off some of the off-putting features, but not all of them. The decision to switch to Discord is accompanied by a plan to actively mitigate possible student concerns by telling them that we are aware of its reputation, have an actively enforced code of conduct, and will work to create a safe and inclusive space. Research on students' perceptions of the tool will be conducted as well.

4.5 Reviewing Changes

Before accepting changes into a project, it's important that they are carefully reviewed to ensure that they do not break the existing system, implement what they purport to, are well designed, and meet the project's quality standards. In a typical open-source project this review process may take a long time as the maintainers may be working on the project in their spare time. This is magnified if we have a large number of students submitting changes to the project. The first fix to this challenge is to have the shop manager (instructor) review students' work. The advantage is that the shop manager understands their course's need for timely reviews.

However, this brings a new challenge, how is the instructor supposed to complete so many thorough reviews? An instructor may want to have students review other students' work. Regardless of who performs the review, it's important to automate as much of the review process as possible.

The first technique to employ is automated testing. Shop developers (students) must write automated tests to accompany their code as is standard practice in modern agile development. These tests, along with all previous tests, can be run regularly to check that new changes work properly and do not damage existing functionality.

The next challenge is how does the reviewer know if the changes under review will integrate properly with possible new changes in the main development branch. The reviewer must merge the proposed changes with the main branch and run the automated tests. This too can be automated and is part of the practice of continuous integration (CI). In CI, every time a developer makes new changes, they are merged with the main branch into a temporary branch, the automated tests are run on this merged branch and the results are reported. The developer can then adjust their work appropriately until their work successfully merges with the main branch. Similarly, continuous deployment (CD) can be used to automatically test if the merged copy can be successfully installed and run within a known environment. If successful, this deployed instance can be used by the reviewer to quickly manually test new and existing features or to demo new features to a client!

Another important check that a reviewer must perform is to confirm that the work being offered was created by the author offering the work or was licensed in a way that is compatible with the project's license. This is very challenging and onerous for a project to check. Increasingly, modern practice is to have developers sign-off on a Developer Certificate of Origin [3] for each of their commits. The developer's sign-off asserts that they know where the work came from, that they have the right to contribute the work to the project, and that the work is licensed with a compatible license. This may seem a lot to ask of a student, but brings the concepts of copyrights, licensing, and plagiarism to the forefront of the discussion. The reviewer must check that each commit has a sign-off. This check can and should be automated.

Other checks can be automated to ensure that there is consistency in the review process and to ease the reviewing burden for shop managers. The shop manager or other shop members can focus on reviewing essential but hard to automate characteristics like verifying that the new automated tests actually test the new changes, and that the new changes and tests are well designed.

4.6 Story Mapping

Currently LFP is a suite of projects with three clients with differing requirements, and at least 5 instructors working on applications. As more institutions and instructors participate in the project this will increase. Can we unify these differing requirements into a single, but flexible project? Students and faculty are unfamiliar with the food pantry domain and a client's needs, and clients may have a difficult time articulating their requirements or envisioning ways in which an application can help them beyond what they currently do at their own food pantry. Left to their own devices, students want to work on technical stories, as they are more familiar with and interested in technical issues. Unfortunately, a deep dive into a technical issue may result in code that does not benefit the client.

The instructors at the founding institutions are the most familiar with the features that their own food pantries are requesting, but all of the instructors need to have a broader view of these features, both to help determine which features should be included in the future unified project and to be able to suggest new features that their clients may find useful but have not yet thought of.

With an eye toward eventually having the current apps converge to a single app that has features that can be enabled or not based on customer needs, the Coordinating Committee spent much of a day in a story mapping [13] session. The goal was to share current and future needs of our respective customers, develop consistent terminology, and have all the instructors be aware of all possible requirements and features. The story mapping session began with the instructors enumerating user roles and how each user might interact with the proposed systems. Tasks were laid out in a rough timeline to show how a particular user might interact with the system. The user(s) who might perform each task were listed above each task, and details or alternate ways of doing that task were placed below.

This story map is posted on the project's website as a record of the group's current vision for the product. The plan is to walk each client through the story map to get feedback about how the features will meet their needs, and to familiarize them with features they may not have considered. These client walk-throughs will likely result in new feature requests that will be discussed and incorporated in a future story mapping session.

5 CONCLUSION

The LibreFoodPantry community continues to tackle issues as we prepare for our fall courses. The multi-institutional community has provided momentum across campuses, as well as allowed us to think deeply about issues that will impact the suite of projects. The support system developed by the Coordinating Committee provides a structure for new faculty, courses and institutions to become involved with the community and its projects in a way that we have not encountered before. With a wide variety of institutions, courses, and faculty backgrounds working together, we are making it easier for new faculty and students because we understand the difficulties that may arise and are continually modifying what we do. Because the community consists of multiple faculty, at multiple institutions using LFP in courses across semesters, this model is sustainable because the community will continue to grow as other institutions participate.

We believe that engaging students in an HFOSS project that helps their own, or other institution, will positively impact all students, but especially women and traditionally underserved populations.

6 FUTURE PLANS

The LibreFoodPantry community is in the early stages of developing food pantry applications and much work remains to be done. Students at each of the three institutions will interact with LFP in a variety of ways during the fall and spring terms. Faculty at NCC will continue to develop a better understanding of the needs of their campus food pantry by meeting with volunteers who run the pantry and to develop activities to introduce students to the project. Students will learn new communication models and tools, encouraging inter-institutional communication, as well as how to contribute to LFP using the agreed upon workflow. At WNE LFP will be used in two different courses, a human-computer interaction course and a software engineering course. During the fall term teams of students will select different sections of the LFP story map to work on. They will expand upon the existing ideas, perform client validations, develop interaction models, and begin to sketch out an appropriate interface. This work will inform the spring term software engineering course. At WSU the capstone course which utilizes LFP is offered only in the spring term. One student will be working in the Fall to refine requirements, set up the build and deploy infrastructure, and the high-level architecture design. The student will evaluate the onboarding process, community tools, workflow, and associated documentation developed as part of LFP.

The growth of the community is at the forefront of our efforts. We will continue to develop documentation to ease the onramp for faculty at other institutions who want their classes to participate in LFP. Documentation for onboarding students, contribution workflow, and assessment are of primary interest. Much of the documentation for student participation will be developed as part of the initial effort, but we understand that an influx of students from different institutions and different courses is likely to require a refinement of these documents. It is also hoped that individual developers, who may or may not be students, will want to contribute, providing the community with yet another perspective. For a community to grow, it is important that the community be open and welcoming to all, a tenet that always guides our decision-making process. Sometimes those decisions are easy and at other times much more difficult. Our decision regarding a communication tool is an example of the latter. We carefully considered our technical needs and settled on a tool, but were concerned that aspects of the tool may have a negative impact on student impressions of CS. Our decision was to turn this into a research question and are especially interested in knowing the impact on underserved populations. We are also interested in exploring whether students have the same rewarding experience when contributing to LFP as they do when contributing to a traditional HFOSS project.

ACKNOWLEDGMENTS

This material is based on work supported by the National Science Foundation under Grant Nos. DUE-1525039, DUE-1524877, and DUE-1524898. Additional support provided by an Aisiku 2019 Undergraduate Summer Research Fellowship from the Worcester State University Aisiku STEM Center. Thank you to GitLab for a free GitLab Gold license through their GitLab Open Source Program. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation (NSF).

REFERENCES

[1] 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. DOI: https://doi.org/10.1145/3145476

[2] Darci Burdge, Lori Postner, Heidi J. C. Ellis, and Gregory W. Hislop. 2014. Preparing for student participation in HFOSS projects: FOSS tools and techniques. J. Comput. Sci. Coll. 29, 3 (January 2014), 74-75.

[3] Developer Certificate of Origin, https://developercertificate.org/, accessed 2019-08-19

[4] Heidi J.C. Ellis, Gregory W. Hislop, Josephine Rodriguez, and Ralph A. Morelli. 2012. Student Software Engineering Learning via Participation in Humanitarian FOSS Projects. 119th Annual ASEE Conference and Exhibition (2012).

[5] 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 (SIGCSE '12). ACM, New York, NY, USA, 415-420.

[6] 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 (SIGITE '14). ACM, New York, NY, USA, 95-100. DOI: https://doi.org/10.1145/2656450.2656468

[7] 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 (December 2015), 23 pages. DOI=http://dx.doi.org/10.1145/2684812

[8] Heidi J.C. Ellis, Gregory W. Hislop, Monisha S. Pulimood, Becka Morgan, and Ben Coleman. 2015. Software Engineering Learning in HFOSS: A Multi-Institutional Study. 122nd Association for Engineering Education Annual Conference and Exposition (2015).

[9] GitHub Flow, https://guides.github.com/introduction/flow/, accessed 2019-08-18

[10] Christoph Hannebauer, Matthias Book, and Volker Gruhn. 2014. An exploratory study of contribution barriers experienced by newcomers to open source software projects. In Proceedings of the 1st International Workshop on CrowdSourcing in Software Engineering (CSI-SE 2014). ACM, New York, NY, USA, 11-14. DOI=http://dx.doi.org/10.1145/2593728.2593732

[11] 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 (ICER '15). ACM, New York, NY, USA, 199-206. DOI: https://doi.org/10.1145/2787622.2787726

[12] Gregory W. Hislop, Heidi J. C. Ellis, and Herman Jackson. 2018. Student contribution to HFOSS: challenges and opportunities. J. Comput. Sci. Coll. 33, 6 (June 2018), 181-182.

[13] Jeff Patton, User Story Mapping: Discover the Whole Story, Build the Right Product, O'Reilly Media, 2014.

[14] Lori Postner, Heidi J.C. Ellis, and Gregory W. Hislop. 2018. A Survey of Instructors' Experiences Supporting StudentLearning using HFOSS Projects. In Proceedings of the 49th ACM Technical Symposium on Computer Science Education (SIGCSE '18). ACM, New York, NY, USA, 203-208. DOI: https://doi.org/10.1145/3159450.3159524

[15] Lori Postner, Darci Burdge, Heidi J. C. Ellis, Stoney Jackson, and Gregory W. Hislop. 2019. Impact of HFOSS on Education on Instructors. In Proceedings of the 2019 ACM Conference on Innovation and Technology in Computer Science Education (ITiCSE '19). ACM, New York, NY, USA, 285-291. DOI: https://doi.org/10.1145/3304221.3319765

[16] Michelle Purcell, Heidi J. C. Ellis, and Gregory W. Hislop. 2013. An approach for evaluating open source projects for student participation. J. Comput. Sci. Coll. 28, 6 (June 2013), 199-200.

[17] L. J. Sax, K. Lehman, J. A. Jacobs, A. Kanny, G. Lim, L. N. Paulson, and H. Zimmerman. Anatomy of an Enduring Gender Gap: The Evolution of Women's Participation in Computer Science.

[18] Igor Steinmacher, A. P. Chaves, T. U. Conte and M. A. Gerosa, "Preliminary Empirical Identification of Barriers Faced by Newcomers to Open Source Software Projects," 2014 Brazilian Symposium on Software Engineering, Maceio, 2014, pp. 51-60. doi: 10.1109/SBES.2014.9

[19] Igor Steinmacher, Tayana Conte, Marco Aurélio Gerosa, and David Redmiles. 2015. Social Barriers Faced by Newcomers Placing Their First Contribution in Open Source Software Projects. In Proceedings of the 18th ACM Conference on Computer Supported Cooperative Work & Social Computing (CSCW '15). ACM, New York, NY, USA, 1379-1392. DOI: https://doi.org/10.1145/2675133.2675215

[20] Igor Steinmacher, Tayana Uchoa Conte, Christoph Treude, and Marco Aurélio Gerosa. 2016. Overcoming open source project entry barriers with a portal for newcomers. In Proceedings of the 38th International Conference on Software Engineering (ICSE '16). ACM, New York, NY, USA, 273-284. DOI: https://doi.org/10.1145/2884781.2884806

[21] Burçin Tamer, 2018. Helping Others is the Highest Rated Career Value for Both Undergraduate and Graduate Students in Computing. Computing Research News, 30, 10 (November 2018). https://cra.org/crn/2018/11/helping-others-is-the-highest-rated-career-value-for-both-undergraduate-and-graduate-students-in-computing/

[22] Allen Tucker. (2019). Client-Centered Software Development: The CO-FOSS Approach. 10.1201/9780429506468.