Revitalizing Computing Education Through Free and Open Source Software for Humanity

By Ralph Morelli, Allen Tucker, Norman Danner, Trishan R. de Lanerolle, Heidi J.C. Ellis, Ozgur Izmirli, Danny Krizanc, and Gary Parker

The humanitarian focus of socially useful projects promises motivation for community-minded undergraduates in and out of computer science.

What if undergraduate students saw computer science as, in part, a discipline that designed and built free software to help one's friends and neighbors in need? Would that bring more of them in the front door of academic computing departments? What sort of curricular and pedagogical changes would be required to support such opportunities for these students? Would these changes improve and help revitalize computing curricula and enrollments throughout the U.S.?

The Humanitarian Free and Open Source Software (HFOSS) Project is addressing these questions. The goal is to help revitalize U.S. undergraduate computing education by engaging students in developing FOSS that benefits humanity. What started as an independent study by two undergraduates in 2006, the Project today includes students from several U.S. colleges and universities engaged in variety of FOSS development projects, both global and local. Here, we provide an overview of the Project, along with some of the lessons learned and the challenges that remain. Our experience over the past three-and-a-half years suggests that engaging students in building FOSS that serves society is a positive step toward strengthening undergraduate computing education.

The Project has been supported since September 2007 by a National Science Foundation CPATH[a] grant, aiming in part to build a collaborative community of individuals from multiple educational institutions, diverse computing and IT organizations, and non-profit social-service agencies to support undergraduates in the development of socially useful FOSS. In general, the Project aims to answer whether getting students involved in humanitarian FOSS indeed also helps revitalize undergraduate computing education.

The Project's inspiration comes from the Sahana project, a FOSS disaster-management system that was developed by a group of Sri Lankan volunteers in the aftermath of the December 2004 Asian tsunami. We began working with Sahana in January 2006 after the author Trishan de Lanerolle learned about it during a visit to Colombo, Sri Lanka (see the sidebar “Five HFOSS Software Projects”). That spring, two Trinity College students installed Sahana on an Apache server and began exploring its LAMP architecture (Linux/Apache/MySQL/PHP) as part of an independent-study course. They worked with the code and seemed to enjoy the opportunity and challenge of being engaged in a real-world software project (as opposed to a class exercise). That summer, with support for a summer research student and in collaboration with community-minded volunteers from Accenture Corporation (http://accenture.com), we developed a volunteer-management module that was eventually accepted into Sahana’s code base. Thus began an ongoing collaboration with the Sahana community.

Our initial experience with Sahana dovetailed with two ideas outlined by former ACM president David Patterson in his “President’s Letter” columns in *Communications.* In “Rescuing Our Families, Our Neighbors, and Ourselves” (November 2005), he suggested it might be the civic duty of computing professionals to be more involved in helping their communities recover from natural disasters and, by doing so simultaneously help the profession [5]. In “Computer Science Education in the 21st Century” (March 2006), he explored the disconnect between how programming is taught in the classroom and how cutting-edge software is written in industry, urging educators to involve themselves in the open source movement [6].

The call to build open source software to help our neighbors resonated with our Sahana experience, suggesting that a project organized around that theme might yield beneficial outcomes for undergraduate computing, including:

* Giving computing students experience with the open source development process in a real-world practitioner environment;
* Letting them see first-hand the importance of software-engineering principles;
* They use their programming and problem-solving skills to contribute to the expanding volunteerism movement that characterizes many of today’s college campuses;
* Making it possible for them to gain firsthand contact with IT professionals in the computing industry;
* Enabling computing faculty to experiment with a variety of approaches for incorporating FOSS into the curriculum; and
* Inviting all participants—faculty, students, IT professionals, and the humanitarian community—to join in a mutually beneficial educational and social enterprise.

Problems that beset undergraduate computing education in the U.S. include sagging enrollment, out-of-date curricula, changing demographics, and rapidly evolving technologies. While they are most closely associated with the academic computing discipline, these problems are also associated with a number of myths and misconceptions that extend well beyond the academy to society in general: computer science is nothing but coding; computing students are geeks; programming is an isolating activity; and computing jobs are being outsourced to Asia and Eastern Europe.

These problems and myths cannot be addressed within the academy alone. Rather, what's needed is a sustained effort involving a broad coalition of computing educators and industry professionals. Only that kind of effort can change false perceptions about computing in the larger society. The effort also requires substantial support from the computing industry, which stands to benefit from a revitalized computing curriculum. It also may require the kind of infrastructure and publicity one finds in other communities (such as Teach for America and Habitat for Humanity) that attempt to mobilize students and others to take on real-world projects for the social good. The HFOSS project aims to build a new community and explore its potential to change undergraduate computing.

Serve Society

While FOSS applications run the gamut of computer software, HFOSS, as we define it, is software that aims to serve society in some direct way. This deliberately broad definition is meant to be inclusive of a wide range of socially beneficial projects and activities.[b] To date, the HFOSS Project has not had to face the question of where to draw the line between humanitarian and non-humanitarian FOSS. As a practical measure we use the guideline that any software artifact the Project creates must intrinsically benefit a nonprofit organization pursuing some kind of public-service mission.

As described in [2] by Chopra and Dexter the free-software movement has roots that go back 60 years to the beginning of the computer age when sharing programming ideas and code was the norm. The modern free-software movement began in 1983 when Richard Stallman defined "free software" as the freedom to use, study, copy, change, and redistribute software "so that the whole community benefits" (http://www.gnu.org/philosophy/free-sw.html).

Following the spectacular success of the GNU/Linux project (http://www.gnu.org/), the free-software movement has grown in scope and importance. An April 2008 study by the Standish Group (http://www.standishgroup.com/) estimated that open source software costs the software industry $60 billion in potential annual revenue [9]. SourceForge (http://sourceforge.net), the primary open-source hosting site, lists more than 180,000 projects and 1.7 million registered users worldwide**.** Many top software and Internet–related companies, including Dell, Google, Hewlett-Packard, IBM, Intel, and Microsoft, support the FOSS model in one way or another. According to an August 2008 Linux.com article, students are beginning to join open source projects as a way to gain relevant work experience needed for many entry-level computing positions (http://www.linux.com/archive/feature/143415).

The free-software movement is characterized by the way it distributes its products. The GNU General Public License (GPL) was the first of many free-software licenses stipulating how the software can be freely used and shared. As Stallman said, software freedom, in this sense, is "a matter of liberty, not price"; it is free as in free speech and not (necessarily) as in free beer. The free-software philosophy is supported and promoted by the Free Software Foundation (http://www.fsf.org).

The free-software movement is also characterized by an open development process, a highly distributed, nonhierarchical, peer-based activity. The FOSS approach stands in sharp contrast to the top-down, hierarchical, legacy-based model of traditional commercial software development. This distinction is often exemplified by the difference between how Linux and Microsoft Windows were developed. FOSS programmers collaborate in loosely organized communities, freely working on the projects and problems that are of most interest to them. The FOSS development process is also closely tied to the user community and marked by frequent releases closely monitored and tested by end users. To use a metaphor coined by Eric Raymond, author of *The Cathedral and The Bazaar,* [7] the free software development process resembles a “babbling bazaar,” unlike the “cathedral” model historically employed in commercial software development [7].

The free-software movement split into two competing philosophies in 1998 when a group led by Raymond and Bruce Perens co-founded the Open Source Initiative (OSI) to make free software more commercially attractive (http://www.opensource.org). OSI has become the steward of the open source definition and serves (together with the FSF) as a standards body for vetting and approving open source licenses, of which there are dozens (http://www.opensource.org/licenses/alphabetical).

The FOSS movement today is led by both OSI, which emphasizes the practical benefits of the FOSS development model, and the FSF, which emphasizes the moral and philosophical importance of free software [8]. As reflected in its name, the HFOSS Project accepts the principles and practicalities of the FOSS movement as characterized by both FSF and OSI.

Since spring 2006 the HFOSS Project has engaged students from Bowdoin College, Connecticut College, Trinity College, Wesleyan University, the University of Connecticut, and the University of Hartford in a number of software-development projects that serve the community**.** Its main software-development activities take place during its annual 10-week summer internship program, now in its third year (see the figure here). But students also work on HFOSS projects in courses, independent studies, and thesis projects outlined in the sidebar.

Given its primary goal of contributing to the revitalization of undergraduate computing education, the HFOSS Project has six specific objectives that, if met, would represent significant progress toward its overall community building and revitalization goal:

* Introduce new concepts and methodologies;
* Attract a new demographic;
* Debunk the computing-is-coding myth;
* Unite town and gown;
* Contribute to society; and
* Create a portable and sustainable model.

Concepts and Methodologies

As a concept, HFOSS is clearly attractive to university computer science students and may help attract new students to computing. This is reflected not only in the interest that has been generated in the summer HFOSS Institutes, where typically two to three times more students apply than can be accommodated but also in the feedback we receive from students in HFOSS software-engineering and software-development courses throughout the curriculum.

To explore this concept further, in spring 2008 a "general education" course called "Open Source Software for Humanity," was taught (via videoconference) at Trinity College and Wesleyan University [4]. Its "hook" was getting students to reflect on their own experience with FOSS products (such as Wikipedia and the Firefox browser). Not surprisingly, the students were receptive to the ideals of sharing and community and the public good. They were also enthusiastic about discussing their experience with Wikipedia, blogging, open source politics, and other aspects of the free and open culture they had grown up with. As suggested in reading (such as [1] by Benkler and Nissenbaum), they see the distributed FOSS model as an alternative means of producing culturally useful goods (Wikipedia) and services (SETI@home). Similarly, students generally see elements of the FOSS ethic in their own experience with file sharing. They recognize that this is a time of change in public thinking about intellectual property and the common good.

But despite their everyday use and enjoyment of FOSS products and their widespread acceptance of the freedom and openness characterizing the FOSS model, few students see the connections between the FOSS movement and the computing discipline. As one said, "Wow. I really got to look at how computer science can relate to humanitarian efforts. I now really understand [FOSS] and know why it came about." As this suggests, there is an educational opportunity here.

As a methodology, the FOSS development model represents a revolutionary break from traditional software development [7]. However, despite its commercial success, relatively little effort has gone toward incorporating the FOSS development model into undergraduate computing curricula. Our efforts to see how others have incorporated FOSS into their curricula revealed only a handful of reports (reviewed in [3]).

Our experiments with introductory and advanced courses, independent studies, and summer internships have shown that FOSS software and tools, including Apache, PHP, MySQL, Eclipse, PhpMyAdmin, and SVN, are quite accessible to today's undergraduates.

Students are also eager to engage the HFOSS methodology, which requires an approach different from the traditional mode of undergraduate instruction**.** Working with mentors and in teams on real-world development projects is a motivator for students, despite the extra challenge the approach means for instructors. Similarly, working with local clients and international development communities is another strong motivator. For example, students get to see directly that writing good documentation is as important as writing good code. The quality of their work improves as they recognize their increased level of public accountability. This message is constantly reinforced by mentors, peers, and clients.

Depending on specific course or project, students come with different levels of expertise, ranging from no prior programming experience for an introductory course, to having nearly completed the major requirements for upper-level and software-engineering courses. Engaging students through HFOSS must be done with sensitivity to their backgrounds and interests. But the projects themselves are rich and varied enough to accept contributions from students with different backgrounds. For example, students with no programming experience are still able to make significant contributions in requirements-gathering and documentation-writing [4].

Students are comfortable working in virtual teams and groups, having grown up with Facebook and Instant Messenger and interacting with friends through all kinds of electronic media. They respond equally well to wikis for working collaboratively on documents and presentations and sharing their source code on Sourceforge. One student said, "I now have a better understanding of what it is like to work with and contribute to a team of people, even when I may never meet them in person."

Thus, HFOSS as both a concept and a methodology has much to recommend it.

New Demographic

Computer science has not been broadly attractive at the undergraduate level, especially to women and other underrepresented groups. An April 2006 article in *Computing Research Association Bulletin,* based on data from the National Science Foundation and other sources, reported "[c]omputer science has the dubious distinction of being the only science field to see a fall in the share of its bachelor's degrees granted to women between 1983 and 2002" (http://www.cra.org/wp/index.php?p=83).

Attracting women and other underrepresented groups to computing remains a particularly challenging HFOSS Project objective. Only four women were enrolled in a 13-student introductory course in spring 2008, and for the summer 2008 internship program, only six out of 29 applicants were women. Of the 10 CPATH-funded summer interns only three were women, and two others were African-American. These numbers are not good, though they are somewhat better than the numbers in non-HFOSS computer science courses. For example, the fall 2008 CS1 courses offered at Connecticut College, Trinity College, and Wesleyan University, included only 10 women and two African-American students, out of a total of 69 students.

While these data are too sparse to support conclusions one way or the other regarding the appeal of HFOSS to women and other underrepresented groups, evaluations received from participating students suggest that the HFOSS approach has the potential to attract more women students to computing in the future. The responses from women students suggest they speak positively about the project to their female friends. To help address this issue, in summer 2010 we will be extending the HFOSS Project to include a women's college and a traditionally black university. However, given the relatively small number of women and minorities who come to college with an interest in computing, this initiative may not solve the problem altogether; the solution, if there is one, may ultimately extend beyond the academy.

A widespread misconception about computing is that it is all about programming or coding. At most U.S. schools the introductory sequence focuses largely on teaching a programming language, further reinforcing this misconception. The HFOSS approach addresses this by contextualizing programming within a broader problem-solving activity. Being engaged in real-world projects with teams of developers, students see that programming is just part of a complex, team-oriented, creative process that produces software to benefit society. Working closely with real clients, students see the need for transparent and secure code, extensive testing, and writing excellent user manuals and other supporting materials. They want to master these activities to improve their systems rather than step through mere academic exercises.

Another important HFOSS element is the ethic of sharing and collaboration. For this reason, the HFOSS Project teams students with one another, as well as with mentors, IT professionals, and HFOSS community members. The HFOSS development process has no room for lone programmers working in isolation.

Student feedback on these points reflects these observations. For example, one student wrote, "[this activity] shows how computer science can be a very helpful field of study than what we just know of it as programming at different programming languages." Another said, "[this activity] definitely changed my views of how effective software projects can be run. If we work collectively for the greater good (which HFOSS certainly encourages) then we can get much more done."

The HFOSS Project has focused on individual courses and internships and only just begun to think how its approach might fit into an undergraduate curriculum. Reinforced throughout our experience is the longstanding view that computer science must be presented as a problem-solving discipline, and the more this value is built into the computing curriculum the more attractive it will be to a wider variety of bright students eager to solve problems. Georgia Tech and other institutions have begun exploring curricular models that contextualize programming within broader applications of computing (http://www.insidehighered.com/news/2006/09/26/gatech). The HFOSS approach would clearly complement such a model.

A common software industry complaint is that new computing graduates are strong on theory but lack practical understanding of the modern IT workplace. A common complaint from academics is that IT professionals want colleges and universities to serve as training centers for their latest programming languages and software platforms.

HFOSS addresses both by recruiting computing and IT professionals as advisers and mentors for its summer interns. For example, IT consultants from Accenture Corporation help mentor HFOSS students and serve as advisers in project management and other areas. Students appreciate the mentoring as they begin to understand the complexity of software development. They see that challenging problems rarely yield to "textbook" solutions and that the design process is often a protracted interaction between programmers and end users. As one student said, "[this activity] definitely helped me understand more options of the IT profession. Now I know one more aspect of it, and how exciting it can be."

Portable, Sustainable Model

If the HFOSS model is to make a positive contribution to undergraduate computing curricula, it must continue to grow beyond the three campuses—Connecticut College, Trinity College, and Wesleyan University—where the Project began. During the past 18 months, with the support of the CPATH grant, we have seen evidence that such growth can be accomplished, as new HFOSS began at Bowdoin College, Brunswick ME, and the University of Hartford, Hartford, CT. However, continued growth will require development of a supportive infrastructure and portable model that is easily adopted by other institutions.

Part of the effort to build a sustainable HFOSS model must include faculty development. Toward this end, we conducted outreach workshops for faculty at SIGCSE08 in Portland, OR, and CCSCNE08 in Staten Island, NY, (http://www.cs.trincoll.edu/hfoss/wiki/SIGCSE_2008_Workshop) to promote the HFOSS model as something that could be tried by undergraduate faculty at other institutions. Feedback from workshop participants indicated that both the humanitarian and the FOSS aspects of the have substantial appeal to computing faculty. However, despite the basic appeal of HFOSS, many challenges remain before more than a few other schools are able to integrate HFOSS into their computing curricula:

Faculty development. As with any new pedagogical endeavor, developing a new approach to teaching software design requires considerable initiative, time, and support. Faculty need time to learn new languages and tools and become active in the HFOSS community on their own before they are able to introduce HFOSS into their courses. To support this endeavor we are planning a summer training experience for faculty, similar to the week-long NSF-funded Chautauqua workshops (http://www.chautauqua.pitt.edu).

Software-tool support. Although FOSS software technology is free, creating a platform of FOSS tools to support a course or student project requires considerable time and effort. Faculty do not normally have time for downloading and installing software and making sure it works. One potential solution is a one-click installation that works on a variety of platforms. Another is for instructors to enlist such support among their universities' IT staff. The HFOSS project has begun to develop resources and processes to help other colleges and community groups get involved, including a set of free and open Web-based resources, software tools, and other support materials (http://repository.hfoss.org).

Community development. Being involved in HFOSS means joining and taking an active role in one or more HFOSS communities or projects, a process that can be somewhat bewildering and intimidating, especially for large well-established projects. We have identified and worked with communities and projects (described in the sidebar) that are accessible and welcoming. Sahana, OpenMRS, and InSTEDD are appreciative of student contributions and accepting of the compromises imposed by academic calendars and curricula. This summer we have begun working with the GNOME project on user accessibility problems (http://projects.gnome.org/accessibility/). And a group of HFOSS students from several schools are currently working on the Portable Open Search and Identification Tool (POSIT), a disaster-management tool for the Google Android phone (http://code.google.com/p/posit-android/). All are ongoing projects that welcome contributions from faculty and students at other schools.

Cultural, institutional, curricular buy-in. Creating a new course or revising an existing course requires department support and approval. New courses must serve the general goals of the department's curriculum. So the computer science academic community needs a more widespread and systematic discussion of how HFOSS might fit into the curriculum. Similarly, faculty development itself is not possible unless faculty and their departments see this kind of engagement as an important form of community outreach and are therefore willing to invest the additional time and accept the additional complexity it requires. This may represent something of a cultural shift for some faculty.

Helping address these challenges, the HFOSS Project organized the first of what are planned to be an annual symposium on "Integrating FOSS into the Undergraduate Computing Curriculum" (http://www.hfoss.org/symposium09/). The March 2009 symposium's main goal was to bring together representatives from academia, industry, and the FOSS community to explore ways of integrating HFOSS into undergraduate teaching. The lively discussions that took place in Chattanooga, TN, helped identify a number of issues that stand in the way of more widespread adoption of the HFOSS model. Faculty participants identified a number of activities that could help them get involved, including summer training workshops and support for hosting open source code repositories.

The discussion focused on what kinds of support faculty and their students would need to get started. One of the most promising ideas now being explored is establishment of a number of "HFOSS Chapters" whereby a faculty member and a couple of students could take on a FOSS project beginning summer 2010. The software industry and FOSS-community representatives at the symposium expressed their eagerness to support the effort, including by helping train faculty to use FOSS tools and by providing "on ramps" to help faculty and students become integrated in the FOSS community.

To date, fifteen additional schools have expressed interest in becoming HFOSS Chapters. Similarly, several more industry and FOSS-community supporters have volunteered to serve on the HFOSS Project steering committee and advisory board, including representatives from Google, Sun Microsystems, RedHat, the Mozilla Foundation, and the GNOME project.

Sustainability. No such project can succeed in the long term without first encouraging the wide adoption of its methodologies and goals. But what would a sustainable model look like? In order to broadly influence undergraduate computing, high school and college students must be able to learn about FOSS and its humanitarian applications, thus requiring some kind of national organization and infrastructure to manage three functions:

* Funding internships (summer and, perhaps, academic year, too) to support student involvement in specific HFOSS projects;
* Funding a campaign to advertise HFOSS to prospective students, much as Teach for America and Habitat for Humanity advertise themselves; and
* Creating and managing an infrastructure and a process whereby students are matched with specific HFOSS communities (such as Sahana and OpenMRS). The Google Summer of Code project, in which FOSS projects apply to Google for student-internship support, might serve as an adaptable model (**http://code.google.com/soc/**).

The hope is that the computing industry and FOSS communities embrace the potential value of HFOSS for computing students. In addition to revitalizing undergraduate computing education, a strong and diverse cohort of U.S. college graduates who come into the work force with FOSS experience will enrich the computing industry, as well as the various FOSS communities.

Conclusion

Given the relative youth and scale of the HFOSS Project, it would be premature to make sweeping claims on its behalf. However, its ongoing objective is to systematically monitor the project's effects on undergraduate education to determine what would happen if students begin to see computing as a discipline that developed software to help their friends and neighbors in need. Toward this end, the Project employs instruments and metrics, including student and faculty questionnaires, presentations at computing education venues, and outside consultants from academic institutions and industry.

Though this evaluation is still preliminary, a number of promising signs have emerged. First, HFOSS as a concept and methodology can indeed be introduced into the undergraduate computing curriculum. Our pedagogical experiments suggest that positive results are achievable via several approaches. For example, a general-education course can provide a coherent one-semester introduction to HFOSS techniques and to the broader cultural and societal effect of the HFOSS movement. Independent-study projects and internsips provide a flexible venue through which students and faculty can contribute to specific HFOSS projects in both the academic year and the summer. Upper-level software-engineering courses can be used to engage students in real-world HFOSS projects as part of their course work.

Second, feedback from faculty outreach activities, including 2008 SIGCSE and CCSCNE workshops and the 2009 symposium, suggest there is significant faculty interest in integrating FOSS into the computing curriculum in many undergraduate institutions. Despite ongoing questions involving where, when, and how best to do it, the FOSS model is flexible enough to allow different institutions to answer these questions in ways that best suit their programs.

Third, the students engaged thus far are attracted to the HFOSS concept for both the opportunity to learn concepts, languages, and skills they don't see in other courses and for their interest in community service. Over the long run, these motivations promise to help attract a wider range of capable students to computing, including more women and members of other underrepresented groups.

Fourth, student feedback suggests that engaging students in HFOSS projects helps foster a more constructive perception of the craft of programming and problem solving while generally reducing the computing-is-coding misconception. The ongoing HFOSS challenge is to spread this more positive perception across the entire undergraduate landscape. To some degree it will happen through word of mouth, as students share their positive HFOSS experiences with one another. But, as noted earlier, truly changing perceptions of computer science requires a concerted and sustained effort with broad support from the computing industry, the FOSS communities, primary and secondary schools, and society at large.

Finally, the HFOSS Project has expanded from its three initial schools, one corporate partner, and a single software project into a vibrant community that today includes active faculty participants from eight U.S. colleges and universities (and expressed interest from many more), industry representatives from five IT corporations, ongoing software-development projects with two local nonprofit organizations and five international FOSS communities. This growth—largely unplanned at the beginning of the Project—is indicative of a latent (inter)national interest in the HFOSS concept. If such expansion is sustained, it will help demonstrate that HFOSS can have a significant and lasting effect on the undergraduate computing community, especially its curriculum, culture, and enrollment demographics.

(start sidebar)

Five HFOSS Software Projects

The first three projects are international in scope and involve students in global communities and ongoing software-development projects. The other two are local and engage students in local nonprofit organizations and development of custom software that helps the organizations directly. Participation in all five depends on the Internet and on various communication, collaboration, and software-development technologies. In addition to list servers for shared email messages, students use Wiki pages and version-control repositories to share documentation and code with one another and with their mentors. Development teams in each project hold regular virtual meetings through videoconferencing, Internet relay chat, and other communications media.

Sahana project. Sahana (Sinhalese for relief) is a FOSS disaster-management system built initially by Sri Lankan volunteers in the aftermath of the 2004 Asian tsunami (http://www.sahana.lk). It addresses the common coordination problems that arise during disaster recovery--finding missing people, managing aid and volunteers, and otherwise assisting the effort. It has also been deployed in numerous disasters around the world over the past five years, most recently in the 2008 Burma cyclone and the 2008 earthquake in China. From its beginnings in Colombo, Sri Lanka, it has grown into a worldwide community of individuals and organizations that support ongoing development. It received the 2006 Social Benefit Award from the Free Software Foundation (http://www.fsf.org/social-benefit-award-2006).

Our involvement with Sahana began in January 2006 and has focused on development and support of the volunteer-management (VM) module, which was incorporated into the Sahana code base in December 2006 [3]. A first prototype of the module, which supports registration and management of relief volunteers, was field-tested with Sahana at the June 2006 Strong Angel III Disaster Response Demonstration in San Diego (http://www.strongangel3.net). One of the students said of that activity, “This isn’t a typical computer science project. How many students get to publish software that can potentially affect millions of people’s lives?” (http://www.trincoll.edu/AboutTrinity/News_Events/trinity_news/061013_sahana.htm).

Over the past three years HFOSS has become both an integral contributor to and a beneficiary of the Sahana community. The author Trishan de Lanerolle now serves on Sahana’s management committee, and two HFOSS alumni have been granted “committer” status, giving them direct access to the Sahana repository.

On the educational side, Sahana has been used as a teaching platform in numerous courses, independent studies, and summer internship projects.[c] Students in the 2007 HFOSS Summer Institute performed a complete refactoring of the VM module based on a model-view controller design.[d] In spring 2008 Sahana was used as the software platform for an introductory course involving 13 Trinity and Wesleyan students [4]. And in summer 2008 a team of four undergraduates developed a credential-verification module under the direction of Frank Fiedrich of George Washington University's Institute for Crisis, Disaster, and Risk Management (http://www.gwu.edu/~icdrm).

In May 2008, in an engagement that probably best illustrates how the HFOSS community makes a positive contribution, a team of six students and faculty worked closely over 10 days with a Sahana team in Sri Lanka and an IBM team in China to support deployment of Sahana in Chengdu following the devastating 2008 earthquake there (see sidebar Figure 2). One China team member later said, “It was really an emotional moment of truth when we saw the happy tears as people were reunited with their families.

Eventually, we can say with pride that what we have done is worth remembering for our whole life. We helped people in the disaster area with our technology" (http://blog.hfoss.org?cat=29).

Finally, in keeping with the sharing nature of FOSS culture and licensing, the VM module has found application beyond the Sahana system and the disaster-recovery domain. For example, a modified version of the original VM module has been incorporated into a coastal flood emergency-preparedness system for the New York City Office of Emergency Management**.** The OEM system is designed to manage the evacuation of 1.2 million people from low-lying areas and shelter 600,000 evacuees in temporary shelters. Similarly, the Office of Emergency Management in Galveston City, TX, is looking at the Sahana system and the VM module for its own disaster-preparedness purposes. Using Sahana as the basis for disaster preparedness systems like these could provide a way for students in many schools around the U.S. to be involved in HFOSS development projects.

Open Medical Record System project (OpenMRS). OpenMRS (http://www.openmrs.org) is a FOSS electronic medical record system designed to assist health professionals in the treatment of AIDS, malaria, and tuberculosis in the developing world, particularly in Africa. The project began in 2004 as a joint venture of the Regenstrief Institute (http://www.regenstrief.org) and Partners In Health (http://www.pih.org), aiming to provide health-care professionals the information-management tools they need to combat these diseases and provide quality care. OpenMRS has been deployed in a number of African countries, including Kenya, Lesotho, Mozambique, Rwanda, Tanzania, South Africa, Uganda, and Zimbabwe.

Like Sahana, the OpenMRS community is a worldwide network of individuals and organizations contributing to the development of the software. It makes extensive use of Java-based development technologies, including Java server pages and servlets, the Spring application framework**,** and other advanced FOSS tools. Unlike Sahana's relatively simple PHP/MySQL platform, OpenMRS is substantially more complex and challenging. Nevertheless, HFOSS students have made several important contributions to the OpenMRS project**.**

During summer 2007, they developed a module that enables the system to be used with a touchscreen monitor, an effort that continued as a senior thesis project and resulted in a generic touchscreen application, AutoTouch, providing an API to add a touchscreen interface to any Web-based application (http://sourceforge.net/projects/autotouch). During the summer 2008 HFOSS Institute students created an OpenMRS module for uploading and editing patient medical images and defined and implemented a new systemwide data structure that allows physicians to input numeric observations as ranges (1-5), inequalities (<100), ratios (2:5), and qualitative values (too few to count).

During spring 2009, a software-development course based on OpenMRS was offered (via videoconference) to students at Connecticut College, Trinity College, and Wesleyan

University. Focusing on how software-engineering techniques play out in an FOSS setting, it required that students to put theory into practice by contributing to OpenMRS.

Innovative Support To Emergencies, Diseases and Disasters (InSTEDD) Project. InSTEDD (http://www.instedd.org) is a lab devoted to developing software for early disease detection and disaster response. Founded in 2006 by Larry Brilliant of the Google Foundation, **it** is funded in part by Google.org and the Rockefeller Foundation *(*http://www.rockfound.org). It is designed to integrate, tag, classify, and visualize data from various sources (such as news, weather reports, sensor data, and field reports) with the goal of detecting and managing disease outbreaks. Like Sahana and OpenMRS, InSTEDD is an international effort**.**

Two students in the summer 2008 HFOSS Institute collaborated with researchers in Seattle and Buenos Aires to help develop and test data mining algorithms for the Evolve project. After studying support vector machines and Bayesian networks and mastering software tools (such as Eclipse and LIBSVM) (http://www.csie.ntu.edu.tw/~cjlin/libsvm/), they developed Alpaca Light Parsing and Classifying Application (ALPACA), a parsing and classification tool for categorizing documents into user-provided classes (http://2008.hfoss.org/ALPACA). ALPACA allows Evolve developers and others to test a variety of classification technologies across a number of data sets. Two other students are following up on this work as part of the 2009 HFOSS Summer Institute.

Ronald McDonald House Homebase project. RMH Homebase is a Web-based volunteer management and scheduling system used at the Ronald McDonald House in Portland, ME (http://www.rmhportland.org). Developed in spring 2008, it addresses the Portland RMH's need for software to replace its error-prone, time-consuming manual rolodex and calendar-scheduling process. The development team included four Bowdoin College students, one professor, Bowdoin IT staff, and several RMH employees and volunteers who would eventually use the system. The development took place almost entirely within a software-development course (http://hfoss.bowdoin.edu) using a distributed-development process and the same FOSS tools used in the global projects [10].

The four students earned independent-study credit in computer science, as well as a valuable learning experience. The Portland RMH gained a valuable piece of software that arguably would never have been developed outside the FOSS framework. The software is published on Sourceforge (http://www.sf.net/projects/rmhhomebase) under a GNU GPL and is available to other volunteer organizations.

One difference between this project and the three international projects is the software was designed and built from scratch, though it relied on study of the Sahana system. Also, unlike the international projects, it involved close interaction with end users throughout the development process. It also provides a potential basis for groups of students and faculty at other colleges and universities to join in by, say, customizing and adapting the RMH Homebase system for other Ronald McDonald Houses or for other local nonprofit organizations.

AppTrac project. Literacy Volunteers of Greater Hartford provides literacy training to adults, mostly through specialized software applications in its Hartford, CT, computer lab. The staff manually keeps track of student logins, the applications the students use, and similar information that is then painstakingly compiled into reports to the organization's board and funding agencies.

In spring 2008, students from the University of Hartford developed requirements and built a prototype application-tracking system (AppTrac) as part of an upper-level software-engineering course**.** During the 2008 HFOSS Summer Institute a four-student team from three different colleges—Connecticut College, Trinity College, and the University of Hartford—developed the prototype into a kiosk-based system (http://sourceforge.net/projects/apptrac/). In fall 2008, working through virtual meetings, code-sharing repositories, and Wikis, the same students modified the system for eventual release as a generic kiosk system for similar applications.

Unfortunately, due to the loss of its technical staff position, the Literacy Volunteers of Greater Hartford ultimately decided not to deploy the AppTrac in its lab**.** However, the software is being modified by students in the 2009 HFOSS Summer Institute (http://2009.hfoss.org) for deployment in the Hartford Public Library's computer lab. This provides another example of how software sharing benefits both the community and the educational process.

(end of sidebar)

Acknowledgments. HFOSS is supported by the National Science Foundation under grants #0722137, #0722134, and #0722199 and by the Mellon Foundation. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation or the Mellon Foundation.

(figure): 2008 HFOSS summer-internship program students and faculty, Trinity College (http://2008.hfoss.org).

(sidebar) Figure 1: Field testing the Sahana volunteer-management module at the Strong Angel III disaster response exercise, San Diego, CA (http://www.strongangel3.net).

(sidebar) Figure 2: Screenshot of the Chinese-language version of the volunteer-management module (http://blog.hfoss.org/?p=28).

(footnotes)

[a]CPATH is a National Science Foundation program within the Directorate for Computer & Information Science & Engineering, formally known as CISE Pathways to Revitalize Undergraduate Computing Education (http://www.nsf.gov/cise/funding/cpath_faq.jsp).

[b]See also a similar definition in http://en.wikipedia.org/wiki/Humanitarian-FOSS.

[c]These and other activities involving Connecticut College, Trinity College, and Wesleyan University are funded by a Mellon Foundation grant for videoconferencing facilities.

[d]The 2007 HFOSS Summer Institute was funded by a grant from the Aidmatrix Foundation (http://www.aidmatrix.org).

References

1. Benkler, Y. and Nissenbaum, H. Commons-based peer production and virtue. *Journal of Political Philosophy 14,* 4 (Nov. 2006), 394–419.

2. Chopra, S. and Dexter, S. *Decoding Liberation: A Philosophical Investigation of Free Software.* Routledge, New York, 2007.

3. Ellis, H., Morelli, R., de Lanerolle, T., Damon, J., and Raye, J. Can humanitarian open source software development draw new students to CS? In *Proceedings of the 38th ACM SIGCSE Technical Symposium on Computer Science Education* (Covington, KY, Mar. 7–11), ACM Press, New York, 2007, 551–555.

4. Morelli, R. and de Lanerolle, T. FOSS 101: Engaging introductory students in the open source movement. In *Proceedings of the 40th ACM SIGSCE Technical Symposium on Computer Science Education* (Chattanooga, TN, Mar. 4–7), ACM Press, New York, 2009, 311–315.

5. Patterson, D. Rescuing our families, our neighbors, and ourselves. *Commun. ACM 48,* 11 (Nov. 2005), 29–31.

6. Patterson, D. President's letter: Computer science education in the 21st century**.** *Commun. ACM 49,* 3 (Mar. 2006), 27–30.

7. Raymond, E.S. *The Cathedral and the Bazaar*. O'Reilly Media, Sebastopol, CA, 2001; see also http://www.catb.org/~esr/writings/cathedralbazaar/.

8. Stallman, R. Viewpoint: Why 'open source' misses the point of free software. *Commun. ACM 52*, 6 (June 2009), 31–33.

9. Standish Group International**.** Free open source software is costing vendors $60 billion, new Standish Group International study finds. Standish Group International Press Release (Apr. 2008); http://www.marketwire.com/press-release/Standish-Group-International-844462.html.

10. Tucker, A. Teaching client-driven software development. *The Journal of Computing Sciences in College 24*, 4 (Apr. 2009), 29–39.

Ralph Morelli (ralph.morelli@trincoll.edu) is a professor of Computer Science at Trinity College in Hartford, CT.

Norman Danner (ndanner@wesleyan.edu) is an associate professor in the Department of Mathematics and Computer Science at Wesleyan University, Middletown, CT.

Allen Tucker (allen@bowdoin.edu) is the Anne T. and Robert M. Bass Professor Emeritus in the Computer Science Department at Bowdoin College, Brunswick, ME.

Heidi Ellis (hellis@wnec.edu) was a visiting professor at Trinity College at the time of this work and is now an associate professor and chair of Computer Science and Information Technology at Western New England College, Springfield, MA.

Gary Parker (parker@conncoll.edu) is an associate professor and chair of the Computer Science Department at Connecticut College, New London, CT.

Danny Krizanc (krizanc@wesleyan.edu) is a professor of Computer Science at Wesleyan University in Middletown, CT.

Trishan R. de Lanerolle (trishan.delanerolle@trincoll.edu) is project director of the Humanitarian FOSS project based at Trinity College, Hartford, CT.

Ozgur Izmirli (oizm@conncoll.edu) is an associate professor of computer science at Connecticut College, New London, CT.

Pull Quotes

Despite the great commercial success of **FOSS**, little effort has gone toward incorporating the **FOSS** development model into undergraduate computing curricula.

Students see that challenging problems rarely yield to “textbook” solutions and that the design process is often a protracted interaction between programmers and end users.
If women and members of other underrepresented groups begin to see computing as a field that serves humanitarian purposes, then many more students will eventually be computing majors.