Review Essay

Paul Resnick
University of Michigan

Service-Learning in the Computer and Information Sciences:
Practical Applications in Engineering Education

Brian Nejmeh, Editor

In Service-Learning in the Computer and Information Sciences, editor Brian Nejmeh offers a collection of reflective case studies describing projects and larger university programs that involve undergraduate students in service-learning projects. It provides, first and foremost, a set of existence proofs, that service-learning is something that can occur in the context of Computer and Information Sciences, not only in the social sciences and humanities where it might seem more natural. Students have engaged in many kinds of service, with projects structured in various ways. The reports offer tips and not a few tales of woe that would be valuable reading for any professor or staff member organizing or considering organizing service-learning projects in the computer sciences. Partner organizations might also find the reports valuable for thinking through their own motivations and readiness, especially if they are going into the project anticipating that a service-learning project will be an inexpensive way to meet their information technology needs.

The reports offer many insights about best practices and potential pitfalls. One key challenge is finding the sweet spot of projects that are genuinely useful to the individuals or organizations receiving the service and that exercise the skills that students have developed or that universities want the students to develop. Another challenge is managing the mismatch between course timelines and the natural timelines of projects. A variety of approaches to both these challenges are explored. The reports offer inspirational examples of individual transformation that occurred as a result of service projects. They also hint that such transformations may not be common, and may be less frequent the more technical and "indirect" the service becomes.

The book leaves some important questions unanswered, and some not even asked. None of the reports documents the costs of running service-learning programs and compares that to the number of students who participate. They do not address the preparation of students for service careers or how service-learning projects may fit into a larger ecosystem that also includes professional service providers. And they do not address the service-learning opportunities that may be available through internship placements.

About the Content

Perhaps the most obvious value this book provides is its illustration of diverse service-learning projects appropriate for students majoring in the computing and information sciences. Not only does it show that there's a way to fit service-learning into engineering education, it shows that there are many ways to do it.

The first chapter, by the book's editor, outlines a useful framework for thinking about what might be called the design space of service-learning programs in the computing and information sciences. One dimension is the "project type", or the kind of service to be delivered, ranging from training to systems selection, to support, to custom development. A second dimension, the "activity range," describes which parts of a project's life cycle the students participate in, from analysis to design to implementation, testing, deployment, or assessment. A third dimension describes the "project mode," i.e., whether the service-learning is co-curricular (independent of credit-bearing coursework), curricular (part of credit-bearing coursework), or a hybrid of the two.

Another dimension, not explicitly mentioned in the introductory chapter's framework but evident in the case studies in the book, is who is being served. In many cases, students tried to improve the internal operations of nonprofit organizations. In other cases,
students served individuals more directly, but even then there was typically an intermediary partner organization that helped to facilitate the service. Some of the client/partner organizations were local, but some were on other continents.

The most common project type described in the book was some kind of custom development, typically of software systems. For example, the EPICS program (Engineering Projects in Community Service), pioneered at Purdue University and now adopted at several other universities, frequently focuses on software development. As described in chapter 2, projects at Purdue have ranged from creating iPad apps to help students with disabilities to energy modeling systems for low-income homes to volunteer management systems for local nonprofit organizations. Chapter 3, describing ten years of using the EPICS model at Butler University, describes two especially successful long-term projects, creating a suite of web-based educational software for teaching languages at a local middle school and a system for a public radio/TV station to store and manage video and related metadata. Chapters 5 and 8 describe software development projects that go under the umbrella of HFOSS, Humanitarian Free and Open Source Software, starting at Trinity College and spreading to Bowdoin College and other campuses. Some were directed at creating tools to aid in disaster management and recovery such as food distribution after the Haiti earthquake. Others were more similar to some of the EPICS projects, like the Bowdoin project described in Chapter 8 to create an online volunteer scheduling system for Ronald McDonald house.

Several projects focused on training. Chapter 12 reports on students taking a 10-day spring break trip to Brazil where they conducted training workshops on typing, email, and MS Word and Excel. Chapter 14 reports on students who were paired with local senior citizens to tutor them weekly on technology skills.

A few projects were organized around provision of hardware. Chapter 18 describes a student group’s vision of collecting outdated laptops from individuals and organizations, sanitizing/reimaging them with Linux and OpenOffice, and then routing them for delivery to developing countries by individual volunteers traveling there for other reasons. Chapter 22 describes, among other things, a multi-year project outfitting schools in Belize with computer labs. Chapter 17 describes setting up computer labs in Los Angeles area schools and conducting workshops as part of an after-school program.

Key Insights

Several valuable insights about benefits, challenges, and best practices appear as themes across the case studies. First, projects for real clients seem to be a great way to teach some of the softer, less tangible skills in the engineering curriculum. Many of the course-based projects include explicit instruction about project management tools and techniques. Students learn teamwork and communication skills, often through bitter experience. Several chapters offer examples of communication mishaps between students and clients, usually with a happy ending. The ones with unhappy endings may not have been reported, but presumably they were valuable learning experiences as well. Some of the professors also report seeing student acquire meta-skills of how to acquire new technical skills on their own, outside the typical course experience where professors provide explicit scaffolding.

Several chapters have interesting reflections on ethical issues that students and faculty are forced to engage through their projects. Some have a limited definition of ethics that seems to focus on professional behavior: showing up for appointments, keeping commitments, and honesty. Others reflect on more fundamental problems of what to do when the client wants you to do something that violates your own sense of right and wrong. Chapter 22 offers a fascinating reflection on outfitting refurbished computers for developing countries with open source software when the locals strongly prefer to use pirated versions of commercial software. The author comes to a sympathetic understanding of why the locals prefer commercial software and are willing to use pirated copies, but he is not willing to help provide pirated software and tells his students not to do so, either. There is no easy solution, and it is clear that the self-imposed limitation adversely affected their ability to meet the needs of their clients/partners.

A critical recurring theme is that client project timelines rarely align well with course timelines. In courses where students participate in picking or defining projects, too much time is spent in that process and not enough time is left for the project work. Even when professors line up projects in advance, few valuable projects can be completed from start to finish in the 10-14 weeks available for a course. Students need input from clients in bursts; clients are not always able to provide it but then are surprised when the students who had been very available are no longer available when the course is over. Successful projects almost invariably involve some kind of continuity beyond the semester. Sometimes this involves committed students or faculty doing fol-
low-up work after the course ends, on a volunteer basis. Some programs, such as the Purdue EPICS program, are explicitly designed around projects that last beyond the life of single courses. Pieces of larger projects are carved out for student groups to tackle during a course. Between courses, some students are hired to keep projects going. The same problem plagues student-led, co-curricular projects: Campuses empty during summer and one cohort’s enthusiasm is not always passed on to a new crop of students. Anyone organizing a service-learning program around engineering projects will need to confront the need for continuity in some way; this is something that should be planned for from the beginning.

Even beyond the mismatch of college and client timelines, finding appropriate computing service projects is challenging. The ideal project will allow students to exercise specialized technical skills. In the context of some courses, only projects that exercise particular technical skills covered by that course are appropriate. For example, in a software engineering course, students have to write code. Sometimes, a client’s needs may be met with custom code, written from scratch. But that may not be the best way to meet the client’s needs. Installing and configuring some existing software package might leave the client with more maintainable software, but not meet the learning needs of the students. Capstone courses where the major goals are building the soft and intangible skills mentioned above are less constraining in this regard than courses organized around more specific technical skills.

A good client or partner is critical. The most important element of a good partner is willingness to engage with the students, on the students’ schedule. Students need to understand the project goals, and then get feedback at various stages. Some chapters describe using agile development methods where, ideally, a staff member of the client organization joins the project team in order to give feedback almost continuously. None of the projects described achieving that ideal. At the other extreme, some projects failed because clients did not provide timely feedback or even assign anyone to be trained on using software that students created for them. On international projects, chapter 18 was particularly eloquent about the need for a partner that acts as a cultural bridge, which requires significant understanding of both the local culture and American culture. In domestic projects, too, it can be helpful if clients or partners understand and are committed to the educational mission of the project, as well as the service outcomes.

An interesting tension may occur when clients, who are receiving free labor, are less critical of student performance than they would be of professionals. As described in chapter 9, professors at Washington & Jefferson College found that, to maintain high standards for their students, they could not rely primarily on clients’ evaluations. Indeed, after some experimentation, they settled on a two-event solution to presenting projects: an open house to which clients are invited and a department defense two days later when students defend their work according to the department’s program-level learning outcomes. The tension can be even larger if clients do not treat the projects themselves as the main purpose of their engagements with university students. Rick Homkes, in chapter 21, writes, “Quite simply, some community partners were more interested in having students start a life of volunteerism than in doing an honest appraisal of student appearance and work (or, in one case, even producing a useable product).”

Service-learning projects in the computing and information sciences can, in addition to solidifying some of the soft and intangible skills of an engineering curriculum, also result in personal transformation for students and faculty involved. Most of the evidence provided is anecdotal, and focuses on those who became most engaged. Some of these are powerful. An alumnus in chapter 4 wrote, “…influenced every part of my life, including my graduate studies in leadership and management and my career as a long-term missionary in Burkina Faso.” A professor, describing one of his early project experiences, writes in chapter 22, “I never helped these people, but they sure helped me. They helped me to understand that people want friendship before advice, and to be understood more than changed.”

Some of the chapters reflect on the difference in student experiences between direct, one-on-one service and the more indirect service of building software or information systems. Some students paired with seniors for direct service formed long-lasting relationships and received life-mentoring in return for technology mentoring. In the introductory chapter, Nejmeh comments that students struggle initially to see “indirect service” of providing information systems to organizations that provide direct service as a real form of community service. The indirect nature of the service removes some of the warm “fuzzies” that students experience from directly helping another human being. I suspect interpersonal interactions are also a major source of personal transformation, so that indirect service projects, even though they have the potential for larger-scale impact on the world, tend to have less direct personal impact on the participants.

Chapter 23 makes an admirable attempt to assess more systematically the impact of service-learning on students’ desire to help others and on cultural understanding. Only a few students were able to participate in an international design project at
Valparaiso University, but the study measures impacts even on those who participated in the less glamorous, less service-oriented senior design projects. Ultimately, however, I did not find the evidence convincing. The only control group was students from earlier years, who participated before any international service projects were available. There are too many potential confounds, from other changes in the organization of the senior design project activities to general changes in student attitudes over time to self-selection of who returned surveys years after participating in the activity.

It was interesting to note the differences among authors in how they treated the objectives of personal transformation. Discussions of service-learning in this journal are replete with evidence about bolstering lifelong volunteerism and civic engagement. Many of the programs described in the book, however, limit their stated aspirations to developing soft skills like project management and communication as well as professional ethics. The quote earlier from Rick Homkes notes the problems for project evaluation with partner organizations more invested in students' future volunteerism than in project accountability, but fails to treat that motivation of the partners as a resource for building student commitment to a higher calling beyond the self.

The programs at Christian Colleges were exceptions in engaging broader aims of students' personal development. Chapter 4, by David Vader, is particularly eloquent about how the program fits into Messiah College’s promise to educate whole persons. He unabashedly criticizes not only a culture of materialism, but a culture of accumulating experiences. He argues that we educators have lost the courage to ask students for too much, and so we have nothing to offer to students who are searching for a worthy cause. He writes, “Like many colleges and universities, we anticipate for each student a growing intellect informed by commitments of character and belief that, when mature, is manifested in fruitful action for the public good.” While he attributes this as a shared goal of many colleges, both secular and religious, it is clear in this book that the religious colleges have a higher comfort level and a language for engaging with questions of building character. I have found Harry Boyte’s language around “public work” to be a way, in a secular environment, to engage with students about a higher calling beyond the self and beyond a paternalistic approach of charity to those less fortunate [Boyte & Kari, 1996]. It would have been nice to hear more about how other professors at secular colleges and universities have engaged in these conversations with students.

A couple of chapters suggest the promises and pitfalls of using service-learning projects to build bridges between the computing disciplines and other programs on campus. In several cases, other units on campus, typically non-academic units, are clients for projects that students complete. Chapter 10 describes attempts to co-teach a course aimed at generating local businesses, repeatedly disrupted when the business school faculty member was pulled from the course at the last minute. Chapter 13 describes a software engineering class that partnered with a technical communications class: The software engineering students wrote code and the technical communications students wrote documentation. Unfortunately, coordination costs and mishaps in communication led the communications students to be less satisfied with the experience than the software engineering students.

Neglected Issues

While the collected essays do an admirable job of reporting and reflecting on a variety of experiences, I found a few important issues neglected. One, perhaps the elephant in the room, is the cost effectiveness of running these programs. Some of the chapters describe courses that, on close inspection had as few as four students in them. At best, a faculty member can personally manage only 4-6 projects acting as a “player-coach,” and even that would require more time commitment than teaching a regular course. Over time, many programs have tried to pass off some of the project management tasks to staff or volunteer mentors. None of the chapters estimate the dollars spent on faculty and staff per student credit hour and offer a comparison to the similar costs per student credit hour in other courses. One suspects that directors of the programs have not been anxious to make such calculations, for fear that department heads and deans will cancel the programs. Some of the programs depend on charitable donations to defray the costs of staff or a faculty director, but it is not clear whether they have a sustainable model, or whether the external funds are meant to be “seed funding” for launching new programs.

Another issue is the connection between service-learning and service careers in the computer and information sciences. Around 1999, I had worked with students on a few course-based service projects as part of the University of Michigan School of Information’s master’s degree program. One day I asked one of the students if she was considering a career doing public interest computing work. It hadn’t occurred to her that it was possible and I found that I was unable to give her concrete examples of potential career paths. This led me to organize the Community Information Corps. For many years, we had a series of speakers visit, from founders of community media centers to open source activists to “circuit riders” who helped nonprofit organizations with
their information systems. At some point, I always asked, “How can students like these get a job like yours?” There was an ongoing tension, however, between the majority of participating students who were more interested in volunteer opportunities as a sideline to their regular work and the minority (including me) who were trying to explore how to serve the public as a full-time career.

This connects back to the issue of direct vs. indirect service: As a volunteer activity, direct service is far more rewarding, because it involves human contact and because it is easier to see and to imagine the impacts on people’s lives. But you can’t make much money providing direct, human service to individuals with limited resources. Students who want to make a decent living doing public interest computing work will inevitably need to embrace the value in indirect service. I hope that the service-learning programs described in the book do engage the question of future career options, at least with those students who show some interest; I would have liked to have read about which ways of engaging were most successful.

A related missing element, and perhaps a pathway toward engaging the service career options, is a discussion of the eco-system in which service projects live. There are commercial solution providers and even consultants and consulting shops that focus on serving the needs of nonprofit organizations. Once one takes into account the “total cost of ownership” for an organization, including staff time and ongoing maintenance costs, the free services provided by students may not be the most cost effective for the organization. In some cases, student service-learning may be a good way to launch a project, with a plan from the beginning to transition to maintenance or even reimplementation by professionals if the concept proves to be useful. Sometimes, it may be the reverse, with the best role for students being customization and training around an existing product. None of the chapters directly address roles for professional service providers or product developers. It would have been helpful to get some guidance about when a hybrid of service-learning and professional services would make sense.

There is also no mention of another model for student involvement with nonprofits: student internships in organizations. In our master’s program, we have taught various project-based courses similar in spirit to the ones described in the book. But many more students engaged in projects mentored by a supervisor at an outside organization. These placements are credit-bearing and are supervised by staff who monitor students’ weekly reflections. Many of the employers are companies with no special social mission, but many are social ventures, nonprofits, and government agencies. The challenge with placements involving computing work has been that small nonprofit organiza-

tions often do not have anyone on staff qualified to mentor the students. Our rule of thumb has been that the mentor must be capable of doing the project assigned to the student intern. This, however, has limited our placements to somewhat larger organizations. I would have liked to hear more from the authors about experiences at other universities with reflective practice around internship placements.

The Bottom Line

Despite these wishes for more, on various fronts, I highly recommend this book to anyone involved in or contemplating the creation of service-learning courses/programs in the computer and information sciences. In addition to the high-level orientation to possibilities and problems, the chapters contain a wealth of detailed information. There are samples of memos/contracts with clients as well as samples of student reporting forms and grading rubrics. Many chapters are refreshingly frank, offering readers a chance to learn from others’ mistakes. Nemjeh has performed an extremely valuable service in coaxing and collecting these essays from the many authors who have been experimenting over the past two decades with service-learning in the computer and information sciences.

Notes

1 Following the typical language in the software services industry, many chapters refer to these organizations as “clients.” Other chapters refer to them as “partners” to emphasize greater mutuality of benefits in service-learning engagements.

2 As several chapters point out, especially 2, 10, and 21, these softer skills are considered important academic outcomes in engineering programs, mandated by the ABET accreditation process.

Reference


Author

PAUL RESNICK (presnick@umich.edu) is a professor at the University of Michigan School of Information and director of the Ph.D. program. From 1999-2006, he co-organized the Community Information Corps, which helped students explore public interest careers in informatics. He received a Ph.D. in Electrical Engineering and Computer Science from MIT in 1992. He was a pioneer in the field of Recommender Systems, work which led to the 2010 ACM Software Systems Award. In 2012, he co-authored the book Building Successful Online Communities: Evidence-based Social Design.