Learning trajectories in Open Source Software: Implications for designing problem-based learning experiences in support of higher order thinking

Contents

Problem Statement

Note: Faculty development paper has sources that say real-world experience is a factor in attracting women to computing.

Note: to do, need to clarify what is HOT an what I will focus on measuring as learning outcomes. For now, appears that critical thinking and problem solving are elements of HOT. Hot Table at end of this document might be a good way to discuss HOT and specific learning outcomes in terms of participation in FOSS.

A general problem in computing education (CE) is that it still focuses mainly on knowledge acquisition, while effective practice relies on effective problem solving. (12 O'Grady). As well, there is a shortfall in students pursuing computing majors. In 2008, the most recent year for which data are available, less than 58,000 computing degrees were granted. (http://www.nsf.gov/statistics/nsf11316) However, the Bureau of Labor Statistics reports that new jobs for software engineers alone will increase by 295,000 between 2008-2018. (U.S. Department of Labor, Occupational Outlook Handbook – Occupational Projections and Training Data, 2010-2011 Edition) Furthermore, there are issues with retention and a lack of women and minority participation. (refs)

Problem-based learning has been touted as a learning experience that helps with development of higher-order thinking skills (Wilson, Hmelo-Silver 04, others?) and has been proposed by many as approach that should be taken for computing education. (12 O'Grady, Ellis?, others) Besides cognitive skill development, proposed affective learning aspects of PBL include increased motivation generated by the problem challenge and satisfaction from participation in an authentic, applied learning exercise. Wider spread adoption of PBL may help not only development of the cognitive higher-order skills, but with improved self-efficacy (05 Dunlop)

that may improve retention in the major and with other affective aspects such as motivation and interest that may impact attraction to the field. (refs)

However, despite these claims little adoption of PBL in CE has occurred because institutional and pedagogical reasons (12 O'Grady); this research proposal focuses on addressing pedagogy and learning outcomes. Regarding instruction there's the difficulty of developing adequate problems (Clark, O'Kelly, Huang, Nuutila, Duke) and the lack of instructional design prescriptions to support structuring the PBL process. What if there existed a ready-made learning environment that provided authentic problems for students to learn from that may also provide some of the support structure needed for PBL? Open source software systems, as informal learning environments, have the potential to do so.

Paragraph on FOSS and how can support PBL..

However, while there has been some student participation in FOSS for CE there is a lack of instructional design support, lack of rigorous measure of skills exercised and learning outcomes. This research proposes to study FOSS communities to identify the resources provided that support PBL activities and their relationship to higher-order skill learning practice. This research will be used to influence the instructional design of PBL-based learning activities in support of higher-order thinking skill development. Success in achieving learning objectives will be determined through testing of activities with undergraduate computing students.

Vision statement (wrap it up)

RQs

1. What opportunities does FOSS community participation provide to exercise higher-level thinking skills and CE skills? What is relationship to resources provided by the community and those opportunities (e.g., tasks, technology, community)?
2. What resources (tasks, technology, community) can be leveraged as part of a PBL instructional design to assist with the development of higher-order thinking skills while satisfying CE learning goals?

Lit review

What is PBL, why PBL?

Problem-based learning (PBL) is an instructional model based on constructivist learning theory, which describes the learning process as "*meaning making* … rooted in the context of the situation." Benefits of constructivist-based learning design include engagement in authentic activities that relate to how people actually learn, higher-order learning outcomes, integration of affect and emotion leading to less "academic" more expertise-related discussion, and better transfer of skills outside the academic setting. (Wilson)

Constructivism has three central tenets: 1) Understanding comes from interacting with the environment, 2) the stimulus for learning comes from problems which in turn direct the learning process; and 3) knowledge is derived through social interactions. (Savery 01) Savery 01 lists instructional principles following from this philosophy, noting that among the many learning environments, PBL addresses these most closely:

1. **Anchor all learning to a larger task or problem**- learning must have a purpose for the learner.

2. **Support the learner in developing ownership for the overall problem or task** –align learner goals with instructional goals
3. **Design an authentic task**-the cognitive demands should align with the cognitive demands where the learner will perform ultimately.
4. **Design the task and the learning environment to reflect the complexity of the environment they will function in at the end of learning**
5. **Give the learner ownership of the process used to develop a solution-**don't simplify the environment, but support the student within the complex environment
6. **Design the learning environment to support and challenge the learner's thinking-** don't "proceduralize" how the student approaches solving the problem.
7. **Encourage testing ideas against alternative views and alternative contexts**-Because knowledge is socially constructed, interactions in collaborative group can be used to test individual understanding.
8. **Provide opportunity for and support reflection on both the content learned and the learning process**

Problem-based learning is conducted in small collaborative groups working on realistic ill-structured problems. The teacher acts as a facilitator guiding the learning process helping students learn the skills involved in problem solving and collaboration. Student groups analyze and define the problem by identifying the important components of the problem scenario. From there, they generate hypotheses of how to solve the problem. During this process self-directed learning is employed as students encounter gaps in knowledge. Students then apply their new knowledge, evaluate their hypotheses and reflect on what they've learned. For students to be successful they must learn how to be self-directed learners which requires they be reflective and apply critical thinking about what they are learning. (04 Hmelo-Silver)

In general, it has been proposed that PBL helps students: 1) develop an extensive, flexible knowledge base; 2) develop problem-solving skills; 3) become self-directed learners; 4) become effective collaborators and 5) become intrinsically motivated learners. To elaborate on knowledge construction this also means students learn to connect information across multiple domains. Developing effective problem solving requires developing metacognitive and

reasoning strategies. Metacognition development is important to both evaluating problem solving progress and learning progress for self-directed learning. Developing collaboration skills involves working with others to identify actions needed, come to agreements over issues and effective communication within the group and also aids in the facilitating the social aspect of learning. PBL can aid in intrinsically motivating students that through involvement in problems that are proximal and relate closely to their own learning goals. (04 Hmelo-Silver)

Research studying learning effectiveness of PBL

Learning outcome results using PBL are "mixed.", although much of the evaluation has focused on outcome measures such as testing scores. Furthermore, most research has primarily been in higher education, mainly in medical schools. Of note there is evidence for more accurate hypothesis generation and development of problem solving skills due to use of hypothesis-driven reasoning and better problem identification. Studies of self-directed learning in PBL show that it depends on the learner's learning history, self-regulation of learning strategies, self-efficacy and features of the PBL environment. There has been little research on PBL effects on motivation and collaboration. (04 Hmelo-Silver)

Why PBL in CE? Note need to tie in Parnham some how to make case about HOT skills being important, probably not here, but somewhere

Problem-based learning lends itself well to computing education because computing is driven by problems; requires life-long learning to stay current with emerging technology; work is mainly done in project teams and it generally crosses multiple disciplines. (98 Ellis, 02 Aramego) In addition, while not the focus of this proposal, studies have shown also that when used in CE it may help attract more female students (04 Hamainen, 98 Barrows (need to read), 98 Lambrix), improve retention (04 Hamainen), motivation (04 Hamainen, 10 Wang), and passing rates (04 Hamainen, 03 Parham).

However, CE still focuses on knowledge transmission (12 OGrady), not developing higher-order thinking or advanced reasoning skills, which are critical for a career in the computing industry (12 OGrady, 12 Exter). Effective instructional design in support of this skill development is still an open question. (12 OGrady) Furthermore, given the potential for PBL in CE to increase female enrollment (why?) and improve retention (sources) there is further reason learn how to use effectively. (need to tie PBL to females and application and retention from literature)

PBL has been adopted for use in CE classes covering topic areas including artificial intelligence (Cavedon, Shen), computer modeling techniques (Pereira), first year CS courses (Ambrosio, Fekete, Matzko, Barg, Nuutila, Duke, Okelly, Beaumont), intermediate programming (Clark), information technology (Lambrix), microprocessor architecture (Martins), operating systems (Yi-Ran), software engineering and design (Wang, Qui, Dunlap, Huang), and theoretical CS (Hamalainen, Bednarik). Reasons include teaching generic skills such as communication, critical thinking, teamwork, problem solving, and self-learning (Shen, Ambrosio, Fekete, Barg, Lamrix, Wang, Hamalainen, Clark); the ability to help students make connection across multiple areas (Pereira); difficulties with students grasping abstract nature of content (Huang, Nuutila, Beaumont); enabling deeper learning (Okelly, Yi-Ran, Clark); relating content to real-world practice (Martins, Qui, Dunlap, Hamalainen); and a desire to improve student motivation (Ambrosio, Martins, Bednarik) and success in courses (Cavedon, Hamalainen).

What are HOT skills, why are they important in CE?

Study of skill development using PBL in CE

Many studies publishing learning outcome results focus on class-specific content learned. In that area, there is research comparing the PBL instantiations of classes to other methods and pre-and post-course exam scores. Hamalainen reports no significant difference in grades between the

non-PBL and PBL students in a theoretical CS class, but a decline in dropouts and an increase in passing rates. In addition, weak students fared well given the passing rate was much higher and a lack of prior mathematical course experience among students did not affect grades in the PBL cohort as much as the non-PBL cohort. Similarly, Clark in an intermediate Java programming class also saw a reduction in failure rates, while seeing scores rise and reports from instructors in follow-on units that students were better prepared.

In foundational CS courses Barg reports substantial improvement in basic programming ability and a better grasp of advanced topics comparing a post and pre PBL class. This improvement was thought to be due to problem-based presentation of the topic compared with a lack of student attendance and student perceptions of irrelevance of material in the lecture format of the class. In a computer engineering class, PBL students reported significantly higher scores, however, it is noted the students volunteering for the PBL approach may have been inherently more motivated. (Garcia-Robles)

Among studies examining pre and post PBL course scores, substantial improvement was reported in coding ability (Matzko) and Java knowledge (Shen). Yi-ran reports student "achievement" in a PBL-based operating systems class close to that of students who took the traditional class.

While generic skills of communication, critical thinking, teamwork, problem solving, and self-learning are mentioned as important considerations for adopting PBL in CE there has been little measurement of student exercise and development of those skills. And, in cases where results are discussed many studies rely on students' perceptions of those abilities using questionnaires (Wang), anecdotal evidence (Yi-ran), or assumption that because there was general problem

success by students certain general skills must have been exercised or learned. (Shen) This lack of study of critical thinking and problem solving specifically is in line with O'Grady's statement that instructional design in support of PBL in CE remains an open question.

Other methods to develop HOT skills during CE

Have they worked?

Why student involvement in FOSS?

Not only does a successful PBL approach require authentic problems, but industry practitioners have also called for student exposure to authentic problems indicating that such an approach during formal education would be the best method to prepare students for work in computing. It was noted that real-world types of large projects that involve technical writing, testing, version control, and exposure to design concepts and problem solving were needed. (12 Exter)

However, a common problem expressed through literature on PBL in CE has been the difficulty in developing problems. (05 Clark, 06 OKelly, 08 Huang) But, problems alone are not the only resources used in PBL. PBL requires many other resources to be successful: 1) facilitation, 2) subject guidance, 3) scaffolding, 4) production, 5) assessment. (98 Ellis) Open source software communities provide a ready-made pool of authentic problems that can be leveraged, and other technological and social elements that can aid in providing resources in support of PBL.

What is FOSS?

Note: See Intro in Faculty Development for FOSS in Education for more info on extent of FOSS.

There are many success stories in FOSS from Apache and Firefox to lesser known but no less important projects as those in humanitarian free and open source software. Sizes of projects in terms of contributors range from tens to thousands. As of 2008, a conservative estimate of the

number of active FOSS projects was 18,000 and that it was growing at an exponential rate. (08 Deshpande – The Total Growth of Open Source)

To understand the extent of FOSS use, the internet infrastructure mostly runs on open source products. For instance, Apache Tomcat is used on over 60 percent of websites. (ref) In addition, FOSS is generally free of direct cost and traditionally better than proprietary systems in terms of portability. Given the openness of code it also generally more reliable and secure; A famous saying related to open source, "Given enough eyeballs, all bugs are shallow," attests to that. (Deek says Raymond, 1998, however check source) There is some evidence to back these claims. For instance, an empirical study of MySQL showed six times less defects than similar commercial databases (Tong, T. (2004). Free/Open Source Software in Education. United Nations Development Programme's Asia-Pacific Information Programme, Malaysia).

The open source software movement grew out of a desire to keep software innovation open, by making source code generally available, much like the intellectual style of science. (Deek 08) These projects consist of self-organizing groups of distributed contributors. Generally artifacts and process are transparent and most contribution is voluntary. However, some companies pay their workers to contribute. (Mockus 02 Deek 08) These communities often use lean media to coordinate activities and share knowledge, tools such as CVS and Git for version control of code, web sites and online tutorials for instructional content, and Internet Relay Chat (IRC) and mailing lists for communication. (08 Weller)

Why would one want to incorporate FOSS as a resource in a PBL approach? (tie back to how FOSS provides not only problems but authentic problems and other resources that support PBL)

If we revisit the three main tenets of constructivism one can see that learning as a participant in FOSS is a constructivist process. This is because learning within FOSS involves: 1) deriving understanding from interacting with the environment, 2) problem-solving; and 3) developing knowledge through social interaction. First, FOSS communities are authentic software development communities; learning is not occurring in a simulated environment and participants are learning as an outgrowth of their own goals within the community. Second, participation in FOSS, which can take many forms from testing to bug fix to documentation to design and development, arises out of a real need (need to tie this better to problem solving). Third, knowledge in FOSS is socially constructed through participants' inquiries to the community and reexamination of community artifacts.

To consider at a lower level how learning occurs in FOSS one can take the three themes of constructivism-- authentic environmental interactions, challenge, and social knowledge construction—and look at those in regard to related learning theories to explicate the process and help identify the resources available by FOSS that can be leveraged as part of a PBL learning design. For instance with regard to social construction of knowledge, many studies of learning in FOSS discuss it in social-constructivist terms referring to situated learning theory and the notions of communities of practice and legitimate peripheral participation (Lave and Wenger). Often these studies are interested in newcomer learning and sustained contribution and do not specifically focusing on the cognitive aspects an individual employs as part of the learning process. However, they are important for considering how students' socialization and participation within FOSS may impact their learning when FOSS is included as part of a PBL instructional design.

Social knowledge construction/Situated learning

Social knowledge construction studies, however, take different approaches in attempting to explicate elements of the learning process. For instance, model development examining large numbers of SourceForge projects (Shaffer and Singh) to more qualitative approaches examining single large OSS projects (Ducheneuat, Fang, Sigfriddson, Berdou, Davis, vonKrogh) Shaffer attempted to document a relationship between knowledge diffusion and participation levels mining data from the top SourceForge projects, while Singh, also mining SourceForge project data, modeled learning states among developers based on experience and peer interaction. In general, Singh found regardless of learning state, developers benefitted from interaction with peers and code contribution was positively related advancement in learning states.

Similar results were also found studying qualitative and quantitative case studies of large projects. For instance, Ducheneaut in a study of the Python community also found contribution and interaction was important, however developing an identity in the community and socialization of ideas was also important, suggesting movement within OSS may not be as meritocratic as many suggest. A study of the phpMyAdmin also found that sustained participation was predicated on conceptual as well as practical contributions and identity construction. (Fang) And, from a questionnaire of over 4000 mainly male participants in the GNU/Linux community, Davis also found similar results. However, he measures the relationship between types of participation and community identification and learning. There was a significant correlation between participation overall (software participation, software contribution, communication participation) and communication and software participation and learning overall –programming, teamwork, and general computer knowledge. Furthermore, all types of participation were significantly related to identity construction.

Other research in the same situated learning vein, provide insight into community aspects that can support newcomer learning specifically. Sigfridsson examined PyPy's use of opening sprints—specialized development sessions-- to newcomers as a way to attract new members and help accelerate learning for all. In these collocated events newcomers are shown tutorials and pair programming enables experienced developers to mentor new developers. However, acculturation is also aided virtually by core developers being accessible via mailing list and IRC.

From interviews with new and experienced developers in the GNOME and KDE projects it was learned that newcomers face challenges related to learning tools used by community, developing an understanding of the development environment and architecture of the project, finding appropriate tasks to undertake, and learning how socially participate in the community. Despite those challenges those interviewed said it was a better learning experience than traditional CS courses. This echoes what Davis learned from studying the GNU/Linux project where regular contributors stated that participation was educationally beneficial. GNOME and KDE senior developers also reported characteristics they look for in newcomers they assist: 1)"self-reliance"; 2)"commitment"; 3)"productivity." It is expected that newcomers make an effort to available resources prior to seeking assistance from experienced developers.

Lastly, von Krogh et. al identified activities a joiners in the Freenet project undertook before becoming members of the developer community and labeled them as the "joining script." It was proposed that the joining script would act as a barrier to entry and communities wishing to add new members could work to lower the barrier to entry. ***Describe in more detail*** (there are elements in von Krogh's work that relate to choosing a community to join)

Individual Cognitive Processing

However, while these studies point to FOSS communities as environments with potential to support social knowledge construction, examining communities based on that aspect alone doesn't consider the role of practice in the community *and* reflection and conceptualization, or the cognitive aspect of learning. (Hemetsberger) Drawing upon experiential learning theory (Kolb, 1984), however, Hemetsburger was able to consider concrete experience and abstract conceptualization in the KDE project and the role of technology in those processes. This is also important because unlike social-constructivist theories which are based on face-to-face interactions such as … work and learning in FOSS communities is often an individual process (Howison, Hemetsburger) but coordinated/facilitated by technology.

Kolb's experiential learning theory is a holistic learning process that expresses how knowledge is generated through "transformation of experience." It involves concrete experience, reflective observation, abstract conceptualization, and active experimentation. (http://www.learning-theories.com/experiential-learning-kolb.html) Hemetsburger used the phases of this cycle to identify how technology used by KDE developers contributed to knowledge creation collectively and individually. Table 1 presents the technological tools, learning and knowledge building processes, and resulting knowledge artifacts generated by the KDE developers. These knowledge artifacts enable re-experience and action that permit learning and knowledge building to occur in FOSS communities. The use of these types of tools contributes to the ability to leverage those as resources in the PBL process.

Table 1. (source 06 Hemetsburger)

Technological Tools	Learning/knowledge building processes	Knowledge artifacts
CVS repository	Full cycle of re-experiencing: Concrete experience Reflective observation Abstract conceptualization Active experimentation	Code
Website content and hyperlinks (e.g. FAQs, content)	Productive inquiry Reflective observation	Transactive group memory
Online tutorials and screenshots	Active experimentation Reflective observation	Instructive content
IRC (Internet relay chat)	Reflective observation Collective reflection	Instructive discourse
Asynchonous communication (e.g. mailing lists, newsgroups)	Collective reflection Collective conceptualization Virtual experimentation*	Reflective discourse

*Virtual experimentation refers to ideas for experimentation programmers had.

Discuss examples above in relation to learning processes and problem solving. [Discuss Sowe 06 here, too]

Problem Solving

Discuss Lin article discussing FOSS and ill-structured problems tie that to can be leveraged in PBL ID.

06 Okelly discusses ill-structured problems and software design. Nuutila discusses nature of programming problems. Jonassen discusses characteristics of ill-structured and well-structured problems.

What is the extent of student participation in FOSS for learning?

What skills are developed? Discuss methodological concerns and lack of study of HO skills?

State of Instructional Design for FOSS participation in support of CE

While there has been student participation in FOSS for learning there is a lack of instructional design for using. Discuss lack of formalism from literature and many problems in applying.

Research gaps/methodological limitations

Here making a case to set up a case study of students participating in FOSS that follows an ID incorporating the PBL process to measure HO and other skill development.

Methodology

RQ1 - study of artifacts (mailing lists, IRC, etc.) related to participants' progression in FOSS to identify skills exercised, in particular HO. This will be used to tie FOSS activities to learning goals and to understand how to leverage the community and technology as part of PBL instructional design that will be tested to answer RQ2.

Here discuss findings from Higher Order Thinking literature review and how to consider in methodological approach.

RQ2 - Case study of PBL instructional design of student participation in FOSS. As part of an independent study students will participate in an FOSS community. Instructor will follow PBL instructional designs developed from output of study for RQ1. Assessment will include pre and post testing of higher-order thinking skills, and evaluation of activity in the community, and student learning diaries.