EXPLORING SOCIAL INTERVENTIONS FOR COMPUTER PROGRAMMING: LEVERAGING LEARNING THEORIES TO AFFECT STUDENT SOCIAL AND PROGRAMMING BEHAVIOR

By

DANIEL MICHAEL OLIVARES

A dissertation submitted in partial fulfillment of
the requirements for the degree of

DOCTOR OF PHILOSOPHY

WASHINGTON STATE UNIVERSITY
School of Electrical Engineering and Computer Science

MAY 2019

© Copyright by DANIEL MICHAEL OLIVARES, 2019
All Rights Reserved

www.manaraa.com

ProQuest Number: 13812273

All rights reserved

INFORMATION TO ALL USERS
The quality of this reproduction is dependent upon the quality of the copy submitted.

In the unlikely event that the author did not send a complete manuscript
and there are missing pages, these will be noted. Also, if material had to be removed,
a note will indicate the deletion.

ProQuest 13812273

Published by ProQuest LLC (2019). Copyright of the Dissertation is held by the Author.

All rights reserved.
This work is protected against unauthorized copying under Title 17, United States Code
Microform Edition © ProQuest LLC.

ProQuest LLC.
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106 – 1346

www.manaraa.com

© Copyright by DANIEL MICHAEL OLIVARES, 2019
All Rights Reserved

www.manaraa.com

To the Faculty of Washington State University:

The members of the Committee appointed to examine the dissertation of DANIEL MICHAEL OLIVARES find it satisfactory and recommend that it be accepted.

_____

Christopher D. Hundhausen, Ph.D., Chair

_____

Olusola Adesope, Ph.D.

_____

Assefaw Gebremedhin, Ph.D.

_____

Mark Guzdial, Ph.D.

www.manaraa.com

ACKNOWLEDGEMENTS

This work would not have been possible without the support of friends and colleagues I met during my time at Washington State University. I am especially indebted to those who have been supportive of my career and worked to provide me with guidance and feedback in support of my goals. I am grateful to all of those with whom I have had the pleasure to work during this and other related projects. My dissertation work would not have been possible without the founding and collaborative works done over the years by members of the HELP Lab at WSU.

I also want to express my gratitude to my Dissertation Committee who have provided me extensive personal and professional guidance and taught me a great deal about both scientific research and life in general. I would especially like to thank Dr. Christopher Hundhausen, the chairman of my committee, who believed in my ability to finish even when I did not.

Finally, I would not have made it to this point without the love and support of my family and their belief in my ability to achieve my goals.

www.manaraa.com

EXPLORING SOCIAL INTERVENTIONS FOR COMPUTER PROGRAMMING: LEVERAGING LEARNING THEORIES TO AFFECT STUDENT SOCIAL AND PROGRAMMING BEHAVIOR

Abstract

by Daniel Michael Olivares, Ph.D.
Washington State University
May 2019

Chair: Christopher D. Hundhausen

The 2012 report by the US President's Council of Advisors on Science and Technology (PCAST) predicts a deficit in the workforce for science, technology, engineering, and mathematics (STEM) in the following decade and emphasizes the importance of addressing this shortfall. According to the report, less than half of the three million students entering U.S. colleges yearly, as STEM majors, graduate with a STEM degree. Computer science is one of the worst-afflicted STEM disciplines: according to a large-scale longitudinal study of student persistence in U.S. colleges by the U.S. Department of Labor, just 46% of students who began an undergraduate computing degree program graduated with a computing degree.

According to social learning theory, one way to improve persistence is to help learners form and participate actively in a vibrant learning community. Building on prior online social programming environments (SPEs) research, this dissertation contributes a framework for generating interventions within an SPE that are responsive to a continuously-updated stream of learner data. The Goals, Actions, Motivation, and Standing framework (GAMS) is firmly rooted

www.manaraa.com

in relevant learning theory and best programming and learning practices derived from general and computing education literature, thus, providing a principled basis for generating interventions that can effect positive changes in learners' behaviors.

To explore the GAMS framework's potential to foster social learning in undergraduate computing education, this dissertation presents a series of empirical studies conducted over four semesters in a CS 1 course. Results suggest that students who had high levels of engagement with the GAMS-based interventions were more socially active in the SPE, and had higher performance on some course assignments, than those who had low levels of engagement with the interventions. However, overall results failed to show evidence supporting significant positive changes in student programming behavior or attitudes.

The empirical studies' results shed light on the effective design of social programming interventions using the GAMS framework. This dissertation contributes a set of recommendations for designing software-realized interventions to promote social activity in online learning environments, as well as set of best practices for using the inventions to support increased social participation in computing courses.

www.manaraa.com

TABLE OF CONTENTS

www.manaraa.com

www.manaraa.com

www.manaraa.com

www.manaraa.com

LIST OF TABLES

www.manaraa.com

www.manaraa.com

www.manaraa.com

www.manaraa.com

www.manaraa.com

LIST OF FIGURES

www.manaraa.com

www.manaraa.com

www.manaraa.com

www.manaraa.com

CHAPTER 1

INTRODUCTION

Student retention and learning in STEM disciplines is a growing problem. The 2012 report by the President's Council of Advisors on Science and Technology (PCAST) (Graham, Federick, Byers-Winston, Hunber, & Handelsman, 2013) predicts a future deficit in science, technology, engineering, and mathematics (STEM) workforce in the following decade and emphasizes the importance of addressing this issue. According to the report, less than half of the three million students entering U.S. colleges yearly, and intending to major in a STEM field, end up graduating in a STEM field.

The problem appears to be equally serious in undergraduate computer science education, which feeds into one of the fastest-growing professions in the U.S. (Bureau of Labor Statistics, U.S. Department of Labor, 2017). According to a large-scale longitudinal study of student persistence in U.S. colleges, just **46 percent** of students who began an undergraduate computer science degree program actually graduated with a computer science degree (National Center for Education Statistics, U.S. Department of Education, 2011). Thus, even if ongoing initiatives to attract more students to computer science degree programs succeed (see, e.g., Cassel, McGettrick, Davies, Topi, & Sloan, 2007; Resnick et al., 2009; Rodger et al., 2010), the retention problem still stands in the way of U.S. computer science degree programs meeting the projected growth of the profession. More recent data continues to follow this trend, with an ACM report (Stephenson et al., 2018) projecting that computing occupations will compose as many as 73% of all STEM jobs (Lazowska, 2016), while only 10% of STEM graduates will major in computer science ("U.S.

www.manaraa.com

Department of Education, National Center for Education Statistics, Integrated Postsecondary Education Data System (IPEDS), Completions Survey," 2017).

The importance of addressing this issue is stressed in a literature review by Christe and Feldhaus (2013), which issues a call- to-action to encourage instructors to make changes to improve student retention rates. Christe and Feldhaus (2013) argue that *student learning experiences* are a key factor influencing the notoriously low student retention rates in STEM disciplines. Student interaction, perception of instructors, and social interaction can all have an effect on student performance and retention (Vogt, 2008). In fact, a 1997 study by Vincent Tinto (Tinto, 1997) indicated that students put more effort into educational activities that bridge social and academic divides and tend to be more successful in the end. That is, learning communities that provided a way to bridge the gap between the social and academic to meet the needs of both result in higher peer and learning activity scores, which can be linked with increased persistence. This suggests that a reasonable starting point for addressing the retention issue might be to find ways to provide students with more positive learning experiences.

1.1 A Possible Solution: Formative Assessment Tools

How might students' learning experiences be improved? One possibility is through *formative assessment techniques*, which incrementally check student learning and attitudes in order to provide a basis for adjusting the teaching and learning process. Implementing formative assessment techniques is challenging for instructors because of the time and effort required. Nonetheless, formative assessment has been shown to provide great value when applied during the learning process in order to modify and adjust teaching and learning processes (Black & Wiliam, 1998; Wilson & Scalise, 2006).

www.manaraa.com

In modern learning environments, there are many formative assessment tools available to instructors. These include both online (web-based) tools and computer applications used as an adjunct to course instruction (Beldarrain, 2006). Instructors can make sense of the data generated by learners[1] with the help of *learning analytics* (Ferguson & Shum, 2012), which provides information about students' learning processes and progress, thus providing an empirical foundation on which adjustments to teaching and learning can be made.

A range of information gleaned from formative assessment and learning analytics tools might be used to affect student retention and learning. Both internal and external influences can predict success (Rotter, 1966; Weiner, 1979), including a learner's use of learning tools, time on task, and social activity (Chickering & Gamson, 1987). Thus, one potential solution to address the learning and retention problem would be to collect learning data on students, and then use these data to better tailor students' learning experiences.

With the plethora of student learning data that might be collected and leveraged, where does one start? Focusing on computer science education, this dissertation explores the design space of *learning analytics tools* for introductory computer science courses, which traditionally have the highest attrition rates among all computer science courses (Bennedsen & Caspersen, 2007; Watson & Li, 2014), and hence are the place where solutions to the retention problem could have the greatest impact. This focus on leveraging learning analytics tools to positively influence student learning experiences in early courses raises two overarching research questions:

[1] In the context of this dissertation, learners and students will be used interchangeably.

www.manaraa.com

RQ: What information is needed to assess computer science student performance and tailor instruction to students' needs?

RQ: What interventions [2] can leverage *social programming environment* [3] data to improve learning and retention in early undergraduate computer science courses?

There are many known factors which might affect student learning (e.g. (Barker, McDowell, & Kalahar, 2009; Yu & Jo, 2014)). Students come into introductory computer science courses for different reasons: major requirements, minor requirements, elective credit, and even just curiosity. Their experience and intent at this introductory level can have a huge impact on whether they decide to continue down the path of computer science education. With that in mind, each of these research questions requires further exploration.

1.1.1 RQ: What Information is Needed to Assess Learning and Tailor Instruction?

Educators use a variety of assessment tools to gauge student learning in today's undergraduate courses. The methods used are both formative (e.g. reflections, polling, and checks of understanding via activities, quizzes, and assignments) and summative (e.g. performance milestones like exams and project deliverables). All of these approaches can be used as a basis for tailoring learning and instruction. Using fine-grained assessment methods can provide too much information and add additional workload for an instructor (Wilson & Scalise, 2006). At the same time, summative scores (e.g. grades from exams) alone are not enough to aid learners (Wilson & Scalise, 2006). Given increasing class sizes, efforts to automate assessment are on the rise.

[2] In the context of this dissertation, an intervention will be defined as an event in which some combination of information, data, guidance, and feedback is shared with a learner for the purpose of positively influencing the learner's attitudes and/or trajectory.

[3] In the context of this dissertation, a social programming environment (SPE) is defined as a problem-solving environment which includes integration of the *social environment* (e.g. the course management system) and the *programming environment* (e.g. an integrated development environment that allows one to write, execute and debug computer code).

www.manaraa.com

However, these have both pros and cons (J. Carter et al., 2003; Del Fatto et al., 2018; Pieterse & Liebenberg, 2017). Clearly, it is essential to strike an appropriate balance between automated and manual assessment and intervention.

Because undergraduate computer science students spend much of their time working on programming assignments in computer programming environments (a.k.a. *integrated development environments,* or IDEs), computer science educators have a golden opportunity to automatically collect detailed log data on computer science students' programming processes, including their edits, compilations attempts, compilation errors, execution attempts, and runtime errors. Two prominent examples of IDEs instrumented with automated data collection tools are the BlueJ environment (Kölling, Quig, Patterson, & Rosenberg, 2003), and the OSBIDE (Online Studio-Based Integrated Development Environment) plug-in to Microsoft Visual Studio (Carter & Hundhausen, 2015). Building on environments like these, one line of research in computer science education has built predictive models of student achievement based on programming process data [see, e.g., Error Quotient (Jadud, 2006), Watwin score (Watson, Li, & Godwin, 2013, 2014), and NPSM (Carter, Hundhausen, & Adesope, 2015)].

While these predictive models have been modestly successful, they primarily focus on only one aspect of the learning process (programming behavior), without considering affective or social aspects of the process that could be equally indicative of student success (Carter & Hundhausen, 2015). Motivated by social learning theory, researchers in the HELP lab (with which I am affiliated) have collected and analyzed social data in addition to programming data. Carter and Hundhausen (2015) have developed a *social programming environment* (SPE) that integrates a social networking-style activity stream into an IDE, thus enabling asynchronous discussions about computer programming to take place in the same context in which students do computer

www.manaraa.com

programming (see Figure 1.1). The SPE allows for the automatic collection of social data, in the form of asynchronous threaded discussions, alongside programming data. This has led to the development of promising new predictive models of learner success based on both social and programming data (Carter & Hundhausen, 2015). It also opens new possibilities to leverage social data to perform formative assessment and to better tailor instructional interventions in the learning process as evidenced by use of social behavior in predictive models developed by HELP Lab members (see Carter (2016) and Section 2.3).

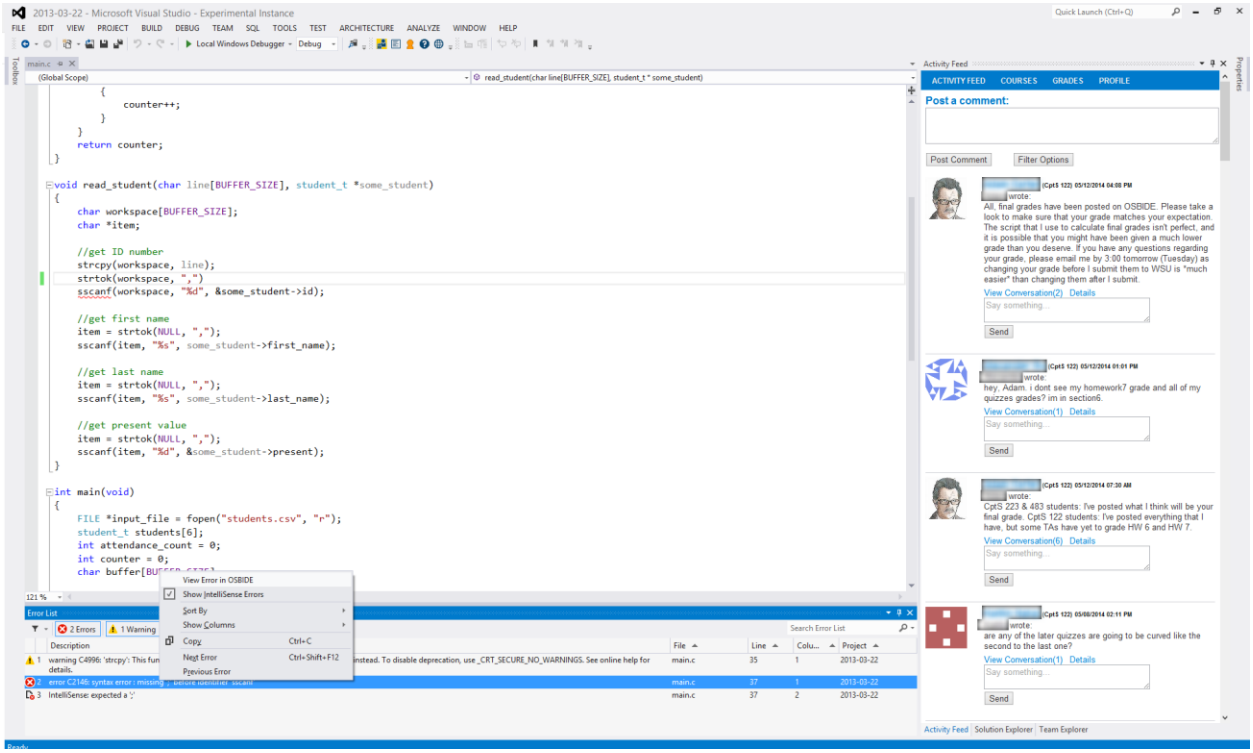

Figure 1.1. The OSBIDE SPE

This dissertation builds on this exciting new possibility. The social learning theory literature (Astin, 1999; Bandura, 1990; Kolb, 1984; Rotter, 1966) has shown that we can correlate certain student social and attitudinal data with desired behaviors (e.g. self-efficacy, social activity, level of community involvement). Hence, collecting data which predicts or can be correlated with these

www.manaraa.com

desired behaviors holds promise in developing interventions that can positively influence student learning and retention.

1.1.2 RQ: What Interventions Should be Used?

Programming and social data could be used as a basis for influencing the actions of both instructors and learners. Given that instructors already tend to be overburdened (Wilson & Scalise, 2006), I have opted to focus on exploring *interventions* to influence the actions of learners. In this dissertation, an *intervention* is defined as an event in which some combination of information, data, guidance, and feedback is shared with a learner for the purpose of positively influencing the learner's attitudes or trajectory. Moreover, while one can imagine a variety of delivery mechanisms for such interventions, I have restricted my focus to *automated, personalized* interventions (derived from learner data collected by the system) that can be delivered to learners through the same online learning environment used to augment a face-to-face course. This choice reflects both the reality that many face-to-face undergraduate courses maintain a strong online presence [e.g. see ever increasing interest in massive open online courses (Dasarathy, Sullivan, Schmidt, Fisher, & Porter, 2014; Del Fatto et al., 2018)], and the reality that the undergraduate courses to which I had access for this research are face-to-face courses augmented with online learning tools.

Given this focus, my dissertation explores the development of effective interventions for promoting student learning and retention in early undergraduate computer science courses. While this work focuses on undergraduate computing courses, its results may also apply to early courses in other STEM disciplines. As a foundation for this exploration, I use guiding learning theory that accounts for how learning and retention occur. To this end, I will start by drawing on strands of social learning theory. Astin's *theory of student involvement* (Astin, 1999) postulates that student

www.manaraa.com

involvement in the community is a critical factor for success; student learning and personal development are directly proportional to the quality and quantity of a student's involvement in an academic program and community.

Bandura's *social cognitive theory* and *self-efficacy* (Bandura, 1997) states that a belief in one's abilities has a significant impact on one's ability to succeed and that an one's actions and reactions, which include social behavior and the cognitive processes, are nearly always affected by actions that one observes in others in their community (through observational learning and social experiences). Kolb's experiential learning theory (Kolb, 1984) is centered around a four stage cyclical process of learning and reflection which emphasizes learning through doing. Rotter's *locus of control theory* (Rotter, 1966) suggests that student involvement in learning can be influenced by the student's belief in how his or her behavior can be controlled by internal or external factors (e.g. "I can affect my success" vs "my success is out of my control").

These learning theories are complimented with Vygotsky's Zone of Proximal Development a.k.a. *ZPD* (Vygotsky, 1978), and scaffolding theory (Wood, Bruner, & Ross, 1976), which stipulate that learners in a specific "zone" (i.e. phase of the learning process) are more likely to be affected by interactions with more skilled persons (e.g. peers or instructors). Scaffolding compliments the ZPD by providing guidance in the problem-solving environment to help learners push past points of frustration or being stuck.

Inspired by this theory and research, my dissertation tries to take the next logical step. Using these learning theories as a guide, interventions designed in my dissertation explore how to positively affect and promote social interaction, sense of community, and involvement within a computer programming environment. These theories suggest that by interpreting student programming and social data, we can help learners to visualize and reflect on their goals and where

www.manaraa.com

they are situated in the learning process. We can also provide guidance regarding what actions they can take in order to improve their learning success. In turn, students will be more likely to persist in the discipline; student retention in early computer science courses will increase.

1.2 Thesis

This dissertation argues that personalized interventions delivered directly to computer science students through a programming environment will positively influence their social interactions, help seeking, and help giving behavior.

In order to develop the most effective interventions, this dissertation systematically explores the design space of personalized interventions derived from programming and social data collected within a social programming environment (SPE) that integrates a social networking-style activity stream into an integrated development environment (IDE) in which students write, compile, execute and debug code (Carter & Hundhausen, 2015). Through a series of *formative* studies, interventions were iteratively designed. A series of *summative* evaluation studies were performed to test hypotheses related to the effectiveness of the interventions and their ultimate effect on student learning outcomes and attitudes. The results of these studies address the following revised research questions:

RQ1: Can the SPE interventions promote *positive changes*[4] in students' *social* and *programming* behaviors in the OSBLE+ environment?

RQ2: Will the SPE interventions lead to *positive changes*[4] in students' learning outcomes, and persistence within the computer science discipline?

[4] Positive changes are defined as changes which can be associated with known metrics defining success. For example, social behaviors supporting *vicarious* and *enactive* engagement (Bandura, 1997) or programming behaviors and states linked to predicted success (Carter, Hundhausen, & Adesope, 2015).

www.manaraa.com

To address these questions, we needed a solid technological foundation for collecting student social and programming data. To that end, this dissertation is organized as follows:

Chapter 2 will discuss the background learning theories and foundational related works more in-depth. Chapter 3 presents a primary contribution, and basis for much of the work done in this dissertation, in a framework for the design of SPE interventions. Chapter 4 provides an overview of the iterative design process explored to develop the social programming interventions tested and discussed in Chapters 5-9. Chapter 10 follows with implications for design based on the results of Studies I-IV presented in Chapter 5-9. Finally, the dissertation concludes with contributions, limitations and future work in Chapter 11.

1.3 Published Works from This Dissertation

The following are published works that came as a result of my dissertation work:

1. Using social network analysis to measure the effect of learning analytics in computing education (Olivares et al., in press)
2. IDE-Based Learning Analytics for Computing Education: A Process Model, Critical Review, and Research Agenda (Hundhausen, Olivares, & Carter, 2017)
3. Supporting learning analytics in computing education (Olivares & Hundhausen, 2017)
4. OSBLE+: A Next-Generation Learning Management and Analytics Environment for Computing Education (Olivares & Hundhausen, 2016)
5. Exploring Learning Analytics for Computing Education (Olivares, 2015)

www.manaraa.com

CHAPTER 2

BACKGROUND AND RELATED WORK

In traditional computer science courses, students are required to work individually on programming assignments, to prevent cheating and ensure that individual students put in the effort required to learn how to program. Accordingly, the programming environments where computer science students spend much of their out-of-class time have traditionally been tailored for individual work; they tend not to support student interaction or collaboration. Indeed, in a comprehensive review of novice programming environments, Kelleher and Pausch (2005) found that, of the 80 systems they reviewed, only 11 were designed to accommodate some kind of social interaction: side-by-side programming (a.k.a. pair programming), remote collaborative code manipulation, or (remote code sharing). Kelleher and Pausch conclude that this is a key limitation of existing programming environments and argue that future research is needed to explore the design of programming environments with greater social support.

Beyond Kelleher and Pausch's (2005) call-to-action, social learning theory, coupled with the empirical results that relate social engagement with learning outcomes in computer science education, motivate the need to explore the design of programming environments with strong support for social interaction. Research studies within computer science education lend further credence to these theoretical orientations. In an exploratory survey by Barker et al. (2009) involving 113 freshmen and sophomores who had taken an introductory programming course, a regression analysis of the results showed that student-student interaction was the *strongest* factor in predicting intention to major; it was shown to be stronger than other factors such as collaborative learning and classroom climate/pedagogy, which are known to influence persistence. Likewise, in

www.manaraa.com

a survey of undergraduates in Penn State's College of Information Sciences and Technology (IST), a program in which students are offered a curriculum similar in many ways to traditional computer science programs, Rosson et al. (2011) found that *self-efficacy* and *social support* were significant predictors of orientation towards careers in computer science.

Given that social engagement has been positively linked to learning and retention, my dissertation explores the design space of interventions embedded within an SPE that can promote increased social engagement. The context of this exploration, as mentioned earlier, will be a traditional face-to-face undergraduate computer science course centered on a series of individual programming assignments. While no prior computer science education research has attempted to design interventions specifically geared toward promoting social engagement within a programming environment during the programming process, there are several lines of related work on which my research builds. In this chapter, I review this work.

The following section gives a high-level overview of the six theories motivating my research. Section 2.2 provides an overview of relevant pedagogical approaches within computer science education that are designed to foster *social* learning. Section 2.3 looks at predictive models of learner success and retention in computer science education based on learners' behaviors and attitudes; such models can be used as a foundation for designing the kinds of effective interventions envisioned by this research. In Section 2.4, I turn my attention to the literature on intelligent tutoring systems (ITSs), which use dynamically-constructed models of learners' progress within a solution space as a means of guiding learners toward effective learning behaviors and correct solutions. Section 2.5 then reviews several existing IDEs that are designed to help students collaborate. Finally, section 2.6 reviews relevant work in the related fields of educational data

www.manaraa.com

mining and learning analytics, which aim to help instructors and learners better tailor their teaching and learning based on continuously-collected educational data.

2.1 Theories of Learning

Learning theories provide the major motivation and grounding for this dissertation, which aims to leverage student social and programming data to intervene in a manner that improves learning outcomes by stimulating social interaction. The most influential of these learning theories are considered below.

2.1.1 Astin's Student Involvement Theory

Astin's Student Involvement Theory (Astin, 1999) builds on the principle that student involvement in the community is a factor of success. The quantity and quality of the energy that students invest (i.e. their level of involvement) in the college experience underpin Astin's student involvement theory. Specifically, the theory postulates that student learning and personal development are directly proportional to the quality and quantity of a student's involvement in an academic program and community. Student involvement in the community is seen as a critical factor of success; therefore, prompting an increase in involvement may lead to an increase in student involvement and (ideally) improved student outcomes.

2.1.2 Bandura Social Cognitive Theory/Self-Efficacy

Bandura's Social Cognitive and Self-efficacy Theory (Bandura, 1990, 1997) posits that learners' actions and reactions, including their social behavior and cognitive processes, are affected by actions that they observe in others in their community. That is, in the learning process, learners will observe their peers and potentially act based on their perceived experiences (observational learning). They will learn from each other both from observing, imitating others

www.manaraa.com

(what Bandura terms "vicarious" experiences), and from having opportunities to model their own behavior accordingly (what Bandura terms "enactive" experiences). Therefore, if we aim to improve learning experiences, we ought to improve learners' interaction with and awareness of their peers.

2.1.3 Experiential Learning Theory: Kolb's theory

Kolb's theory, or experiential learning theory (Kolb & Kolb, 2005; Kolb, 1984), focuses on the idea that *self-reflection* is an important part of the learning process and that including self-reflection in this process can increase motivation and participation. Further, it posits that the effectiveness of learning can be improved when one goes through a cyclical four state process:

1. A concrete experience
2. Reflective observation of that experience
3. Abstract conceptualization, i.e. analysis and conclusions
1. Active experimentation to test hypotheses and introduce new experiences

Through this reflective learning process, experiential learning theory affects learners by helping learners take control and responsibility for their learning and prompting self-authorship and learning from experiences.

2.1.4 Rotter's Locus of Control

The locus of control, as defined by Julian Rotter (1966), is a framework which suggests that learners' involvement in learning (i.e. interactions with their environment) can be influenced by their beliefs in how their behavior can be controlled—whether by internal or external factors. An internal locus of control suggests that one has control over one's behavior and can act to affect an

www.manaraa.com

outcome, whereas an external locus of control suggests one sees oneself as powerless to act; the outcome is seen as being governed by external factors.

2.1.5 Zone of Proximal Development & Scaffolding

The zone of proximal development (ZPD) is defined as the difference between a learner's development level (i.e., current independent problem solving ability) and potential development level with the aid of guidance and/or peer collaboration (Vygotsky, 1978). The application of ZPD theory posits that when learners are in this zone, interaction with those more skilled (instructors or peers) can be an effective way of helping develop skills and strategies. Scaffolding, though not a specific part of ZPD theory, is a method of providing (again by instructors or peers with more knowledge than the learner) activities or guidelines through the ZPD (Wood et al., 1976). The intent is to help learners frame their questions and/or provide guidance for their problem solving in a way that will enable them to get help more efficiently. ZPD and scaffolding do so by giving learners enough resources to get them past periods where, without the guidance, they may become stuck and frustrated. This is usually done by providing some level of minimal support, partial solutions or directions, and other supported learning methods.

2.2 Social Learning Pedagogies in Computer Science Education

Pedagogies that aim to foster social interaction are the most relevant to the work of this dissertation. I define "social interaction" as an exchange between two or more individuals. This exchange can range from basic communication to the sharing of complex ideas and does not necessarily need to be categorically related to course content (i.e., it includes so-called "off-topic" discussion).

www.manaraa.com

There are several pedagogical approaches that situate learning around social interaction. One prominent example is Problem-Based Learning (PBL). While PBL has perhaps been most successfully used in the medical field (Schmidt, Molen, Winkel, & Wijnen, 2009), it can be adapted for use in just about any learning domain. PBL situates the learning experience in the context of a specific problem. Within that context, learners develop problem solving strategies, disciplinary knowledge bases, and skill by acting as active problem solvers. Within computer science education, there have been a number of efforts to employ PBL [see (O'Grady, 2012) for a review]. For example, the web-based IDE JavaWIDE (Jenkins et al., 2012) can be seen to embrace PBL in that it aims to promote social interaction through having students work together towards solving a problem. Students are given a leading question to guide their creation of a Java code solution, directed to brainstorm, and encouraged to develop the solution together. In the process, students are guided by a set of prompts related to core course concepts, with the instructor acting as a facilitator and motivator.

Studio-Based Learning (SBL) is another socially-oriented pedagogical approach [see, e.g., Schön (1983)]. The foundation of architecture and fine arts education, SBL engages students in design projects that are completed within a shared "design studio" space, where students build camaraderie through working on the projects side-by-side (Boyer & Mitgang, 1996). Periodic design critiques (a.k.a. "design crits") provide opportunities for students to present their design solutions to their peers and instructors for feedback and discussion. There have been a number of attempts to adapt the "design crit" component of the SBL model for computer science education [for a review, see (Carter & Hundhausen, 2011)]. For instance, an early version of OSBLE, the technology on which this dissertation builds, was designed to facilitate both face-to-face and online "design crits" of computer code solutions (Hundhausen, Agrawal, & Ryan, 2010). In empirical

www.manaraa.com

studies, these so-called *pedagogical code reviews* were found to promote a positive sense of community and positive shifts in self-efficacy (Hundhausen, Agrawal, & Agarwal, 2013). More recently, Carter and Hundhausen (2015) have explored ways to create an online approximation of the "design studio" component of SBL in early computer science courses. By integrating a social networking-style activity feed into the IDE, their OSBIDE tool, they attempted to provide students with a sense of a shared community while engaging in individual programming assignments. Even more recent works (Braun, Brookes, Hadgraft, & Chaczko, 2019; Prior, Laudari, & Leaney, 2019) show continued interest and success in exploring SBL environments in computing education.

While there are examples of socially-oriented pedagogical approaches that are well-suited for face-to-face instructional situations [see, e.g., peer learning (Porter et al., 2016; Porter, Lee, Simon, & Zingaro, 2011) and pair programming (McDowell, Werner, Bullock, & Fernald, 2006; Radermacher & Walia, 2011)], many computer science educators have used online tools to make the implementation of socially-oriented pedagogical approaches more practical in an online environment. Aside from the online tools for PBL and SBL described above, a number of tools have been developed to support *peer review*, in which students are tasked with assessing the work of their peers (see, e.g., Gehringer, 2001; Gehringer, Ehresman, & Skrien, 2006; Hyyrynen, Hämäläinen, Ikonen, & Porras, 2010; Reily, Finnerty, & Terveen, 2009). This line of work shares this dissertation's interest in building online tools to promote the sharing and discussion of students' programming solutions. However, my dissertation research differs in that, in addition to facilitating student discussions about each other's programming activities and solutions, it uses students' learning process data dynamically to guide them toward learning behaviors that are correlated with successful learning and retention.

www.manaraa.com

2.3 Predicting Student Success Based on Learning Behavior and Attitudes

Within computer science education, a few researchers have aimed to build predictive models of learning success based on programming process data collected within the IDE. These include the Error Quotient (Jadud, 2006), the Watwin score (Watson et al., 2013, 2014), and the Normalized Programming State Model (NPSM) (Carter et al., 2015). The Error Quotient and the Watwin score both focus on the extent to which, and speed with which, students can remove compilation errors from their programs. These models use a point system to score transitions between compilation attempts that yield and do not yield errors and correlate the scores with students' course grades. These models have enjoyed modest success: the Error Quotient has been shown to account for 19-25% of the variance in students' course grades (Jadud, 2006; Watson et al., 2014), while the Watwin has been able to account for between 36-42% of the variance in students' course grades (Watson et al., 2013, 2014). In contrast, the NPSM goes beyond compilation behavior by considering students' progress through a *state model* based on the syntactic and semantic correctness of their programs. Through such a state-based model, the NPSM has been able to account for between 36-45% of the variance in students' grades (Carter et al., 2015). In his recent dissertation research, Carter (2016), expanded the NPSM to include online social participation (within an SPE) in addition to programming behavior (SNPSM). The expanded SNPSM model accounted for between 4-12% more variance in students' grades than the original NPSM.

In a similar vein, researchers have aimed to predict student success in computer science based on attitudinal data collected within computer science courses. For example, an exploratory survey

www.manaraa.com

by Barker et al. (2009) involving 113 freshmen and sophomores who had taken an introductory programming course showed that student-student interaction was the *strongest* factor in predicting intention to major. In a similar vein, in an investigation of predictive factors between attitudinal factors and learners' orientation toward majoring in computer science, Rosson et al. (2011) identified a strong connection between self-efficacy, social support, and orientation towards careers in computer science. Likewise, Lewis et al. (2011) used a survey of 31 introductory programming students to identify five factors that influence students' intention to major: ability, enjoyment, fit, utility, and opportunity cost. Interestingly, social interaction was shown to play a major part in student intent to major. One of Lewis's study findings is that students evaluate their level and relevance of experiences (i.e. the five factors) relative to that of their peers. Finally, in examining the attitudes and beliefs underlying why students decided to enter or leave the major, Biggers et al. (2008) identified a number of influencing factors which can affect computer science retention. A key finding was that "Students who left the CS major have an overwhelming perception that CS is an asocial, coding-only field with little connection to the outside world" (Biggers et al., 2008, p. 405).

In sum, while predictive models focused on programming and social behavior have shown promise in predicting academic success, students' attitudes, especially as regards their social interaction with peers, are revealing when it comes to predicting their intention to persist in an academic field such as computer science. Thus, it is vital to consider both learning behaviors and attitudinal data in any attempt to improve student learning and retention in computer science.

www.manaraa.com

2.4 Providing Students with Dynamic Guidance Based on Learning Data

Intelligent tutoring systems (ITS) are computer-based learning tools that provide learners with automated guidance on learning tasks. They do this by drawing inferences from student learning processes, using those inferences to model student learning so that guidance can be dynamically adapted to students' needs (Ma, Adesope, Nesbit, & Liu, 2014). ITSs typically position learners within a known problem space based on the learners' interactions with the ITS. The ITS can then provide appropriate guidance toward productive learning paths within that problem space.

Learner process data gathered by an ITS might include data relating to indirect interaction with the environment (Did a user read information and how much time did they spend?), direct interaction (replies to system queries), milestone completion (e.g. quiz results), or even understanding checks (e.g. "I do/do not understand this concept"). The responses to this data might include feedback from the system in some manner: whether directly to the student (automatically generated) or to the instructor (allowing an instructor to address misconceptions) (Butz, Hua, & Maguire, 2006).

There have been attempts to build ITSs specifically focused on computer science education For example, Butz et al. (2006) developed an interactive, adaptive ITS to help computer science students navigate course material. The ITS offered tailored pedagogical options or recommendations based on student data to tailor the learning environment to the needs of individual students. Meta studies of ITSs indicate that they can be highly effective in the learning process. A review by Nesbit et al. (2014) of the effectiveness of ITSs in computer science education found that their use was associated with significantly greater achievement than teacher-led group instruction and non-ITS computer-based instruction. This result is borne out in a much

www.manaraa.com

larger meta-analysis if ITSs covering eight subject domains (including computer science) by Ma et al. (2014), which concludes that ITSs outperformed the other modes of instruction against which it was being compared. That study also found that the use of ITSs was associated with significantly higher achievement outcomes than using each of the other methods of instruction except for small-group and individual human tutoring.

Based on these results, we see there is a great benefit to be gained from developing computer environments that make use of learner data to provide *dynamic* guidance in the learning process.

This dissertation takes an approach like that of ITSs. It continuously collects learner data and, based on the context of those data, generates interventions designed to influence student behavior. A key difference between my dissertation work and ITSs is that the technology does not possess an exact solution to a given programming problem, and hence is not aware of a correct solution path. Moreover, unlike ITSs, which rely heavily on individual learning process data within a constrained problem-solving environment, my research focuses on *social* data in an unconstrained problem-solving environment (an IDE). Hence, my research bases interventions on known connections between social behavior, academic success, and retention, rather than on known paths to a particular problem solution.

2.5 IDEs that Support Social Interaction

A number of IDE tools and extensions have been created to help students collaborate during the programming process. For instance, Collabode (Goldman, Little, & Miller, 2011), JavaWIDE (Jenkins et al., 2012), and Saros (Salinger, Oezbek, Beecher, & Schenk, 2010) all support real-time co-editing of program code in a web browser. However, because these tools focus on allowing multiple programmers to program collaboratively (as in *pair programming*; see, e.g., McDowell

www.manaraa.com

et al., 2006; Radermacher & Walia, 2011), they prove inappropriate for the social programming context targeted in this dissertation, in which students interact as they work on their own *individual* solutions to programming assignments.. It is important to note that this dissertation does not claim to foster better forms of collaboration than alternative approaches such as pair programming. Rather, it aims to foster social interaction within an SPE and to show that such interaction positively influences student learning.

HelpMeOut (Hartmann, MacDougall, Brandt, & Klemmer, 2010) attempts to promote interaction in a different manner: via a *social recommendation system* based on a database of possible solutions to programming errors gleaned from IDE data collected on a community or programmers. In an evaluation study, novice programmers found useful fixes for up to 47% of errors. Crowd::Debug (Mujumdar, Hallenbach, Liu, & Hartmann, 2011) expands on the work of HelpMeOut by adding support for dynamic, interpreted web development languages. The results of this work showed a marked improvement, with up to 57% of the proposed fixes being marked as useful. These tools differ from my dissertation research in that, while they draw from the actions of a community or programmers, they do not support or encourage social interaction among community members.

Another flavor of socially-oriented IDEs aims to promote social interaction via peer code review. CaptainTeach (Politz, Patterson, Krishnamurthi, & Fisler, 2014), is a web-based peer code review system intended to help students refine their programming assignments before submission. It does so through sequential, reviewable deliverables with the use of "design recipes" which break the problem down into smaller pieces, each with concrete artifacts that can be verified. Code in CaptainTeach is reviewed via a 6-point Likert scale (agree-disagree) and free form text fields. Alternately, commercial tools such as SmartBear (SmartBear Software, 2012) enable users to

www.manaraa.com

perform real-time code review via a shared code base. Tools like SmartBear are designed to be used by programming teams as they work together on a project.

A final prominent example of an IDE designed to promote social interaction is Scratch (Brennan, 2009; Maloney, Resnick, Rusk, Silverman, & Eastmond, 2010), which focuses on building a development and learning community through collaboration and the sharing of code. While my work shares Scratch's interest in building a community around programming practices, it differs from that in two key respects: (a) it focuses on undergraduate computer science education instead of K-12 education, and (b) it focuses on providing learner-customized interventions to encourage social behaviors within a learning community during the programming process.

2.6 Helping Learners and Instructors Better Tailor Their Teaching and Instruction

Learning analytics, as previously defined, provides information about students' learning processes and progress, thus providing an empirical foundation on which adjustments to teaching and learning can be made. Verbert (2013) proposes a process model to characterize the use of learning analytics a part of the learning process (Verbert, Duval, Klerkx, Govaerts, & Santos, 2013; Verbert et al., 2014). The process (see Figure 2.1) consists of four steps: (1) learning data collected and visualized to students, which (2) allows reflection and questions, (3) motivates sense making and answers, and (4) results in a positive impact on student learning behavior. In their survey of analytics used in learning dashboards, Verbert et al. conclude that impact is hard to demonstrate in the surveyed literature, but that there is much promise. They call for more research to be done in this area, specifically into the usefulness of different data types, collection of new types of data,

www.manaraa.com

data collection setting, and evaluation of the learner data. The research of this dissertation focuses on one data collection setting; a *social programming environment.*

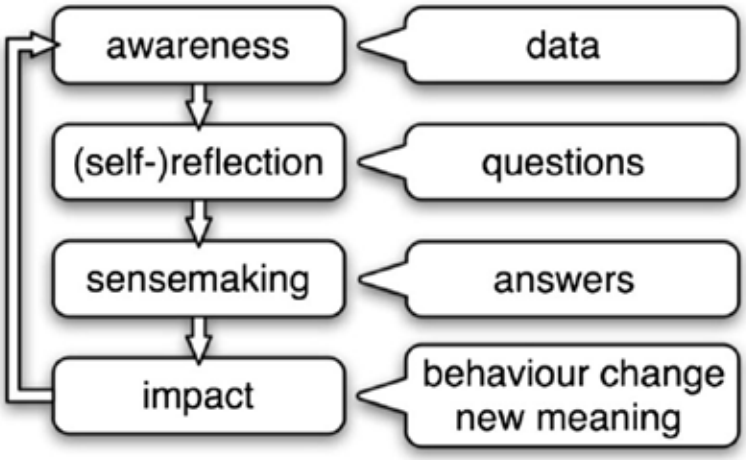

Figure 2.1. Learning analytics process model

Learning analytics *dashboards* attempt to help learners and instructors make sense of the abundance of learner data that are increasingly available [e.g., see Figure 2.2. The student activity meter (SAM) (Verbert et al., 2014)]. These dashboards can be used to augment face-to-face teaching, online learning, or even blended learning settings (Verbert, Duval, et al., 2013). They can be part of general use LMSs such as Moodle, custom LMSs as seen in (Rößling et al., 2008), or part of a student centered dashboard that makes use of both learning analytics and formative assessment (Aljohani & Davis, 2013). Classroom Salon (Barr & Gunawardena, 2012), a dashboard with a focus on social networking, provides an environment in which programming students can collaborate in editing and commenting on code (Figure 2.3). Though code cannot be run or compiled in this environment, it does provide a social environment in which computer science students can cooperatively create, comment on, and modify documents. For instructors, the dashboard provides a "dashboard charts" feature which helps visualize statistical information about document annotations.

www.manaraa.com

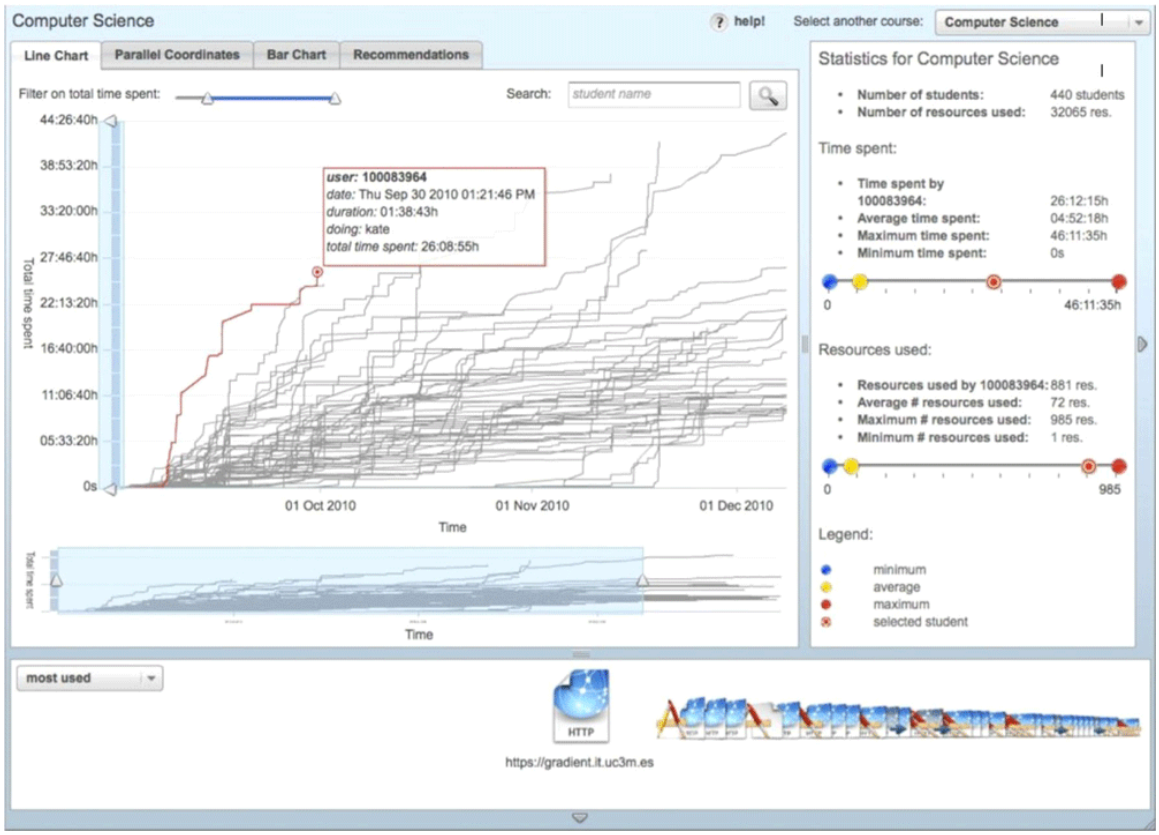

Figure 2.2. The student activity meter (SAM)

Figure 2.3. Classroom Salon

www.manaraa.com

While my dissertation work draws heavily from the kind of learning analytics dashboard provided in Classroom Salon, it differs in that I integrate a dashboard into a combined social and programming environment, thus allowing the interventions to be based on both social and programming behaviors.

2.7 Summary

The preceding review makes a strong case that improved social interaction in the learning process can benefit computer science learners. We have seen that there are benefits to socially oriented pedagogies (e.g. Problem-based and Studio-based learning), and that we can predict some level of student success based on student programming behavior, social behavior, and attitudes. The literature also shows that there are clear benefits from systems designed to automatically guide learning and that there are benefits to IDEs that have a social and collaborative focus. Finally, this literature review has shown that the dynamic collection of learner process data (as in intelligent tutoring systems), and the presentation of those data to learners and instructors (as in a learning analytics dashboard) can be leveraged to positively affect the student learning process.

This literature review has also identified opportunities for future research. Most notably, while there have been many attempts to make the learning process in computer science education more socially-oriented, there has yet to be an attempt to leverage continuously-collected data on students' learning and social processes in order to provide individually-tailored interventions that help students become more socially active— which, according to the social learning theories described earlier, should lead to improved learning outcomes and retention. It is this endeavor that I take on within this dissertation.

www.manaraa.com

CHAPTER 3

A FRAMEWORK FOR THE DESIGN OF SPE INTERVENTIONS[5]

The review of the previous chapter underscored the potential value of leveraging continuously-collected learning process data to positively impact student learning and retention. But how might educationally effective interventions based on such data be designed? In this chapter, I present a key contribution of this dissertation: a conceptual framework, firmly rooted in social learning theory, for generating dynamic SPE interventions based on learning data collected continuously within an SPE.

Figure 3.1 presents a diagram of the intervention generation process within an SPE. At the center of this process is *GAMS* (Goals, Actions, Motivation, Standing), a conceptual framework for transforming continuously-collected SPE data (the inputs) into a set of SPE Interventions (the outputs). In the remainder this chapter, I elaborate on three core components of Figure 3.1 in greater detail. Section 3.1 describes the *inputs* to GAMS in the form of a taxonomy of observable behaviors within an SPE. Section 3.2 describes the *Intervention generator*, GAMS, which is rooted in the guiding learning theories discussed throughout this dissertation (section 1.1.2). Finally, in 3.3, I describe the design dimensions of the outputs of the process—interventions delivered to learners through the SPE.

[5] The contents of this chapter were used, in part, for the creation of a journal article published in ACM Transactions on Computing Education (TOCE) - Special Issue on Learning Analytics: Volume 17 Issue 3, August 2017 (Hundhausen, Olivares, & Carter, 2017)

www.manaraa.com

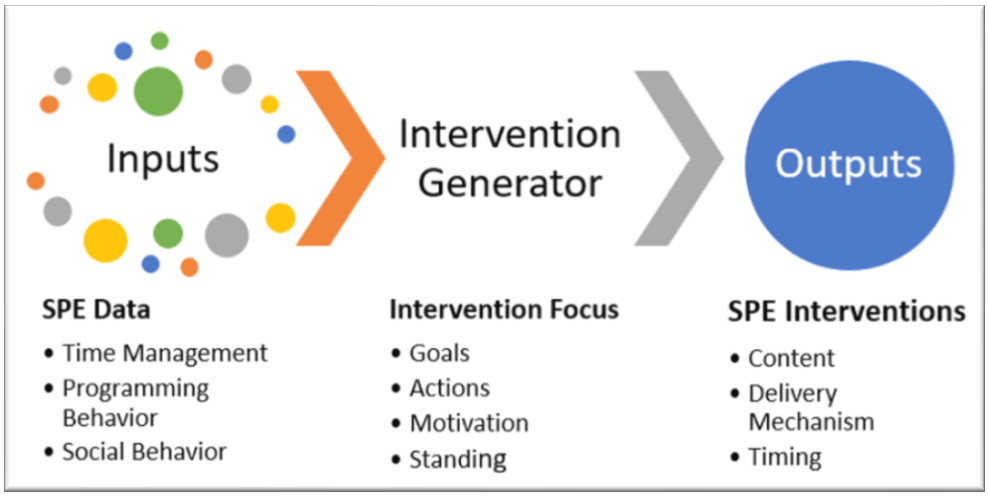

Figure 3.1. GAMS conceptual model

3.1 Taxonomy of Observable Behaviors Within an SPE[6]

The advent of data collection tools embedded within online learning environments (e.g. through *learning analytics* tools and *data mining*, as discussed in section 2.6) has provided new opportunities to collect and analyze large quantities of student learning data. Within an SPE, three broad categories of data can be collected: *time management*, *programming*, and *social data*. Table 3.1 identifies behaviors in each of these categories that can be gleaned from the data collected by a social programming environment such as OSBIDE (discussed earlier in this chapter). An important limitation of this taxonomy is that it includes only observable behaviors within a given system and does not account for behaviors outside of that system, e.g., questions asked via external email or in person.

[6]The taxonomies presented in this section were inspired by discussions that took place at a workshop entitled "Leveraging Programming and Social Analytics to Improve Computing Education," which was held in August 2015 in Omaha, Nebraska in conjunction with ICER 2015. The workshop was sponsored by the same National Science Foundation award that has funded my Ph.D. research.

www.manaraa.com

Table 3.1. Taxonomy of behaviors

TIME MANAGEMENT	
A	Time engaged with the IDE[7]
B	Procrastination (When first programming event generated)
C	Work-time distribution (time spent by day – look at pattern)
PROGRAMMING BEHAVIOR	
	State-based
D	NPSM (compilation and execution behavior – problem-agnostic)[8]
E	Stanford (machine learning over code submissions – problem specific)[9]
F	Error-Quotient &Watwin Score (predictive algorithm based on time in error state[10]
	Time/Count-based
	Student Coding Behaviors
G	Amount of code changed between builds
H	Number of executions between builds (testing a wide range of inputs)
I	Execution in debug mode vs. not in debug mode
	Spread of good ideas (large pastes detected, source citing, change in pasted code)
J	*Unit testing results*
K	Student invoked (students get results)
	Automatic (on every build, results not shared with student)
L	*Program content*
M	Presence of keywords or constructs (compared against desired keywords and constructs in programming assignment prompt)
N	Closeness of match with canonical solution
O	Analysis of comments and style
SOCIAL BEHAVIOR	
P	Post content (content correlated with performance)
Q	Post frequency (measure of participation; participation level)
R	Badges or reputation (gamification / social rewarding mechanisms)
S	Questions asked
T	Questions answered
U	Answers marked helpful
V	Activity feed viewing habits
W	Sense of community

[7] Defined as non-idle time interacting with the IDE. Non-idle time is calculated as the difference between start/stop periods of activity in the IDE. Start time is determined by the first activity timestamp since the last idle period and end time is determined by next idle period (no IDE activity) greater than 3 minutes.

[8] (Carter et al., 2015)

[9] (Huang, Piech, Nguyen, & Guibas, 2013)

[10] (Jadud, 2006; Watson, Li, & Godwin, 2013, 2014)

www.manaraa.com

The first category, *time management*, includes the amount of time students are engaged with the IDE or problem-solving environment, procrastination (i.e., when do students first start programming relative to the assignment due date?), and student work-time distribution (i.e., when do students spend time programming relative to an assignment deadline). The time-management category is important because we know from past research [e.g. (Chickering & Gamson, 1987)] that time-on-task traditionally plays a pivotal role in the student learning and success.

The second category, *programming behavior*, includes state-based, time- based, and count-based behaviors related to computer programming activities. State-based programming behaviors have been shown to provide useful information about computer science students. As demonstrated by the research reviewed in section 2.3, the various programming states in which students spend time can be indicative of their eventual success in computer science courses (e.g. Carter et al., 2015; Jadud, 2006; Watson et al., 2014). Likewise, time- and count-based behaviors can also be indicative of success. We can break these up into three subcategories: *coding behaviors*, *unit testing results*, and *program content*. We can observe and count things like the amount of code that changes between builds, the number of executions, the number of errors, the number of unit tests passed or failed (which can be automatically determined using computer tools such as WebCAT (Edwards & Perez-Quinones, 2008), the presence of keywords or constructs, or even the degree to which a programming solution matches a canonical solution (as in, e.g., Hundhausen, Brown, Farley, & Skarpas, 2006).

Finally, within an SPE, we can observe some aspects of students' social behavior. This is made possible, as mentioned in section 1.1.2, by the presence of an integrated activity stream, through which students communicate asynchronously. Behaviors that can be observed include *activity feed post content and frequency* (as, e.g., analyzed in Carter, 2016), *reputation* (badges, achievements,

www.manaraa.com

helpful marks, or other gamifying methods; see, e.g., Bayliss, 2009; Burguillo, 2010), *questions asked and answered* (as, e.g., analyzed in Hundhausen & Carter, 2014), and *activity feed viewing habits* (posts read and marked helpful).

3.2 GAMS Conceptual Model

Based on the learning theories underlying this dissertation, I have developed *GAMS* (Goals, Actions, Motivation, Standing) as a conceptual framework for generating SPE interventions based on a stream of continuously-collected SPE data. To generate interventions, GAMS focuses on four central purposes that SPE interventions need to fulfill to effect desired changes in students' learning processes. These purposes can be phrased as questions that effective SPE interventions should help the learner answer:

1. Goals: What are my goals?
2. Actions: What actions can I take to move closer to my goals?
3. Motivation: What motivates me to take action?
4. Standing: Where do I stand relative to my goals and others in the community?

Questions 1 and 2 are drawn from Rotter's (1966) *locus of control* theory. Interventions that respond to these questions aim to (a) help learners to define *virtuous* goals that are likely to lead to learning success; (b) guide learners to actions they can take in order to meet those goals. Interventions that respond to these two questions, according to Rotter's theory, will help learners to *internalize* their actions by showing them that they are in control of their learning outcomes.

With respect to questions 1 and 2, a key question arises: Within an SPE, what are *educationally virtuous* goals that computer science students should pursue? Drawing on the general education and computer science education literature, GAMS is rooted in a set of *best learning practices* for

www.manaraa.com

computer science students relative to the SPE data available on student learning processes. Table 3.2 codifies these best practices as a set of learning goals relative to the observable behaviors within an SPE. For each learning goal, the table includes its rationale, rooted in supporting literature, along with an example prompt that might be used to get the learner to take action toward that goal.

Question 3 ("What motivates me to take action") addresses a key determinant of learning and behavior change: *motivation*. Ideally, a learning environment will inspire *intrinsic* motivation for learning (Malone, 1981). However, the use of extrinsic motivators within social learning environments, such as badges, reputation points, and achievements, are commonly used to provide a form of *extrinsic* motivation for participation (Antin & Churchill, 2011). Gamification theory for online social communities (Antin & Churchill, 2011; Nepal, Paris, & Bista, 2015) identifies the need for a set of gamification interventions. This theory holds that community members will be more motivated to take actions within an online community if their actions are codified in a tangible and *relevant* (Prause & Jarke, 2015) system of badges or achievements that are visible within a community. In addition to providing motivation to act, such a system of badges can help community members to establish trust, common ground, and shared values (Schmidt, Geith, Håklev, & Thierstein, 2009).

Gamification interventions might motivate desired behaviors by assigning "points" or "reputation" to the desired behaviors. Gamification methods have also been shown to increase social interaction and sense of community (Deterding, Dixon, Khaled, & Nacke, 2011; Jakobsson, 2011), and can bring a sense of competition and motivation for learners (Burguillo, 2010; Issa, Hussain, & Al-Bahadili, 2014). Thus, in the social communities fostered by SPEs, gamification interventions could prove to be particularly appropriate and useful.

www.manaraa.com

Interventions that respond to question 4 ("Where do I stand in the community?") are rooted in Bandura's (1997) and Astin's (1999) social learning theories, which identify factors and conditions that promote learning within a community. On this view, the SPE is seen as the *community hub* in which students participate and learn. The purpose of interventions that respond to question 4, is to enable students to observe others' learning behaviors within the community (i.e., *vicarious experiences*), and to assess their own learning progress relative to others in the community (i.e., *enactive experiences*). The social learning theories posit that, by having these kinds of experiences, students will obtain a more accurate view of their own abilities (*self- efficacy*), in turn developing a higher capacity to succeed and persist in the class and, ultimately, in the discipline.

Finally, all four questions fit into Kolb's experiential learning theory (Kolb & Kolb, 2005; Kolb, 1984), which holds that the reflective observation of concrete experiences leads to learning. On this view, the four questions of the GAMS framework are educationally valuable in that they steer learners toward interaction and reflection with peers and mentors. Similarly, to situate learners within Vygotsky et al.'s (1978) zone of proximal development, the GAMS framework relies on software scaffolding (Wood et al., 1976) to steer learners toward reflecting on their learning experiences with their peers.

www.manaraa.com

Table 3.2. Possible behavior goals and suggested correction actions

Behavior[11] Category	Goal	Prompted Action	Empirical Rational
A	Spend **X** more minutes coding in-IDE working on [***assignment***] if not finished	Spend an additional **YY** minutes coding in-IDE today	Time on task [12a]
B	Create [***assignment***] project solution on the day it's assigned	Create [***assignment***] project solution today	Time started[3b]
C	Spend at least **XX** minutes working for each coding session until done	Add **YY** minutes to your coding sessions	Time on task[3a]
D	End a coding session without any compile or execution errors	Reduce the number of compile/execution errors by at least **Y**	Time in error state[3c]
F	Make modifications to the code between unsuccessful builds	Make at least **Y** additional change to the code after your next unsuccessful build	Time in error state[3c]
G	Test at least **X** input values on successful builds	Test an additional input value on your next successful build	Test frequency[3d]
H	Code execution at least **XX%** of runs in debug mode	Run in debug mode your next **Y** builds	Programming state[3c]
I	Cite all pasted code sources	Check pasted code for citation where needed	Plagiarism prevention[3i]
J	**100%** unit test pass rate	Fix at least **Y** more failed unit tests	Time in error state[3c]
L	All expected keywords/constructs used for [***assignment***]	Use keywords/constructs **Y/Z** from [***assignment***] prompt	Construct use[3e]
M	Assignment code at least **XX%** match to canonical solution	Consider revising usage mismatch **Y/Z** from [***assignment***] prompt	Construct use[3e]
N	Comment styling matches style guidelines	Consider revising comment lines **Y/Z** to match style guidelines	Style consistancy[3f]
O	**X** posts made with [***#hashtag***] topic	Create **Y** additional posts for the [***#hashtag***] topic	Social interaction and community involvement[3g]
P	At least **X** posts per week made	Make at least **Y** more posts per week	
Q	Gain at least **X** badges before [***assignment***] deadline	Complete **Y** to gain at least **Z** more badges before [***assignment***] deadline	Gamification[3h]
R	Ask at least **X** questions per week	Ask at least **Y** more questions per week	Social interaction and community involvement[3g]
S	Answer at least **X** questions per week	Answer at least **Y** more questions per week	
T	Mark at least **1** answer helpful before [***assignment***] deadline	Mark at least **Y** more answers helpful before [***assignment***] deadline	
U	View activity feed at least **1** time per day	View activity feed at least **Y** more times per day	

[11] Behaviors from Table 3.1. Taxonomy of behaviors

[12] [a](Chickering & Gamson, 1987) [3b](Edwards et al., 2009; Martin, Edwards, & Shaffer, 2015) [3c](Carter et al., 2015) [3d](Buffardi & Edwards, 2013) [3e](Hundhausen, Brown, Farley, & Skarpas, 2006) [3f] (Oman & Cook, 1990) [3g](Astin, 1999; Bandura, 1997) [3h](Burguillo, 2010; Deterding, Dixon, Khaled, & Nacke, 2011) [3i](Mann & Frew, 2006)

www.manaraa.com

3.3 Design Dimensions of SPE Interventions

While the GAMS framework suggests four broad classes of effective SPE interventions that follow from the need to identify and share goals, actions, motivation, and standing, GAMS does not provide specific guidance on the actual design of the interventions. Three design dimensions are relevant here: the content of the interventions, the way they are delivered to the learner (delivery mechanism), and when they are delivered to the user (timing).

3.3.1 Content

The content of an intervention will be dictated by its purpose. Following sound principles of human-computer interaction design, an intervention should share only the information salient to its purpose, and nothing more (Johnson, 2010).

3.3.2 Delivery Mechanism

Three distinct mechanisms might be used to deliver interventions to learners within an SPE: *visualizations*, *notifications*, *and constraints*. Theories of human perception and human-computer interaction can be used to match an intervention to an appropriate delivery mechanism based on the intervention's role or purpose. Below, I elaborate on each delivery mechanism, and present theoretical considerations for matching it to interventions.

3.3.2.1 Visualizations

The *visualization* delivery mechanism aligns with the learning dashboards previously discussed in section 2.6. Within the context of SPEs, interactive visualizations (e.g. graphs and charts) can intervene in the learning process primarily by enabling learners to visually explore how they are doing in relation to their own goals, and their community. By providing a visual indication of their progress, goal- or action-oriented visualizations can help learners see how far they have

www.manaraa.com

come, how far they need to go to get their goals, and what specific actions might help get them to their goals—all important to facilitating an internal locus of control (Rotter, 1966). Community-oriented visualizations, in contrast, can provide learners with a basis for assessing their own activities and progress relative to others in the community, and hence can be used to facilitate *vicarious experiences* (Bandura, 1997).

An important consideration for visualizations is that they are most effective in promoting learning if they *actively engage* the learner (Hundhausen, Douglas, & Stasko, 2002). This implies the need to provide some level of *interactivity* in the visualizations that are used as interventions by, for example, allowing learners to perform what-if analyses. Basic principles of human perception and human-computer interaction must also be taken into consideration when designing visualizations to be used as interventions. For example, visualizations should be designed so as present information relevant to learners' immediate tasks, with the relevant information easily recognizable and within the learner's foveal view (Johnson, 2010; Norman, 2013).

3.3.2.2 Notifications

The *notification* delivery mechanism differs from the *visualization* delivery mechanism in two key ways. First, it is a *textual* message, as opposed to a graphical visualization. Second, whereas a visualization intervention is presented to the learner in a non-obtrusive way (the learner must potentially seek it out by navigating to it and clicking on it within an SPE), a notification may be (but is not necessarily) delivered *obtrusively*, in the form of a *modal* pop-up message that interrupts the learner within the SPE. In an SPE, non-obtrusive notifications that, e.g., relate the learner's activities and interests to those of other learners, may be injected into the activity stream. In

www.manaraa.com

contrast, obtrusive notifications may pop up within an SPE at points at which the SPE notices that the learner is engaging in either undesirable or desirable actions.

As was the case for visualizations, notifications must be designed with consideration of principles of human perception and human-computer interaction. For example, there is a definite cost to interrupting a learner engaged in completing a task (see, e.g., Okoshi et al., 2015). That cost must be carefully weighed against the potential for the notification to positively change the trajectory of the learner's work (Johnson, 2010). In addition, for obtrusive notifications to be most effective, they should be delivered as close as possible to the learning behavior they intend to influence (Aljohani & Davis, 2013).

in the case of non-obtrusive notifications, learners will most likely benefit if the notifications present immediately-actionable guidance that is directly relevant to the learner's immediate goals (Johnson, 2010). In the case of community-based notifications, it will be beneficial to make community activities more relevant to the leaner by explicitly identifying the learner's similarities and differences with respect to others in the community.

3.3.2.3 Constraints

In user interface design, constraints are a well-known means of preventing illegal or undesirable user interface actions, thereby guiding the user toward legal or desirable user interface actions (Norman, 2013). Constraints typically manifest themselves as user interface controls that limit the actions that can be taken—for example, menu items that contain only allowable actions, or selection sliders that can move over only a legal range. In the context of SPE interventions, constraints could be applied to the IDE interface to prevent the learner from taking actions that are incompatible with what are desirable trajectories, thus guiding the learner toward desirable

www.manaraa.com

trajectories. For instance, if the learner has not compiled her program in a long time, the SPE could disable editing (a constraint), and display a message (a notification) informing the user that compilation might be a good idea. Note that constraints can be seen as an *extreme* form of intervention; they prevent the user from performing further actions. Therefore, they should be used with care—only in situations that warrant them, such as when other, less extreme forms of interventions have been tried repeatedly and failed.

3.3.3 Timing

This dimension relates to when a given intervention should be presented to the user. One option is for an intervention to be *persistent*. If an intervention is continuously available in the SPE, it could be part of a tab or area within the SPE to which users navigate when they want to access the intervention. An example of this kind of persistent intervention would be a learner dashboard (visualization delivery mechanism) that shows the learner's current goals and progress. A second option is a *triggered* intervention: an intervention that is dynamically delivered to the learner in response to the immediate actions the learner is taking or a state to which the learner has just transitioned. For example, the NPSM model described earlier (Carter et al., 2015) characterizes programming behavior based on the current semantic and syntactic correctness of the program being edited. If a learner remains for long periods of time in a syntactically unknown state (i.e., the learner has not compiled the program), such behavior negatively correlates with programming success. Hence, an intervention that reminds a learner to compile his or her program would be an example of a triggered intervention. It would be delivered when the SPE detects a long period of editing in the absence of any attempt to compile a program.

www.manaraa.com

3.4 Summary

Resonant with my goal of developing educationally effective interventions within an SPE, this chapter has presented a model for transforming continuously-collected data in an SPE to interventions that can effect positive changes. At the center of this model is GAMS, a conceptual framework that uses learning theory to identify four important classes of interventions—those responding to learners' goals, actions, motivation, and standing. To constrain the space of intervention designs, this chapter has also identified the set of behaviors observable within an SPE, together with the key design dimensions of interventions delivered within an SPE. Having laid the theoretical groundwork for the interventions explored in this dissertation, I turn in the next chapter to the empirically-driven design of those interventions.

www.manaraa.com

CHAPTER 4

DESIGN OF INTERVENTIONS TO PROMOTE SOCIAL INTERACTION

In this chapter, I synthesize the findings of the previous two chapters by presenting the design evolution of the OSBLE+ social and programming interventions studied in this dissertation. The chapter concludes with a refined set of research questions regarding how the interventions can influence students' programming and social behavior.

4.1 Technological Foundation for the Interventions: OSBLE+

In prior work spanning the past ten years, my lab has developed OSBLE (Online Studio-Based Learning Environment) (HELP Lab, 2012), a learning management system (LMS) rooted in the Studio-Based Learning model (Hundhausen, Agarwal, & Trevisan, 2011; Hundhausen et al., 2013; Hundhausen, Agrawal, Fairbrother, & Trevisan, 2009); see Figure 4.1. Over the past five years, we have also developed the previously mentioned OSBIDE, a *social* programming environment (Carter & Hundhausen, 2015; Carter et al., 2015) that integrates an activity stream into an IDE (see Figure 1.1, p. 6).

4.1.1 OSBLE+ System Overview

Since then, I have taken a leading role in developing OSBLE+ (see Figure 4.2-Figure 4.4), which combines OSBLE and OSBIDE into a single integrated system and includes a new learning analytics environment (see Figure 4.3) capable of presenting, in visual form, the social and programming data collected through the OSBIDE plug-in. This is how the name OSBLE+ was derived: OSBLE *plus* learning analytics. In addition to merging the data collection capabilities

www.manaraa.com

from OSBIDE into OSBLE, I developed a new activity feed which facilitates real-time communications (i.e. messages are pushed to all clients as they occur) and integrated it into both the OSBLE dashboard and the Visual Studio plugin (highlighted in Figure 4.2 and Figure 4.4). Thus, I created a tightly-coupled communication system between the learning dashboard and the programming environment.

Figure 4.1. Legacy OSBLE Dashboard

www.manaraa.com

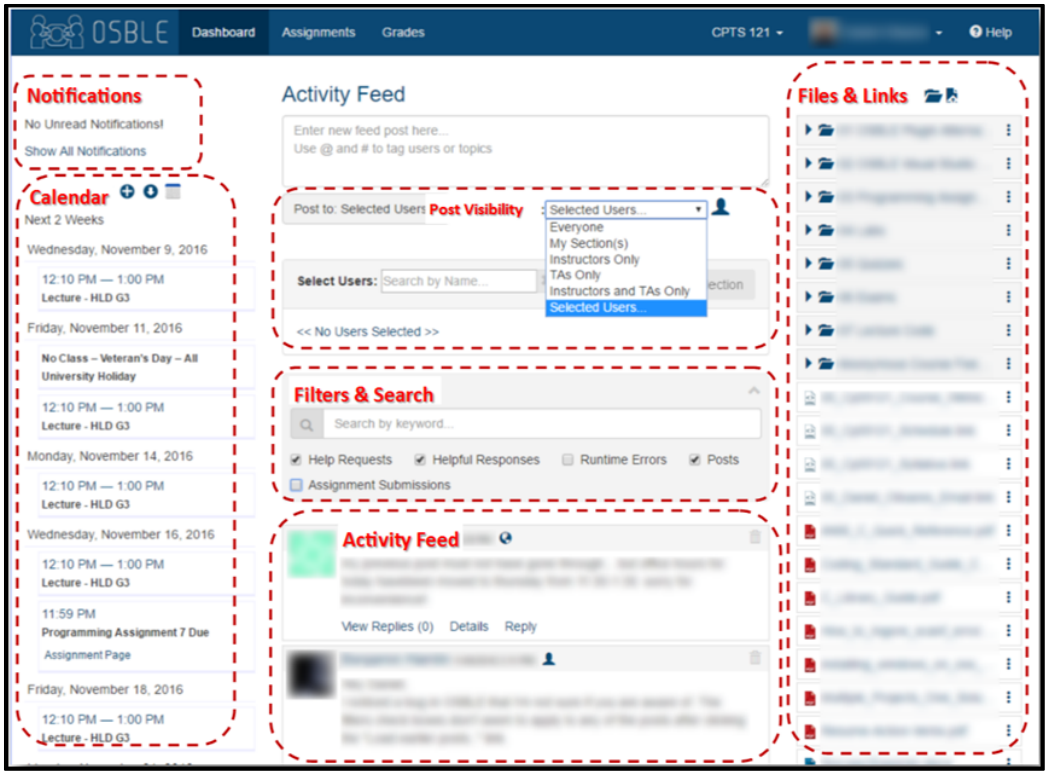

Figure 4.2. The OSBLE+ course dashboard

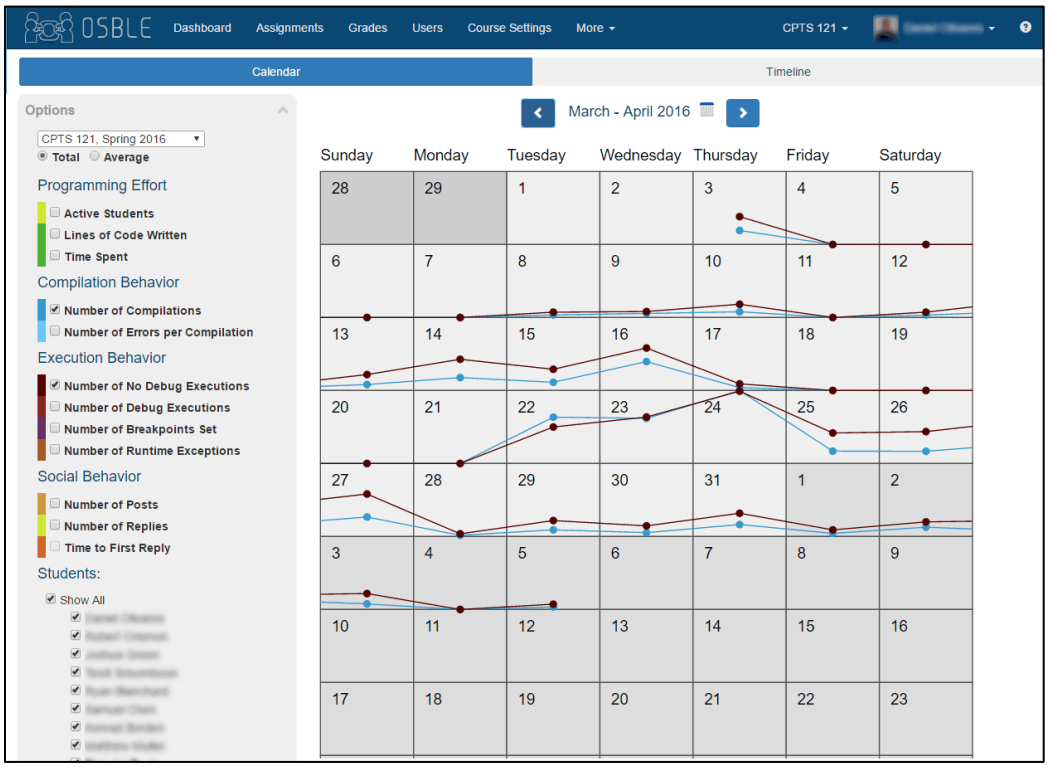

Figure 4.3. The OSBLE+ analytics dashboard. Shows LMS activity metrics (left) displayed visually on a calendar timeline (right)

www.manaraa.com

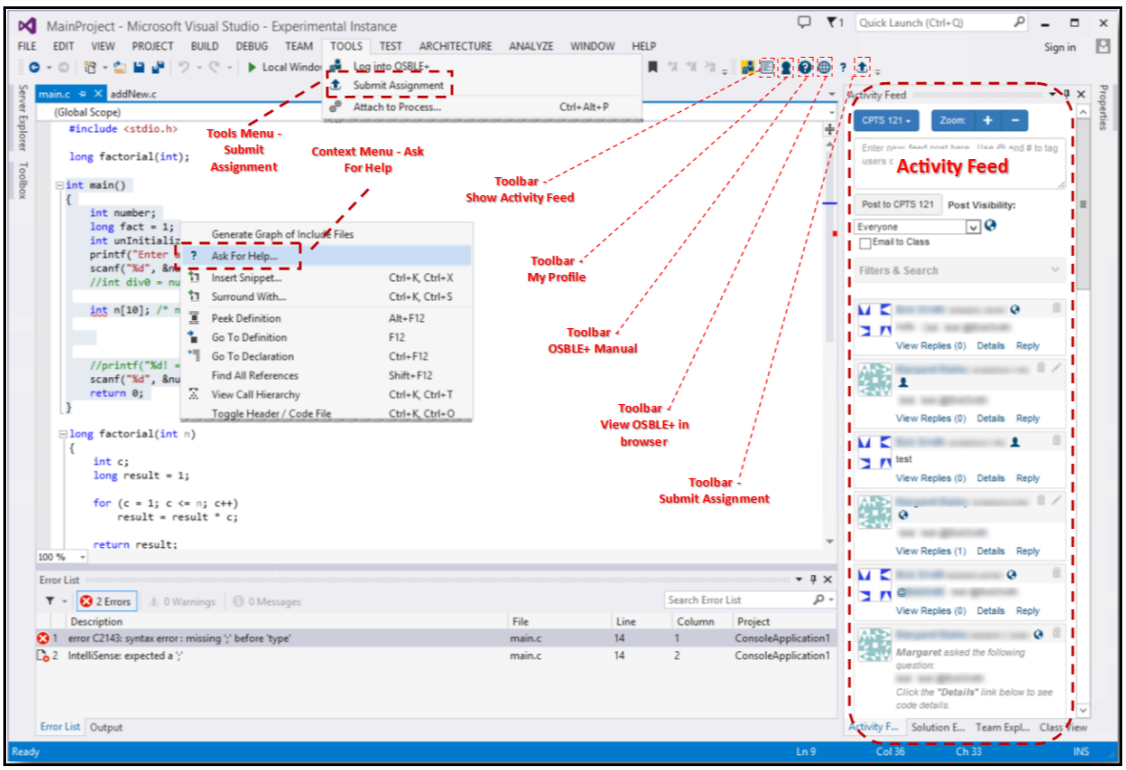

Figure 4.4. OSBLE+ Visual Studio integration overview

All design features present in OSBLE were retained in OSBLE+—most prominently, the traditional LMS functions such as the file system for disseminating course materials, the system for posting and submitting course assignments, and the grade book features for uploading or viewing grades. OSBLE+ also integrated several additional features intended to encourage communication between users of the system. These changes include the real-time communication provided by the new activity feed, an expansion of the posting visibility options to include selected user groups and anonymous posting, the addition of keyword and category filters based on new post sources (e.g. help requests and other activities from the Visual Studio plugin), and an adaptive user interface capable of scaling gracefully to any device, including mobile devices. Notice that many of these changes resonate with learning theories discussed in previous sections by. facilitating student involvement and awareness of community.

www.manaraa.com

The use of OSBLE+ as my data collection tool provided a means to incrementally assess different intervention designs within the context of hybrid computing courses with an online component supported by OSBLE+. I now present the highlights of an iterative, user-centered design process rooted in a series of formative early design studies (Gould & Lewis, 1985; Nielsen, 1993) that systematically explored the design space of interventions. This design process included both an instructor survey to gauge methods of monitoring and addressing student progress (responding to the RQ "What information is needed?"), and a student study used to gain user feedback on initial intervention design prototypes (responding to the RQ "Which interventions should we use?").

4.1.2 Individual Contribution to the OSBLE+ System

My individual contribution to the OSBLE+ system involved taking a leading role in the merging and development of the two foundational code bases: (1) the OSBIDE Visual Studio extension project; coded in C# on the .NET platform using the Visual Studio SDK, and (2) the OSBLE learning management system; created on the ASP.NET MVC2 platform alongside numerous web-based scripting languages and frameworks. Additionally, I made back-end database (MS SQL) changes to accommodate the merging of the two systems and adding of new system requirements.

In the final merged version of OSBLE+, I developed each of the new intervention interfaces (described in the following sections) using a combination of .NET MVC3 C#, HTML, CSS, JavaScript, and numerous other web-based frameworks. I also developed the required back-end functionality for intervention generation and triggering using the OSBLE+ database to store interventions. In total, my development contributions resulted in over 200 commits to the HELP

www.manaraa.com

Lab source control, over one million combined additions and deletions, and more than 30,000 lines of code in the final OSBLE+ code-base during over four years of steady development.

4.2 Early Design Studies[13]

The early design of the interventions was informed by a series of studies which took place in the summer and fall 2016 semesters. Participants included both computer science instructors from throughout the U.S., and students enrolled in CptS121 and 122 courses[14]. Data collected in these early design studies were used to (a) improve the user experience of the intervention interface, (b) improve the intervention delivery methods, (c) refine the information shown to users (based on foundational literature and perceived usefulness), and (d) improve the functionality and usability of the interface and data collection tools (i.e., eliminate software bugs and usability problems).

4.2.1 Instructor Survey

A survey was created in order to learn more about how instructors monitor and assess student progress and thus, to provide initial input into the first research question of Chapter 1 ("*What information is needed to assess learning and tailor instruction?*"). The survey consisted of demographic and instructional background information, five open ended questions, and 17 Likert scale (10 point) questions. The survey was designed to gauge perceived importance of different types of student programming data, level of confidence in monitoring student programming activity, and likelihood of using an instructor dashboard to monitor student programming

[13] See Appendices A, D, F, and G for more detailed reports of these design studies.

[14] CptS 121 is the Washington State University introductory programming course for computer science majors. CptS 122 is the follow up course focusing on data structures.

www.manaraa.com

activities. Participants for the survey were recruited via the ACM SIGCSE mailing list. 73 computer science instructors completed the survey.

4.2.1.1 Summary of Results

Many of the instructor responses aligned with information that can be gathered within an SPE, learning theory, and related literature. Most prominently, instructors' responses significantly overlapped with established best practices (Chickering & Gamson, 1987)—e.g., contact between students and faculty, cooperation between students, prompt feedback, and time on task (Table 4.1).

Similarly, the data instructors considered important to the learning process had much in common with metrics that can be gleaned from the SPE (Table 4.2). These results also align with data considered by predictive models such as those developed by Carter et al. (2015), Jadud (2006), and Watson et al. (2014).

Table 4.1. Instructor survey: open-ended survey response summary

	Count	Percent*
Desired Metrics		
Coding Process	68	42%
Learning Process	39	24%
Design	27	17%
Errors	15	9%
Social	12	8%
Current Practices		
Meetings	77	46%
In-Class intervention	32	19%
Additional guidance	20	12%
Online interventions	18	11%
Peer interaction	13	8%
Design structuring	5	3%
Feedback	4	2%

	Count	Percent*
Desired Additional Design Metrics		
Coding Metrics	23	45%
Understanding	17	33%
Social Metrics	11	22%
Suggested Intervention Methods		
Additional intervention	59	64%
Check Code	26	28%
Share Data	5	5%
Preparation	2	2%
Desired Instructor Dashboard Features		
Class/Student Statistics	30	67%
Hardware/Software	10	22%
Ease of Use	3	7%
Preparation/Resources	2	4%

*percent of combined categories

www.manaraa.com

Table 4.2. Instructor survey: programming and social metrics summary

Question*	*M***	*SD*	Mode	Percent Rated 6+
Programming metrics				
Programming constructs used	6.64	2.27	9	70
Calendar view of student activity	6.78	2.1	8	74
Runtime errors	6.47	2.08	8	71
Build errors	6.16	2.26	8	66
Time on task	7.68	1.58	7	90
Debugger use	5.49	2.59	6	55
Methods written	5.36	2.15	5	51
Comments present in code	5.25	2.17	5	44
Lines of code written	5.14	2	5	44
Social metrics†				
Answers given marked helpful	5.41	2.54	8	52
Programming answers received	5.11	2.53	8	44
Programming questions asked	5.48	2.37	5	48
Answers marked helpful	4.78	2.53	5	37
Answers given	5.36	2.3	4	49
Instructor				
Instructor likelihood of using a learning dashboard	7.14	2.45	7	82
Instructor confidence in being aware of student programming activity	6.12	2	7	66

**N=73, sorted by Mode, **In an online discussion forum linked to the class*

†A 10-point Likert scale was used, with 1 being "not important at all" and 10 being "extremely important"

We also identified some notable discrepancies between instructor responses and the GAMS model's interest in social behaviors. Responses tended to be neutral for the social metrics in the Likert type questions. However, when inspecting open-ended responses, we see a considerable emphasis on processes which involved interaction with peers, instructors, and community, as emphasized by social learning theories (Astin, 1999; Bandura, 1990; Kolb & Kolb, 2005; Rotter, 1966).

The findings provide additional support for making use of student programming and social data in the design of SPE interventions, motivating many design decisions in both the initial prototypes and the final intervention design. They also support the idea that, by aligning with

www.manaraa.com

current instructor practices, properly designed interventions may be more likely to support instructor adoption.

An important limitation of these results is that this survey gauged instructor attitudes and opinions failing to provide evidence of what might work in practice. Nonetheless, such opinions are important to consider: Instructors will be less inclined to make a change or use a new tool if it doesn't fit into their local department, is too complex of an investment, or they don't feel it provides anything of value towards their goal of helping students (Ni, 2009; Ni, McKlin, & Guzdial, 2010). Understanding instructor beliefs surrounding these metrics can therefore help to increase their adoption of the technological interventions explored in this dissertation.

4.2.2 Student Survey and Prototyping

Building on the results of the instructor survey, I developed a series of prototype interventions for integration within the OSBLE+ system (see Appendix D). A survey was administered during the spring 2016 semester to gather impressions on the early prototype designs from the target user base (introductory computer science students) and to prompt for additional design feedback for the next design iterations. Ten students from the spring CptS 121 and CptS 122 courses participated: four from CptS 121 and six from CptS 122.

Five prototype designs were presented in this study: "Goals," "Community Standing," "Goals at a Glance," "Who's Online," and "Recent Activity" (see Appendix D). Participants were walked through a series of questions that contained images of the individual intervention panels. Participants were then prompted to click a feature they would most likely use and to describe (1) how they expected the interface to respond, and (2) what they believed that feature would do. Additionally, they were asked to rate their perceived usefulness of the intervention (1, not at all

www.manaraa.com

useful to 7, very useful). Click interactions were tracked for each intervention along with their responses. Participants were also asked to rank the features in order of most and least important to them and to rate the likelihood (1, not likely at all to 7, very likely) and frequency (never to 3+ times per day) with which they expected they would use the intervention.

4.2.2.1 Summary

Analysis of survey results suggest a moderately positive reception of these interventions and their use (Table 4.3). Overall, most responses on average were neutral to positive (average responses 4-5 out of 7). The only exceptions were the frequency of use responses for the "Community Standing" and "Goals at a Glance" interventions. This was to be expected, however, as frequency of use uses a time-scale instead of a negative to positive rating. Positive feedback provided support for use of the intervention prototypes; open-ended survey responses indicated that participants thought the ability to "*connect through the IDE would greatly benefit the class*" because it could "*help you over your hump in a part of a program that may be just a simple problem.*"

Click interactions and participant interpretation of the interface design and resultant usage mostly aligned with the design goals. For each of the intervention interactions, participant intuitions were in-line with the design model and there were no disconnects between click actions and the participant expectations of system responses. For example, participants clicking on "mail" icons or name links expected to initiate communication with the corresponding user. These interactions were used to create heatmaps for each design (see Figure 4.5). One exception to the expected interpretations is the "Goals" intervention was that some participants clicked on the "OK" or other information labels with the expectation that this action would either dismiss the goal or

www.manaraa.com

mark it as completed, as opposed to the design intention of it being just a status label and not an interaction object. The results discussed here were used to inform design decisions for the following intervention studies.

Table 4.3. OSBLE+ Community prototype feedback responses

	M	*Mdn*	Mode
(Goals)			
Usefulness	5.1	6	6
Likelihood of Use	4.8	5	5
Frequency of Use	4.2	4.5	5
Integrated IDE	5.5	6	7[a]
Rating Rank	1st (sum rating 30)*		
(Community Standing)			
Usefulness	4.6	4.5	6[b]
Likelihood of Use	4.2	4.5	6[c]
Frequency of Use	3.7	3.5	3
Integrated IDE	4.2	4	3
Rating Rank	4th (sum rating 51)		
(Goals at a Glance)			
Usefulness	4.7	5	5
Likelihood of Use	4.7	4.5	6
Frequency of Use	3.8	4	4[d]
Integrated IDE	4.8	4.5	4
Rating Rank	3rd (sum rating 48)		

	M	*Mdn*	Mode
(Who's Online)			
Usefulness	5	5	5[e]
Likelihood of Use	5.4	6	7
Frequency of Use	4.3	4.5	5[f]
Integrated IDE	5.2	5.5	7
Rating Rank	1st (sum rating 30)*		
(Recent Activity)			
Usefulness	5.6	5.5	7[g]
Likelihood of Use	5.7	6	7
Frequency of Use	4.8	5	5[f]
Integrated IDE	5.3	5	5
Rating Rank	2nd (sum rating 39)		

*1st rank tie, lower sum rank indicates higher ranked usefulness

multiple mode responses:
[a]6:7 [b]4:6 [c]3:6 [d]3:4 [e]4:5 [f]5:7 [g]4:5

Figure 4.5. "Goals" click interaction heatmap indicating number and location of clicks on the design

www.manaraa.com

4.3 Design Iteration 1

Originally, the goal of this design iteration was to refine the design of the intervention prototypes (and then field test the changes concurrently with the summer, 2016 offering of CptS 121. Feedback provided on the spring 2016 prototypes over the course of a two day, NSF-funded, learning analytics workshop[15] motivated a change in direction: I decided to narrow the focus of the designed interventions from the broader instructor and student-based community dashboard to student-only intervention prompts designed to stimulate help-seeking, help-giving, and social interaction. The OSBLE+ system was designed such that the user data collected via the IDE and the LMS could both be fed into the intervention generation system (see Figure 4.6). With this structure, it was possible to focus solely on the student intervention aspect of the system without further redesigning the OSBLE+ system.

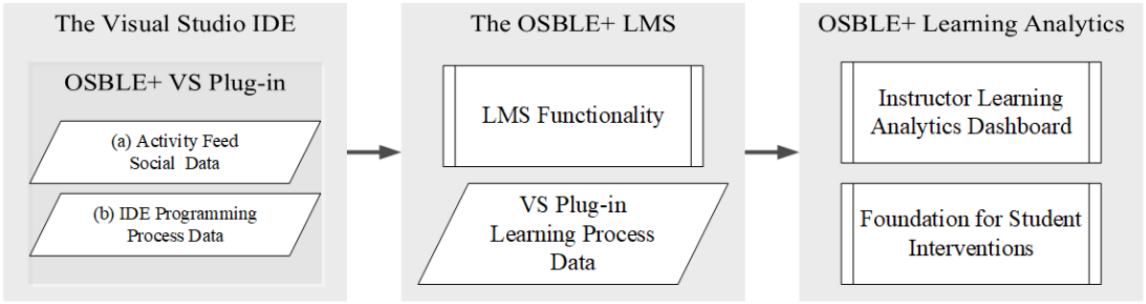

Figure 4.6. An overview of the OSBLE+ software architecture

Though the first round of intervention prototypes were not used for the remainder of this dissertation, there were certain key elements that carried over into the next iteration, including the social interaction, representation of user status, and representation of user goals.

[15] "Leveraging Programming and Social Analytics to Improve Computing Education," year 2 project workshop which was held in June 2016 in Spokane, WA. The workshop was sponsored by the same National Science Foundation award that has funded my Ph.D. research.

www.manaraa.com

The new intervention designs were built around the same motivating learning theories (see section 2.1) as the first set of prototypes. Two initial prototype designs were tested: help-seeking and help-giving (see Appendix F). The help-seeking intervention prototypes used a goal and action model to encourage users to focus their goals and actions to resolve issues (Rotter, 1966) along with scaffolded prompts and questions (Wood et al., 1976) to stimulate interaction with more knowledgeable peers (Vygotsky, 1978) as well as involvement with the community (Astin, 1999) and initiate vicarious and inactive experiences (Bandura, 1997). Additionally, each of these scaffolded question prompts was designed to allow users to reflect on their learning process (Kolb, 1984).

4.3.1 Evaluation Study

This study was designed to elicit initial design feedback, gauge student opinion of the designs, and elicit their help-seeking, help-giving, and social behaviors while programming Five participants, who had previously taken the CptS 121 offering at WSU, were recruited to answer a brief survey that included questions related to their help-seeking and help-giving behavior during the programming process (see Appendix F).

Participants were given a series of simple programming scenarios followed by hypothetical scenarios, e.g. “You have been working on coding this program for a while and are having trouble getting your program to build successfully. Assume you don't immediately understand the cause of the error.” Each scenario was followed by questions asking participants to rate their help-seeking frequency and likelihood of seeking help from instructors, teaching assistants, and classmates as well as their approach to solving the problem provided for the scenario using a 7-point Likert scale, with one representing “never” or “not likely at all” and seven representing

www.manaraa.com

"always" or "very likely." Following the above help-giving, help-seeking, and social interaction questions, participants were asked to revisit the same scenarios but were this time also shown early prototype designs of the newly refocused interventions. For each of the scenarios, participants were asked to describe what the prototype window's purpose was and to rate their perceived usefulness and frequency of use. Additionally, participants were asked to provide design feedback and to rate the likelihood of the prototype helping them resolve the hypothetical issues.

4.3.2 Summary of Results

Overall, participants showed a mixed reception to the presented interventions, and some reluctance to use the system. Across the three main prototype design scenarios, we saw an aggregate mean response of 3.8 (out of 7) for seeking help from their instructor, 4.5 for teaching assistants, and 2.7 for classmates. Participants indicated they were most likely to ask for help from their teaching assistants (M=4-4.8) over their instructor (M=3.4-4.2) and least likely to ask for help from their classmates (M=2.4-3.2). In contrast, open ended responses indicated that they were willing to help their classmates but either felt like the help wasn't needed or they were not confident in their ability to help.

Participant feedback on each of the individual intervention windows indicated that the ability to post their questions anonymously would be desirable. Participants also expressed concerns regarding the immediacy of help being available, i.e., people working at different times might mean that students would not get the help they needed in a timely manner. Further feedback suggested the need for finer grained changes to prompts, e.g., vague or broad prompts such as "You're doing great!" (*how* are they doing great?).

www.manaraa.com

4.4 Design Iteration 2

4.4.1 Redesign

Based on prototyping and design sessions, workshop feedback, and the summer 2016 prototyping study, the prototype intervention designs were further refined into a new set of interventions with a focus on positively affecting three categories of learner behavior: *help-seeking*, *help-giving*, and *social interaction* (see Table 4.4). Each of the three desired behavior categories was broken up into a total of eight individual interventions; three help-seeking, two help-giving, and three social interaction. For each intervention, Table 4.4 describes what triggers the intervention and provides a theoretical motivation for the intervention.

In early prototyping, the most relevant learning theories considered were works from Astin, Bandura, and Rotter. In this redesign, the focus shifted away from some aspects in the original GAMS model—most notably, *community standing* (situating the user within the community). However, the goals, actions and, to a lesser extent, the motivation aspects of GAMS remained relevant to the intervention designs. At the same time, Kolb's experiential learning theory, the Zone of Proximal Development and scaffolding became more important to the design, with the inclusion of focused prompts for question asking, answering, scaffolding of responses, and encouraging reflection in the learning process.

www.manaraa.com

Table 4.4. Intervention categories, triggers and motivation

Category	Intervention	Trigger	Motivation
Help Seeking	*Runtime Errors: Get Help!*	1. ***COUNT*** of runtime errors in the last ***THRESHOLD*** minutes.	1,2,4,5,6
	Build Errors: Get Help!	1. ***COUNT*** of runtime-exceptions in the last ***THRESHOLD*** minutes.	
	Others Available to Help!	1. ***ACTION***: Other users have set themselves as *'Available'*	
Help Giving	*Help other Students!*	1. ***ACTION***: Submit an assignment 2. **ACTION**: NO build errors/exceptions in **THRESHOLD**	1,3,4,6
	Help Your Classmates!	1. ***ACTION***: Submit an assignment at least ***THRESHOLD*** days before the submission deadline	2,4,5
Social Interaction	*You Are Available!*	1. ***ACTION***: User has set their status to Available	1,3,4
	Make A Post: Topical	1. ***ACTION***: Submit an assignment 2. **ACTION**: idle/no course activity in the last **THRESHOLD** period	3,4
	Make A Post: Reflection	1. ***ACTION***: Submit an assignment.	1,2,4,5

[1](Vygotsky, 1978), [2](Wood et al., 1976), [3](Bandura, 1997), [4](Astin, 1999), [5](Kolb, 1984), [6](Rotter, 1966)

While learning theory inspired the end goal of each intervention design, additional guidance was needed to determine the *triggers* for the interventions (see Table 4.5). As discussed in Section 2.3, the prior work of the Error Quotient (Jadud, 2006), Watwin score (Watson et al., 2013, 2014), NPSM (Carter et al., 2015) motivated error- and state-based trigger points from programming activity. Triggers based on other correlated activity such as level of and interactions with the community follow from the learning theories, e.g., no or low interaction or submitting assignments as a trigger for intervention.

www.manaraa.com

Table 4.5. Trigger thresholds

Trigger	Count/Duration	Description
Build or Runtime Error (**BRE**)	5	Number of errors in **TLE** *threshold* minutes before generating a new '*Ask for Help*' intervention.
Time since Last Error (**TLE**)	10 minutes	Time to look back when calculating **BRE** intervention generation
Time Without Errors (**TWE**)	10 minutes	Time before generating an '*Unanswered Questions*' intervention
Early Submissions Window (**ESW**)	1 day	Days before the assignment deadline to determine an '*Early Assignment Submission*' intervention generation
Days Without Activity (**DWA**)	1 day	Days without any post/reply activity before generating a '*Topical*' intervention
Unanswered Questions Window (**UQW**)	10 days	Number of days to look back for unanswered/marked helpful posts
Intervention Refresh Time (**IRT**)	10 minutes	Minimum time since last intervention prompt push before pushing more interventions

For each of the intervention designs, there were various course triggers that determined when to inject the intervention into the SPE. The main categories used for triggers were *count-based, action-based,* and *threshold-based* (see Table 4.4 and Table 4.5). Count-based triggers have a set threshold that needs to be met in order to trigger intervention generation and prompting. Help-seeking interventions are triggered by runtime/build errors and user's (self-determined) status. Help-giving interventions are triggered by assignment submissions. Finally, social interaction interventions are triggered by user status, assignment submissions, and an established threshold of recent feed activity in OSBLE+.

4.4.2 Evaluation Study

Six participants were recruited to elicit final feedback on the prototype designs (see Appendix G for full report). Participants were guided through a brief series of scenarios as in the previous prototype study. Unlike the previous prototype sessions which used mockup images, participants in this study used live interventions within the Visual Studio IDE, which interacted with an OSBLE+ server situated locally in the HELP Lab. Participant session interactions were recorded

www.manaraa.com

with screen capture software and audio was recorded; participants were encouraged to "think aloud" while interacting with the prototypes. In addition to getting their feedback on the intervention design and presentation, participants were also asked questions regarding their ***help-seeking***, ***help-giving***, and ***social interaction*** behavior in the problem-solving environment (see Appendix G).

Observations and feedback from the study were used to adjust minor details of the interventions and delivery prior to the interventions being used in the summative studies discussed in the following chapters. Additionally, the categories and variation of interventions were balanced more equally between the final help-seeking, help-giving, and social interaction interventions in the final design, with the inclusion of additional help-giving and assignment submission interventions. The study additionally helped both in refining the final prototype designs and refining the trigger counts and thresholds , as shown in Table 4.5.

4.5 Design Iteration 3

Based on the results of the previous iterations, this section describes the placement of the final interventions within the LMS and IDE and presents the final designs of each intervention category.

4.5.1 Intervention Placement and Access

Interventions were made available in two locations for users. The first location was directly on the OSBLE+ dashboard located between the actual post content and the filters and search bar (Figure 4.8). The OSBLE+ Suggestions bar was made to hide/show in order to reduce the clutter presented on the activity feed. The suggestions bar also included the user configuration options as described below. Users were given two visual indicators that there were new suggestions: a last updated title and a blinking indicator in the web browser tab.

www.manaraa.com

The second location for user access to the interventions was through the Visual Studio IDE (Figure 4.4), ensuring a tight coupling between the LMS and the problem-solving environment.

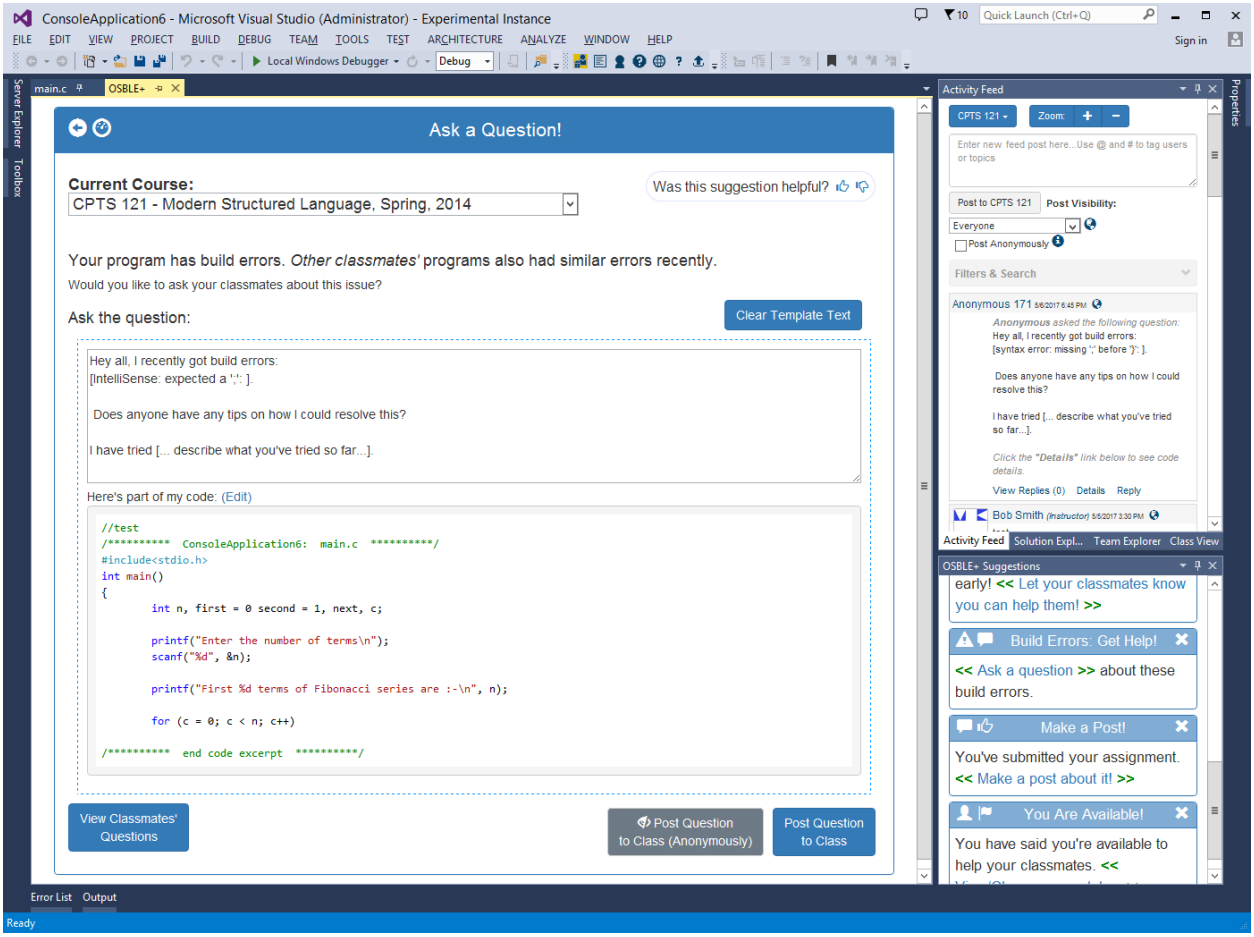

Figure 4.7. OSBLE+ Suggestions integration to the Visual Studio IDE (lower right)

By default, the OSBLE+ Suggestions window is docked below the Activity feed and the Solution Explorer in Visual Studio. The first content users see at the top of the IDE embedded OSBLE+ Suggestions window is a collapsible suggestions dashboard which gives users access to individual features (refer to Figure 4.9):

1. New suggestion notification icon. The header will also blink for 15 seconds when a new suggestion is present.

www.manaraa.com

2. View private conversations (a listing of all conversations between the current user and other users)
3. View your status/availability (see Figure 4.19)
4. View dismissed suggestions (see Figure 4.10)
5. View your user profile page (shows listing of user posts and replies)
6. Provide general feedback for the OSBLE+ suggestions system (open ended feedback submission)
7. View the OSBLE+ help page (URL to the OSBLE+ documentation)
8. Expand/Collapse the Suggestions Dashboard Options
9. Adjust the page zoom level (expand the zoom controls for the OSBLE+ Suggestions window in Visual Studio)

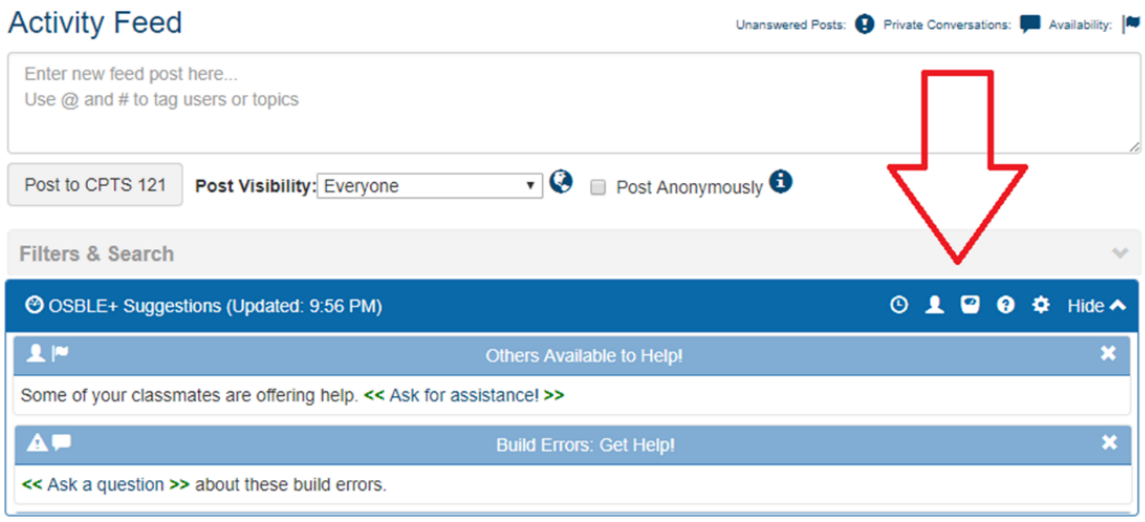

Figure 4.8. OSBLE Suggestions access via the OSBLE+ dashboard

www.manaraa.com

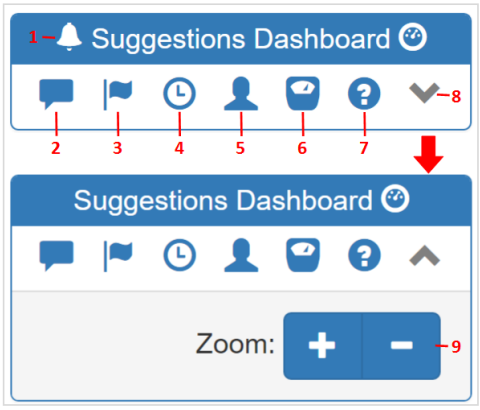

Figure 4.9. OSBLE+ suggestions dashboard in the IDE window

Figure 4.10. OSBLE+ dismissed suggestions overview

It is important to note that only one intervention of each type can be in an active state at a time (i.e. not "dismissed"). Any time an intervention generation threshold is met, if an intervention of the same type is still active it will be automatically dismissed with the assumption that the old event is no longer relevant. This ensures that each active intervention prompt will be the most relevant to the user.

4.5.2 Help-Seeking

For each of the interventions, a smaller prompt was injected into the top of the OSBLE+ Suggestions window (directly below the suggestions dashboard) with brief prompts and a link eliciting an action from the user. Figure 4.11 shows the prompts delivered for each of the help-seeking interventions.

www.manaraa.com

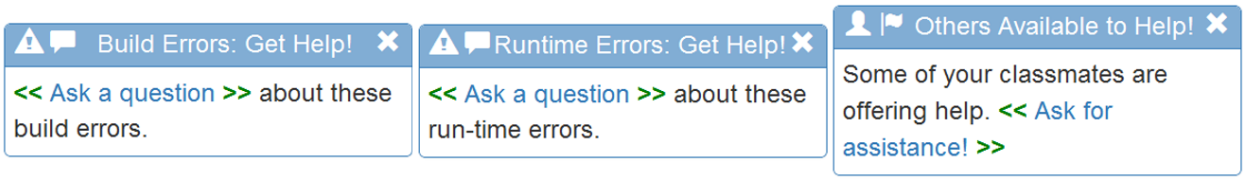

Figure 4.11. Help-Seeking intervention prompts

The prompts were designed to highlight the desired action. For the help-seeking category, interventions prompted the user to ask questions based on the predetermined triggers discussed in the previous section. When the prompting link was clicked, the main intervention window opened in the main window pane in Visual Studio (as seen in Figure 4.4).

4.5.2.1 Build/Runtime Errors: Get Help! (Runtime Errors and Build Errors)

The first of the help-seeking interventions is the runtime errors window (Figure 4.12). Feedback provided during the previous prototyping study motivated the addition of the option to post a question anonymously—an option available on all post and reply intervention prompts as well as the main activity feed.

www.manaraa.com

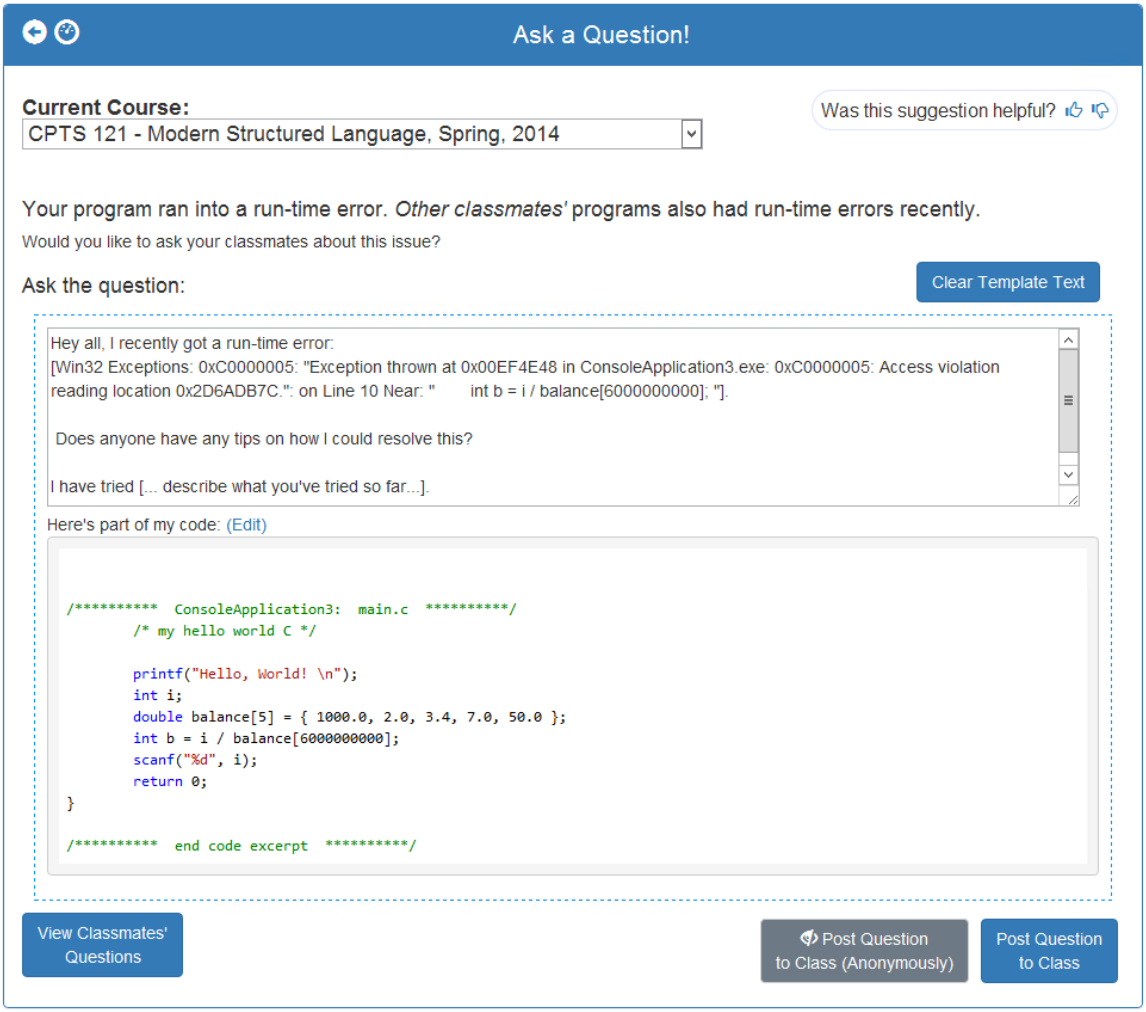

Figure 4.12. Runtime Errors: Get Help! help-seeking intervention

The build error intervention window (Figure 4.13) is similar in construction to the runtime error window with the only difference being that the auto-generated intervention will reference a build error and will be generated and injected when the build error threshold is met.

www.manaraa.com

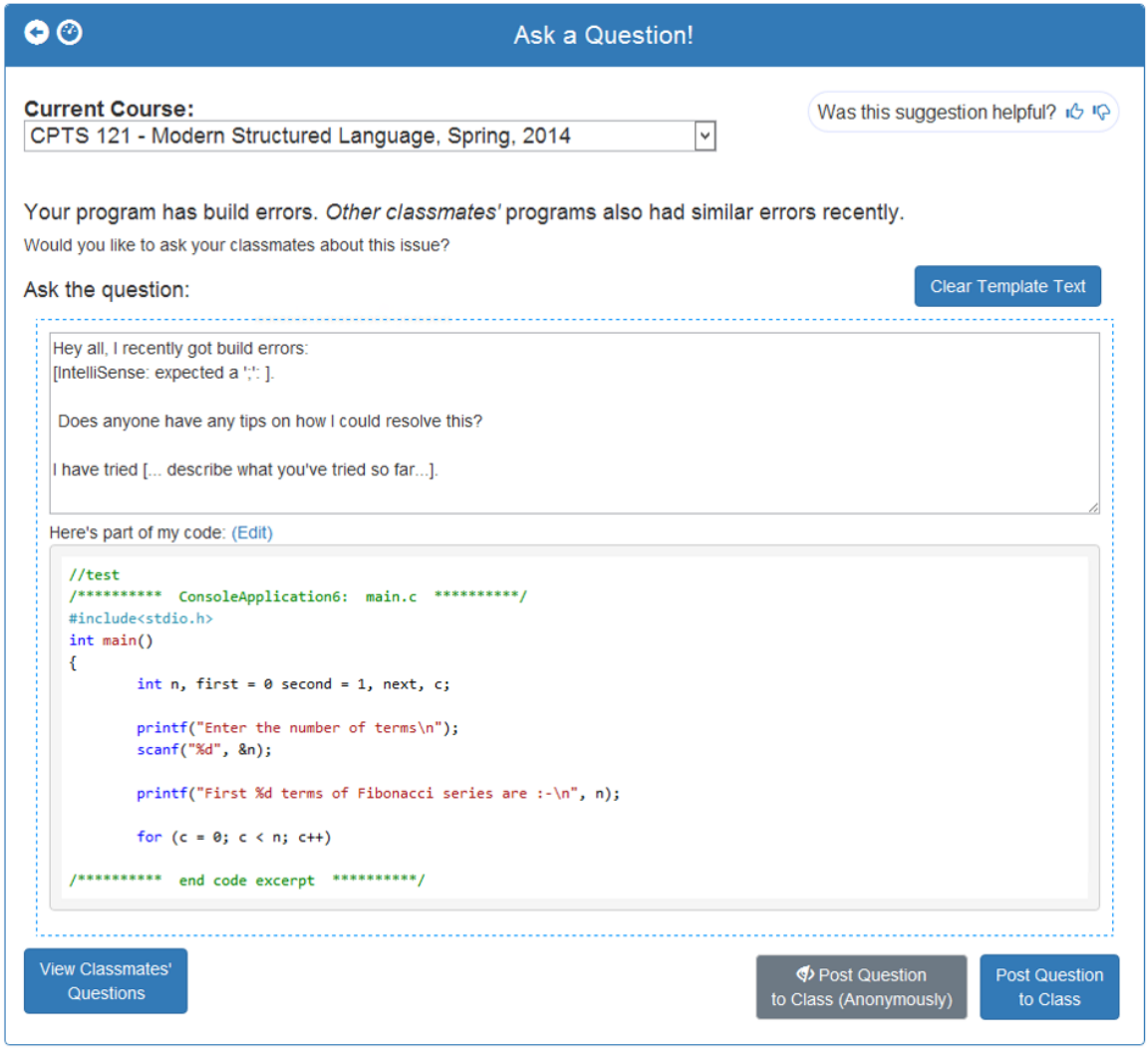

Figure 4.13. Build Errors: Get Help! help-seeking intervention

4.5.2.2 Others Available to Help! (Others are Offering Help)

The final help-seeking intervention focuses on indicating when others have set their status as available. In this window, users see a listing of users who have set their status, what their status is set as, and until what time they have set themselves as available (Figure 4.14). The window provides a scaffolded opportunity for the user to ask a question to one or more of the available users either anonymously or as themselves. This scaffolded question encourages the user to reflect

www.manaraa.com

on the problem they are having and prompts them to include some text about what they have tried to resolve the issue so far.

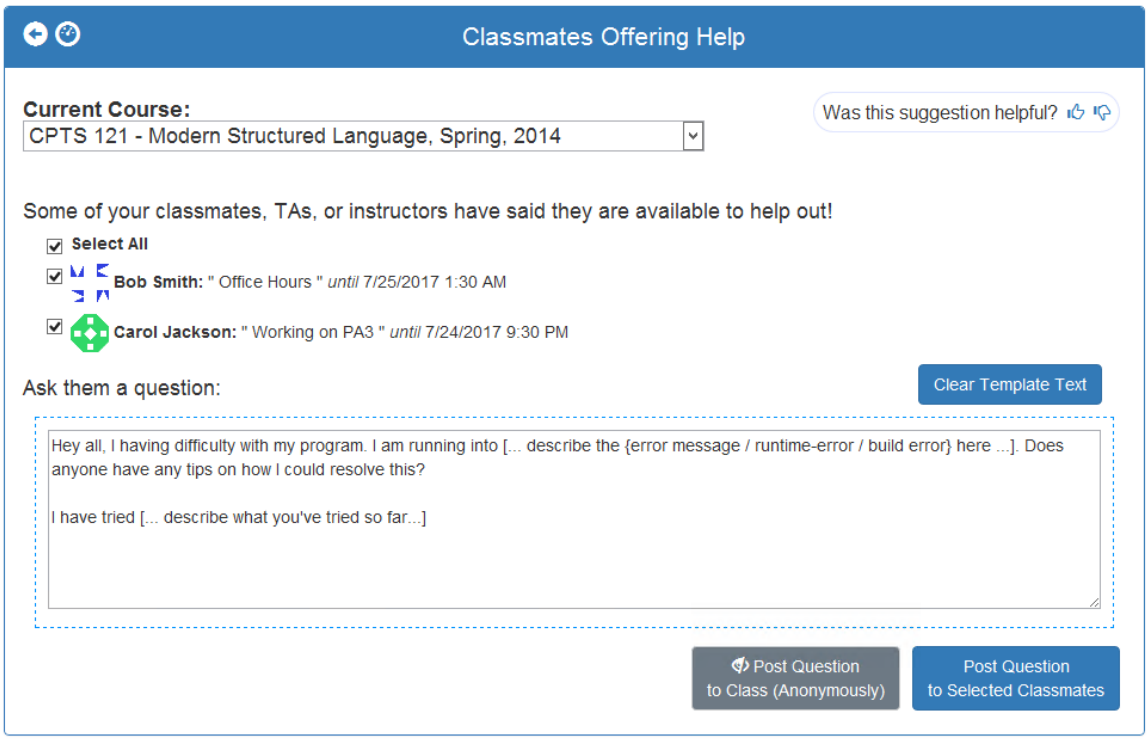

Figure 4.14. Others Available to Help! help-seeking intervention

4.5.3 Help-Giving

For the help-giving interventions, additional prompts are generated with the intent of encouraging additional social interaction with the community. Figure 4.15 below shows the prompts directing users to help classmates with unanswered questions (left) and to set their status to available because they have submitted their assignment early (right).

www.manaraa.com

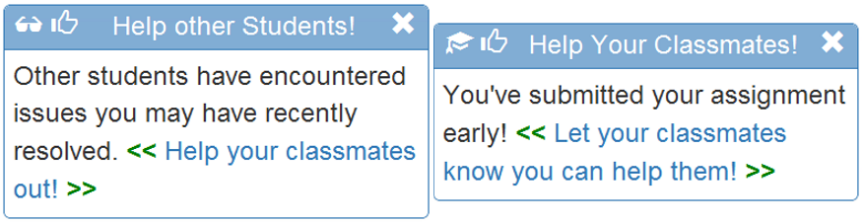

Figure 4.15. Help-Giving intervention prompts

4.5.3.1 Help other Students! (Unanswered Questions)

The first help-giving intervention (when clicking "Help your classmates out!") is focused on unanswered questions and directs users to a custom list of feed posts that consists of posts from users which either do not have replies or do not have any replies that have been marked as helpful (Figure 4.16). This intervention provides a description of what this list of feed posts are as well as guidance encouraging the user to either reply to questions or mark replies as helpful.

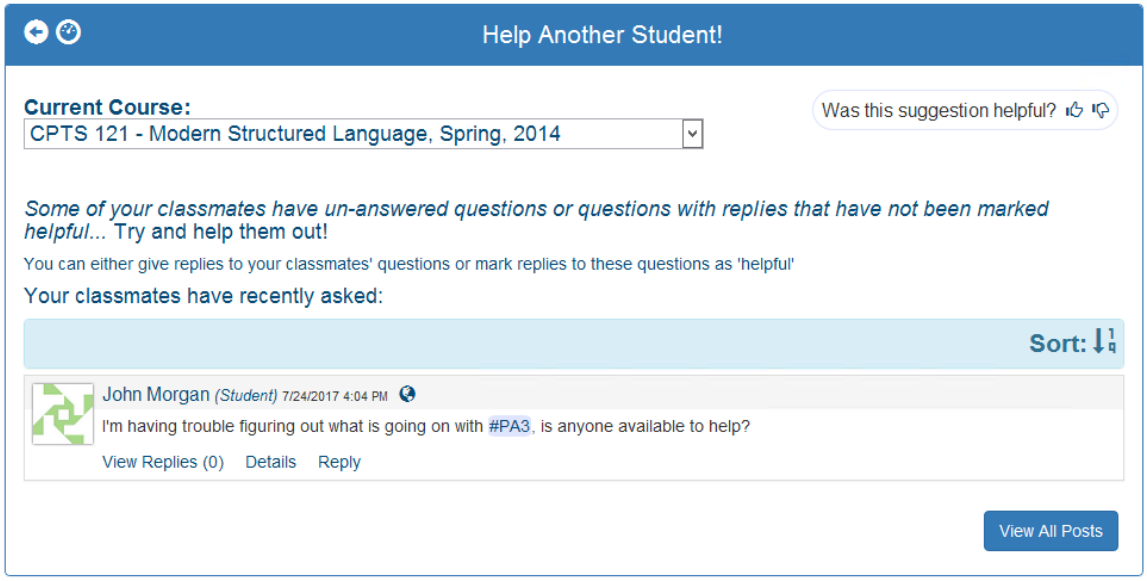

Figure 4.16. Help other Students! help-giving intervention

4.5.3.2 Help Your Classmates! (Early Assignment Submission)

When the early assignment submission threshold is met, the early assignment submission intervention is generated (Figure 4.17). This intervention is designed to encourage social

www.manaraa.com

interaction with classmates, provides a scaffolded means to reflect on the assignment just submitted (the template injects the assignment name into the template response), and encourages students to change their status and let the class know they have completed the assignment and are willing to help.

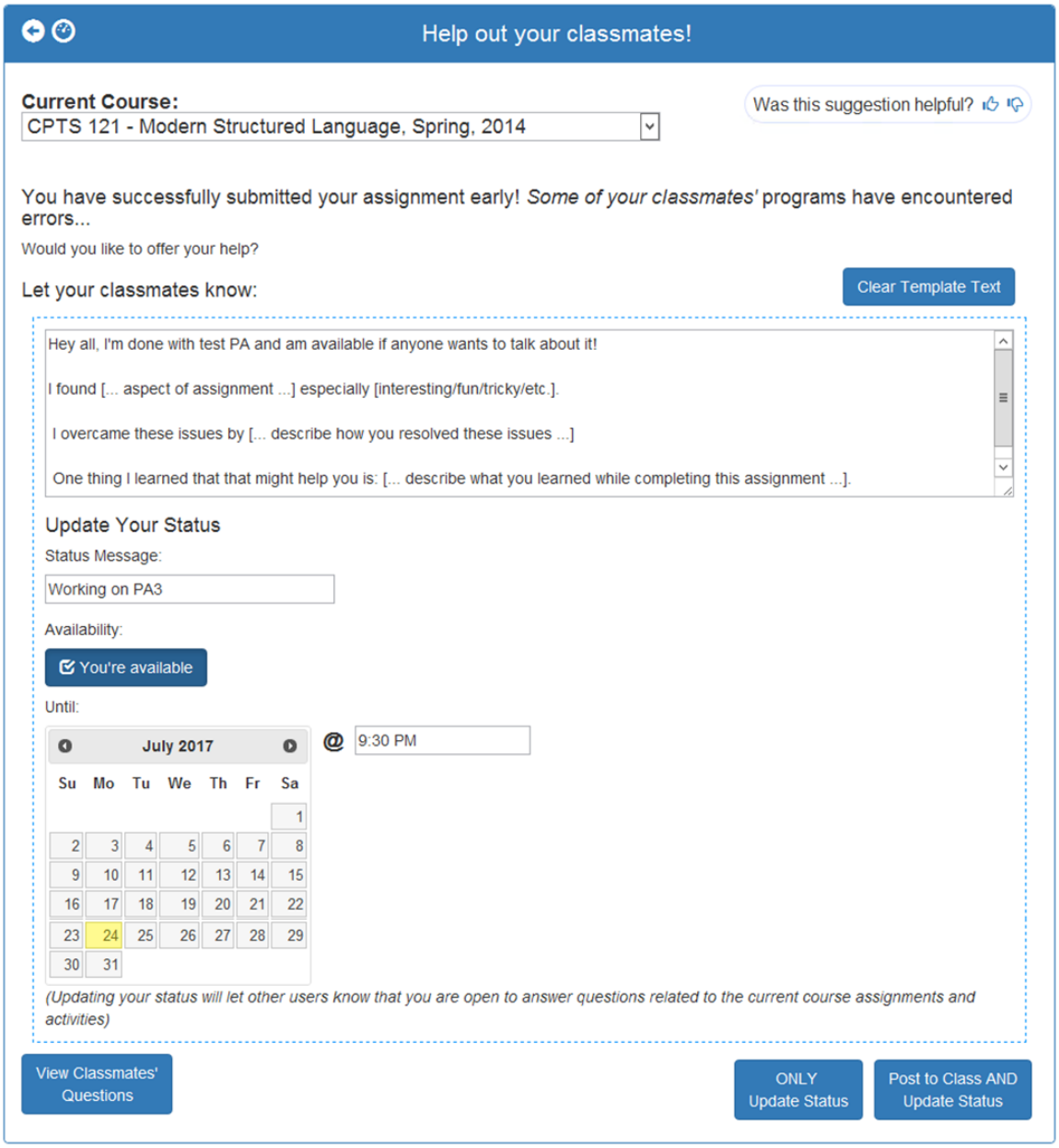

Figure 4.17. Help Your Classmates! (early assignment submission) help-giving intervention

www.manaraa.com

4.5.4 *Social Interaction*

The final intervention category is designed to stimulate social interaction (Figure 4.18). The first intervention (left) prompts users to look at or change their availability status and is generated after the user has used one of the status change pages. The remaining interventions are generated when there has been no activity in a threshold amount of time (middle) or when a user submits their assignment but has not met the early assignment submission threshold (right).

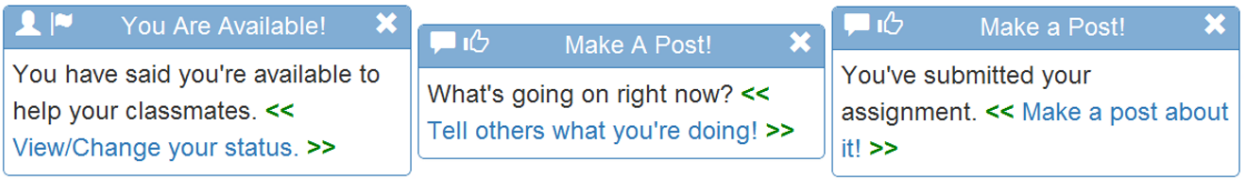

Figure 4.18. Social Interaction intervention prompts

4.5.4.1 *You Are Available! (User Set Status)*

When a user clicks the "View/Change your status" link, they will see the window shown in Figure 4.19. This window is like the "Others Available to Help!" window. It presents an overview of who is currently available, their status and availability time, options to change their status and availability, and the option to ask a question (1) anonymously, (2) directed at the entire class, or (3) directed only to selected users from the availability list.

www.manaraa.com

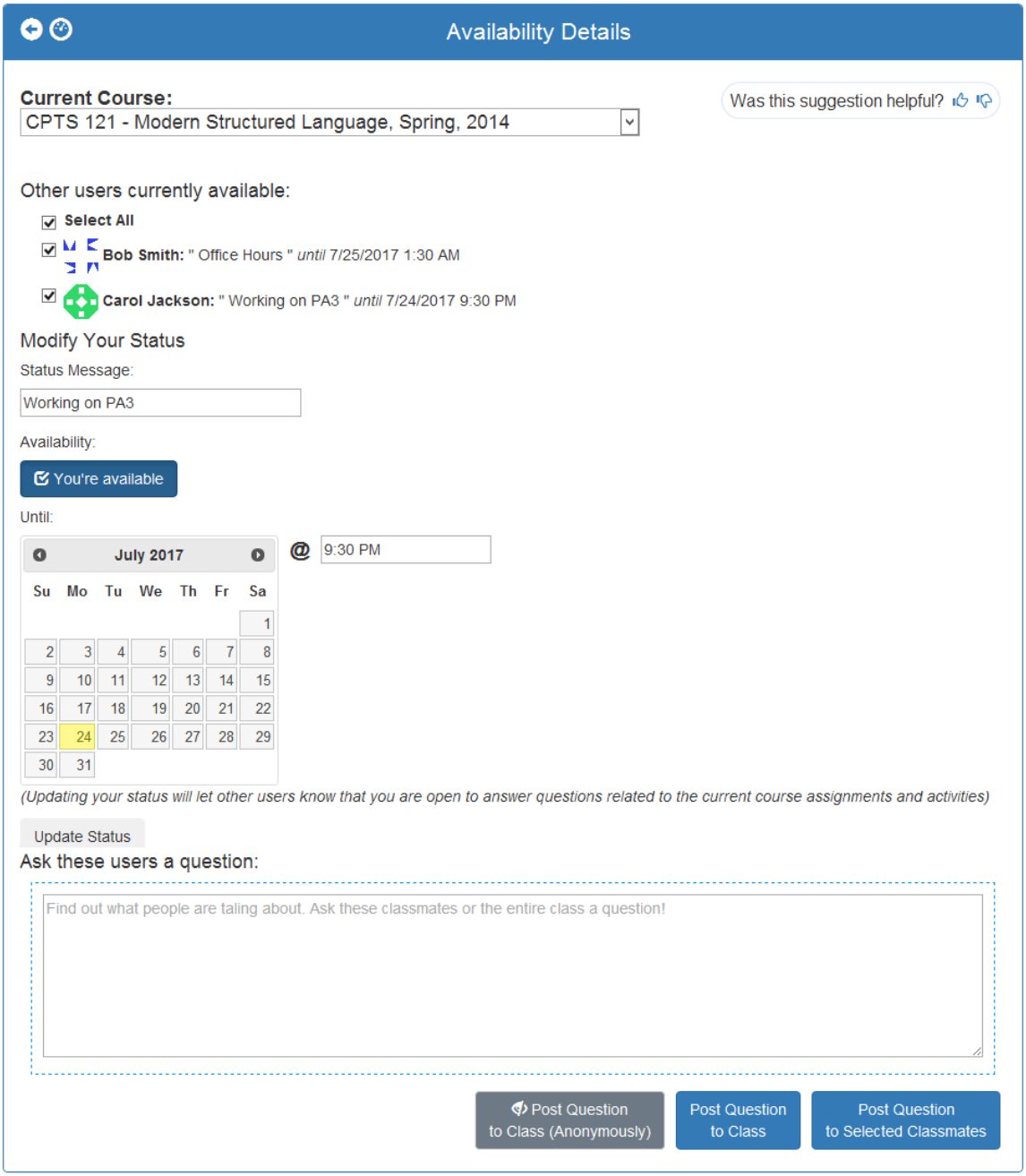

Figure 4.19. You Are Available! social interaction intervention

4.5.4.2 Make a Post! (Topical)

The middle intervention prompt in Figure 4.18 is intended to elicit social interaction when there has been no activity on the activity feed within the threshold amount of time. When generated, it provides a listing of popular topic hashtags most recently used on the activity feed

www.manaraa.com

and includes recent assignment names. Users can type a new post here and then post their message to the class publicly or anonymously. Additionally, the popular tagged topics are hotlinks and, when clicked, will open a tab on the OSBLE+ dashboard listing all posts or replies containing the matching tags or keywords.

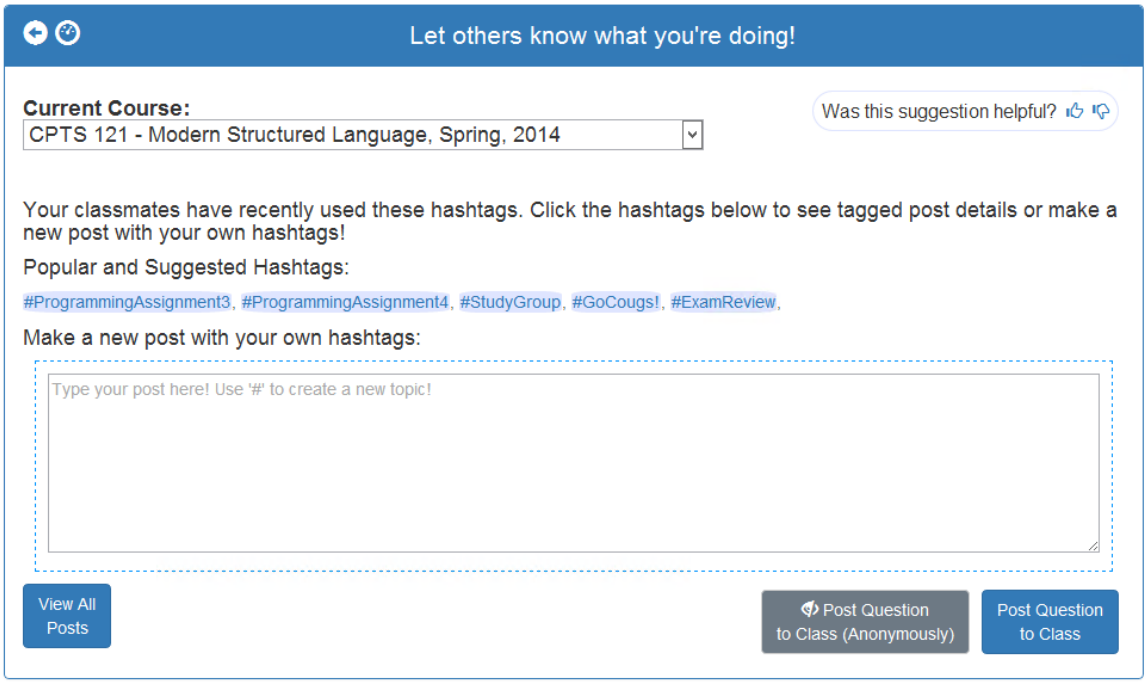

Figure 4.20. Make a Post! (topical) social interaction intervention

4.5.4.3 Make a Post! (Assignment Submission)

The final intervention is generated when a user submits an assignment and they have not met the early submission threshold. Upon clicking the "Make a post about it!" link (Figure 4.18, right), the user is presented with a window similar to the early assignment submission intervention (Figure 4.17) with a modified prompt omitting the early submission text. The template reflection text remains the same.

www.manaraa.com

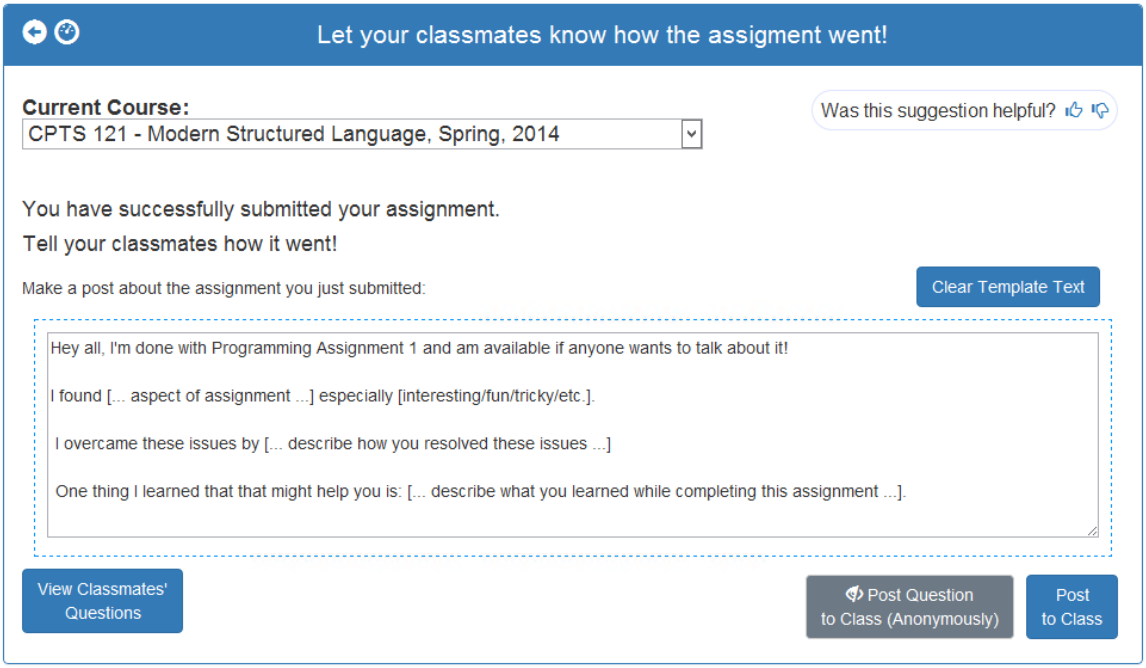

Figure 4.21. Make a Post! (assignment submission) social interaction intervention

4.5.5 Intervention Trigger Flow

As discussed in section 4.4 and shown in Table 4.4 and Table 4.5, the OSBLE+ Suggestions interventions follow a series of ***count-based***, ***action-based***, and ***threshold-based*** triggers for intervention generation. Figure 4.22 presents a flow chart depicting the ways in which interventions are generated based on the events generated within the system.

As shown in the figure, as events are logged in the system, the system takes different actions depending on the source of the event and historical data. As criteria are met (counts, actions, thresholds met), various interventions are generated.

www.manaraa.com

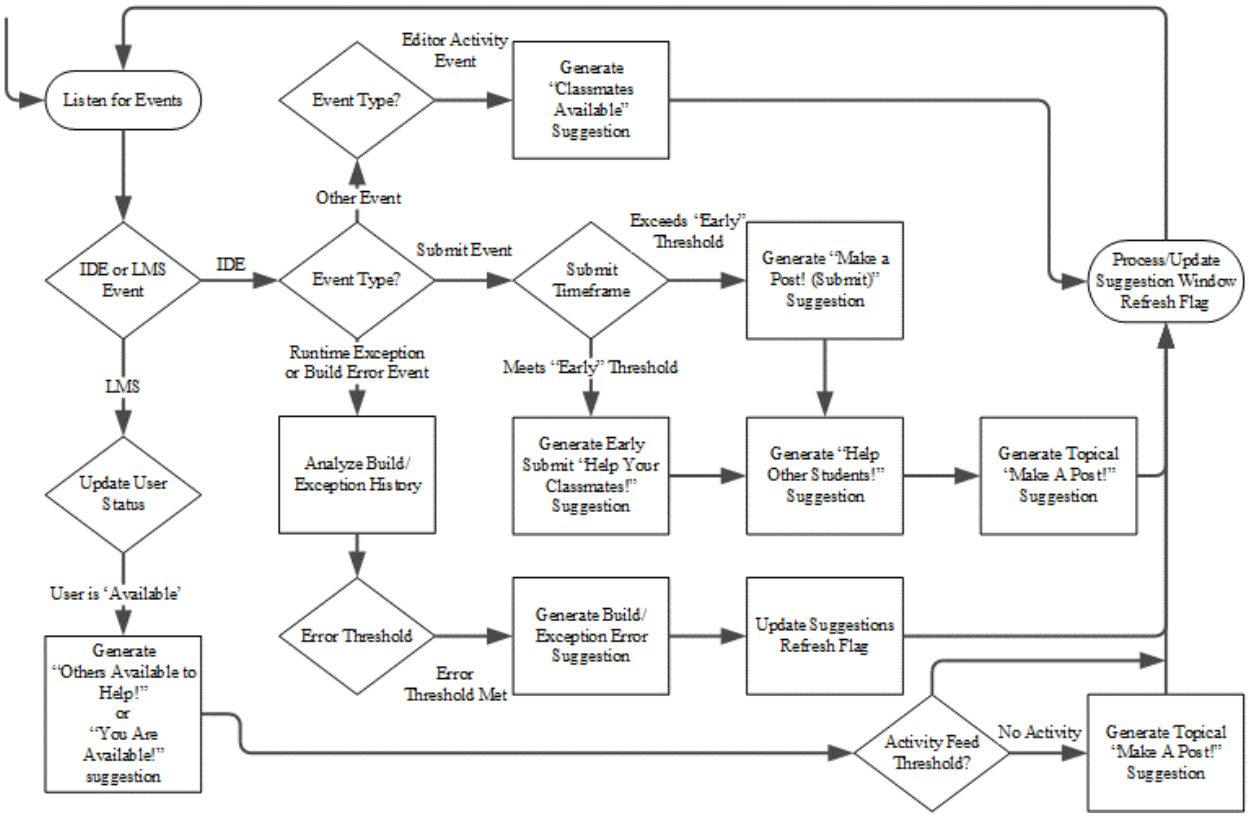

Figure 4.22. OSBLE+ Suggestions activity flow diagram

4.5.6 Intervention Documentation

A user's guide was created to introduce users to the system features. h. The guide included an installation guide for the Visual Studio plugin, a guide to using the plugin's basic functionality, and a guide to submitting assignments via the plugin. Additional guides described the OSBLE+ dashboard features (everything on the dashboard) and the OSBLE+ suggestions windows, including their location in the IDE and what each of the suggestion dashboard features did.

4.6 Discussion

Multiple surveys and participant studies were used to design the final set of interventions. Drawing on learning theories and previous research, the OSBLE+ Suggestion system was developed to steer learners towards desired behaviors that are strongly associated with learner

www.manaraa.com

success. In the end, I arrived at a system aligned with the GAMS framework. It takes a stream of user behaviors as inputs from the SPE, analyzes the behaviors in some manner, and generates interventions to influence learner behavior. In the next chapter, I present a summative study to empirically investigate the effectiveness of the interventions.

www.manaraa.com

CHAPTER 5

STUDY I

This chapter presents the first in a series of four empirical studies to investigate the effectiveness of the interventions motivated and developed in the previous chapters. Table 5.1 presents a set of refined research questions that these studies aimed to address. For each question, the table identifies the data collection and analysis strategies employed, along with the expected empirical outcomes.

Table 5.1. Intervention Research Questions

Research Question 1	
Can the SPE interventions promote positive changes[16] *in students' social and programming behaviors in the OSBLE+ environment?*	
Data Collection Strategy	Usability, user experience, and behavioral data collected in formative design studies; behavior data collected in summative study that compares best options
Analysis Strategy	Analyze design studies to identify and resolve usability and user experience issues. Analyze behavior and usage data[17] in summative study to determine most effective interventions
Expected Outcome	Clear correlations between given interventions and improved social and programming behaviors.
Research Question 2	
Will these interventions lead to positive changes in students' learning outcomes[18]*, and persistence*[19] *within the computer science discipline?*	
Data Collection Strategy	Quasi-experimental study run in conjunction with early computer science course with Control condition (no interventions in SPE) and Experimental condition (interventions in SPE)
Analysis Strategy	Analyze behavioral and persistence data in both conditions to determine whether statistically significant differences exist
Expected Outcome	Statistically-significant differences in the behavioral and attitudinal data of the two conditions

[16] Positive changes are defined as changes which can be associated with known metrics defining success. For example, social behaviors supporting *vicarious* and *enactive* engagement (Bandura, 1997) or programming behaviors and states linked to predicted success (Carter et al., 2015).

[17] Usage levels (interaction counts) of individual interventions will be used to determine individual intervention effectiveness.

[18] Learning outcomes measured by course performance, e.g. assignment, quiz, exam, and course grades.

[19] Attitudinal surveys and intent to major used as a proxy for measurement of potential persistence in computer science.

www.manaraa.com

The following 10 hypotheses more explicitly articulate the expected outcomes of the study in terms of the study's dependent variables:

H1: Students exposed to SPE interventions will exhibit significantly *better* social[20] behaviors than students not exposed to SPE interventions.

H2: Students exposed to SPE interventions will exhibit significantly *better* programming[21] behaviors than students not exposed to SPE interventions.

H3: Students exposed to SPE interventions will achieve significantly higher course grades than students not exposed to SPE interventions.

H4: Students with the highest levels of interaction with their peers will achieve significantly higher course grades than those with the lowest levels of interaction.

H5: Students exposed to SPE interventions will have higher perceived level *of self-efficacy* than students not exposed to SPE interventions.

H6: Students exposed to SPE interventions will have a higher perceived level of *sense of community* than students not exposed to SPE interventions.

H7: Students exposed to SPE interventions will have a higher perceived level of *self and peer learner motivation* than students not exposed to SPE interventions.

H8: Students exposed to SPE interventions will have a higher perceived level of *sociability* than students not exposed to SPE interventions.

---

[20] For purposes of analysis for this study, "*better*" social behaviors are defined as those that align with the motivating learning theories and related works discussed in Chapter 2; better social behaviors will be considered trends of increased social interaction quantity.

[21] For purposes of analysis for this study, "*better*" programming behaviors are defined as those that align with the motivating predictive programming models and related works discussed in Chapter 2; better programming practices manifest themselves in learners producing fewer error states (e.g. exceptions) or decreased amounts of time in error states.

www.manaraa.com

H9: Students exposed to SPE interventions will, express a significantly higher *intention to persist* in the computer science discipline than students not exposed to SPE interventions.

H10: Students exposed to SPE interventions will form a significantly closer coupled social networks than those not exposed to SPE interventions.

5.1 Methods

The studies were designed using a quasi-experimental mixed-methods approach with two treatment groups: Control and Experimental. Each treatment group corresponded with a comparable offering of CptS 121, the semester-long CS 1 course at Washington State University. Each course offering was as similar as possible: it had the same instructor, textbooks, and labs, assignments and exams.

A Mann-Whitney U test was performed to test for significant differences between the Control and Experimental treatments unless noted otherwise. Additionally, a Bonferroni corrected alpha value was used for significance to guard against Type I error in cases where multiple comparisons were made.

5.1.1 Participants

The fall 2016 semester of CptS 121 served as the Control treatment for this study. The class consisted of 263 students, of whom 207 students gave consent for their data to be analyzed for this study and 100 students completed all requested surveys. Missing survey responses were due to a combination of 1) students failing to complete one or more of the surveys for undisclosed reasons or 2) withdrawing from the course and stopping course-related activities (including study participation). As an incentive, participants were offered the opportunity to be entered into a

www.manaraa.com

drawing for gift cards to be delivered at the end of the semester. Additionally, 6 participants were removed from analysis as they produced 0 total events, bringing the total number of students in this treatment group to 201.

The spring 2017 semester of CptS 121 served as the Experimental treatment for this study. The class consisted of 212 students, of whom 142 students gave consent for their data to be analyzed for this study and 78 students completed all requested surveys. Missing survey responses were due to a combination of reasons (1) and (2) as described for the Fall 2016 data collection. Like the Control semester, participation in the study was incentivized with entry into a drawing for gift cards. Additionally, one participant was removed from analysis as they produced 0 total events, bringing the total number of students in this treatment group to 141.

5.1.2 Materials[22]

Materials for the study included the following:

- *SPEs*. The courses in this study used two alternative versions of the SPE— one without the interventions (the Control treatment) and one with the interventions (the Experimental treatment). As discussed in Chapter 3, the SPEs were implemented as plug-ins to the Visual Studio IDE used by students to implement their programming solutions in the courses.
- *Surveys*. Surveys were used to gauge student attitudes at certain points in the semester. The surveys consisted of questions from the MSLQ: Motivated Strategies for Learning Questionnaire (Pintrich, Smith, Garcia, & McKeachie, 1991), C++ Self-Efficacy (Ramalingam & Weidenbeck, 1998), Classroom Community scales (Rovai, 2002), Sociability Scale (Kreijns, Kirschner, Jochems, & van Buuren, 2004) and the International Personality Item Pool

[22] See Appendices J and K

www.manaraa.com

(IPIP) California Psychological Inventory (CPI) sociability scale (Goldberg et al., 2006). The System Usability Scale was also used to elicit opinions on the usability of the OSBLE+ system. In addition, custom survey questions were designed for gauging usefulness of the social interventions and attitudes not covered by the online surveys.

5.1.3 Independent Variable

The study's independent variable was the presence of the *SPE Intervention*. In the Control treatment, participants used a version of the SPE that did not contain exposure to the interventions described in the previous chapter. In contrast, participants in the Experimental treatment used a version of the SPE that included those interventions.

5.1.4 Dependent Variables

Log data was collected through the Visual Studio plugin and the OSBLE+ LMS. The data consisted of both data on students' programming processes, and data on students' social activities within the environment (Table 5.2). These log data were used as a foundation for measuring the dependent variables described below.

It is important to note is that during the fall 2016 semester and partially into the spring 2017 semester, not all build error events were being properly logged. The logging error was not noticed until it was too late to make a correction. For that reason, I was forced to omit build errors from the analysis of Study I.

www.manaraa.com

Table 5.2. OSBLE+ log data

Dependent Variable Related to	Event	Description
Social Activities	Ask-For-Help*	Top-level post initiated from the IDE
	Hashtag**	Clickable link topic tag in posts/replies, e.g. #PA1Help
	Initiate Reply**	Clicking the "Reply" button on activity feed
	Keyword Search**	Using the "Filters & Search" feature on the activity feed (Figure 4.8)
	Mention**	Clicking an @Mention user tag, e.g. @BobSmith
	Post*	A top-level post initiated from the LMS
	Post Details**	Clicking the "Details" link on an activity feed top-level post
	Reply Marked Helpful*	Clicking the "Thumbs up" on an activity feed reply
	Reply*	Posting a reply to a top-level post on the activity feed
	View Replies**	Clicking the "View Replies (#)" link on a top-level activity feed post
Programming Processes	Build	IDE build
	Cut-Copy-Paste	Using the cut, copy, or paste action in the IDE text editor
	Debug	IDE debugging breakpoint added
	Editor	Editing solution files within the IDE - keyboard interactions with a 1-minute threshold for logging
	Build Error	IDE Build errors
	Exception	IDE runtime exceptions
	Save	Saving of solution files within the IDE
	Submit	Assignment submissions initiated through the OSBLE+ VS plugin

**Aggregate Activity Feed; **OSBLE Interaction*

5.1.4.1 Participation

Level of participation was measured by the number of posts, number of replies, number of other logged interactions such as interactions with the OSBLE+ activity feed interface, and interactions with individual interventions. The latter, intervention interaction, was important as it was the main way of determining the possible impact of each intervention based on its actual usage. Additionally, perceived usage by participants was collected in the final survey.

Changes in participation over time. Data collection also focused on changes in participation levels over time—from the start of the course to the end of the course. Analysis of data in this category looked for changes between participation levels and their relation to other variables of interest.

www.manaraa.com

5.1.4.2 Programming Activity

Each of the eight programming-related events in Table 5.2 was used to assess changes in participant programming activity. Exceptions and build errors were used for the "time in error state" variable. Time in error was calculated by using a sequence of multiple errors (runtime exceptions and build errors) during threshold periods of 2 minutes or less (see Table 4.5), i.e., time between sequences of errors less than 2 minutes were summed for total time in error states. Errors occurring greater than 2 minutes from the last error started a new time in error sequence. Calculating this value was used to consistently represent student error state. The remaining programming log data (build, cut-copy-paste, debug, editor, save, and submit) were considered when looking for trends in programming behaviors.

5.1.4.3 Performance – Achievement & Activity Levels

Student' course grades were used as a proxy for their programming performance. These included grades for programming assignments, labs, participation, quizzes, exams, and the overall course (final course grade).

5.1.4.4 Student Attitudes

Pre-, mid-, and post surveys were used to collect attitudinal data on self-efficacy (Adapted C++ Self-Efficacy Survey), sense of community (Classroom Community Scale), self and peer learner motivation (Motivated Strategies for Learning Questionnaire), system usability (System Usability Scale), self and system sociability (IPIP CPI self-sociability and the Sociability Scale) as well as additional surveys to measure participant likelihood of intervention use, perceived usefulness of interventions, and intent to major. Analysis of intent to major considered both CS major and Non-majors separately. The surveys used can be found in Appendix J

www.manaraa.com

5.1.4.5 Social Network Analysis[23]

Social Network Analysis (SNA) was performed to study the changes in behavior of the students in the two treatment groups. SNA can offer numerical and visual insights into the types of relationships and interactions that occur between individuals, groups, and communities within a large number of interactions provided (Scott, 2017). SNA has been commonly used to understand students' interactions on discussion forums (Dawson, 2008; Gašević, Joksimović, Eagan, & Shaffer, 2018; Wang & Li, 2007).

In this study, we investigated commonly used node-level SNA metrics to measure *social centrality*, including degree, closeness, betweenness, and eigenvector centrality. These metrics are usually considered a measure of popularity because they indicate the number of connections around a specific node (Carrington, Scott, & Wasserman, 2005). More formally, degree centrality captures the local structure surrounding the node and indicates the number of connections a node has (Freeman, 1978).

Closeness and betweenness measures, respectively, the distance of a given node to all other nodes in a network and the number of shortest paths between all other nodes to a given node. In other words, closeness is associated with the likelihood that a given node connects with other nodes. Alternately, betweenness can be viewed as a metric of brokerage: the extent to which a node bridges distinct parts of the network. Lastly, eigenvector centrality gives higher values to a node the more it is connected to other highly prominent nodes.

[23] Social network analysis was performed, in part, as a collaborative work for a conference paper accepted to ICALT 2019: 19th IEEE International Conference on Advanced Learning Technologies (Olivares, Adesope, Hundhausen, Ferreira, & Gašević, in press)

www.manaraa.com

Besides the nodes degree measures, I also analyzed the structural features at the network-level, including density, diameter, and average path length—the most used features to examine networks on a SNA (Scott, 2017). Density measures the proportion of actual connections between nodes to all connections. The goal is to measure the extent to which all members of a network are connected to each other (Wasserman & Faust, 1994). Diameter measures the maximum eccentricity of any node in a network—that is, the maximum distance between any two nodes. Finally, average path length measures the average number of steps along the shortest paths for all pairs of network nodes.

5.1.4.6 System Usability

System usability was measured using the System Usability Scale to get a quick but reliable assessment of each treatment group's perception of system usability. This was important because a group's perception of the system usability could play a part (positive or negative) in the usage of the system and influence results.

5.1.5 Procedure

Control data was collected in the fall 2016 semester and Experimental data was collected in the spring 2017 semester. Students in both treatments were required to use the SPEs in the course—both as the learning management system, and as the programming environment in which their solutions to course programming assignments were implemented and submitted. This was done with the full cooperation of the course instructor.

The fall 2016 and spring 2017 offerings were both taught by the same instructor, covered the same content, and used the same course materials.

At the start of the course (within the first 2-3 weeks), students were given the opportunity to take the pre-attitudinal survey (which included the informed consent for study participation). At

www.manaraa.com

the mid-point of the semester, additional an attitudinal survey was administered to those who had given consent to participate in the study. A final post-attitudinal survey was given at the end of the semester (within the last 2 weeks) to gauge changes in attitude since the start of the course. Students' activities within the SPE were logged throughout the duration of each course.

5.2 Results

Table 5.3 presents a summary of the log data collected for each treatment group. Event counts are broken down by treatment. These counts are further normalized on a per-student basis to provide a basis for comparing activity levels across treatments. Below, I present the results related to each dependent variable.

5.2.1 Participation

Participation levels were based on five different activity events; posts, replies, helpful marks given to replies, total event activity in OSBLE+, and OSBLE interaction events (interactions on the activity feed). In order to mitigate Type 1 error and test significance of all participation activity, I constructed an aggregate measure consisting of the sum of all social metrics analyzed above; posts, replies, and helpful marks given to replies.

www.manaraa.com

Table 5.3. OSBLE+ event log data for Study I

Fall 2016 (Control)			*Students: 201*	*Spring 2017 (Experimental) Students: 141*	
Source	**Event**	**Event Counts**	**Counts/ Student**	**Event Counts**	**Counts/ Student**
IDE	Ask-For-Help	11	0	0	0
	Build	175,178	846	141,481	996
	Cut-Copy-Paste	245,222	1,185	172,947	1,218
	Debug	551,535	2,664	376,557	2,652
	Editor	383,987	1,855	275,842	1,943
	***Build Error*	221	1	41,654	293
	***Build Error Counts*	699	3	308,186	2,170
	Exception	16,189	78	15,183	107
	Save	243,849	1,178	168,060	1,184
	Submit	2,766	13	1,587	11
Activity Feed	Helpful Mark	143	1	39	0
	OSBLE Interaction	4,501	22	3,450	24
	Post	263	1	116	1
	Reply	244	1	115	1
***Total Event Counts:**		**1,623,888**	**8,079**	**1,155,377**	**8,194**

**Build errors and counts are excluded from the total as each build error event is also a build event and each build error event can have many build errors; **Does not include some build error events*

Comparison results indicated that there were no significant differences (Table 5.4) with respect to all three measures. Additionally, the same test was performed for programming assignment (PA) periods instead of the entire course period. These periods were determined by using the final solution(s) submitted to each assignment and then looking at IDE activity related to the solution files to determine the start date and time a user was actively programming. There were nine programming assignments (PA) during each semester. This analysis indicated a similar result, with no significant differences between treatments (Table 5.4).

www.manaraa.com

Table 5.4. Study I participation comparisons

Comparison Group		*U*	*p*<=	*r*	*M* Ranks**	Mdn**
Course	Aggregate Feed Activity	13994	.827	.012	170:172	0:0
	OSBLE Interactions	12491	.062	.101	183:163	10:7
	Total Events	14021	.868	.009	170:172	6107:5635
Programming Assignments	Aggregate Feed Activity	13964	.784	.015	170:173	0:0
	OSBLE Interactions	14052	.895	.007	171:172	6:5
	Total Events	13065	.219	.066	164:177	5323:5422:

*N = 342; **All ratios are Experimental:Control, *significant at adjusted α <=.0167*

Thus, these results do not support **H1**; the two groups were not significantly different with respect to their levels of participation.

5.2.2 Programming Activity

Comparisons of the programming activities were performed on seven of the eight programming metrics logged by the OSBLE+ system (Table 5.5). Build errors were excluded from Study I due to missing data. Results indicated that only submit events were determined to be significantly different between the treatment groups (Table 5.5) with the Control treatment submitting significantly more often. The same result was found when analyzing activity during the programming assignment periods (Table 5.6).

Table 5.5. Study I programming activity comparison

Programming Metric	*U*	*p*<=	*r*	*M* Ranks**	Mdn**
Build	13847	.719	.019	174:170	674:641
Cut-Copy-Paste	13919	.779	.015	170:173	824:906
Debug	13255	.309	.055	165:176	1012:1111
Editor	13779	.663	.024	169:173	1681:1731
Exception	13322	.346	.051	178:167	75:57
Save	13616	.537	.033	175:169	902:764
Submit	11144	.001*	.183	150:187	10:11

*N = 342; **All ratios are Experimental:Control, *significant at adjusted α <=.007*

www.manaraa.com

Table 5.6. Study I programming activity comparison during programming assignment periods

Programming Metric	*U*	*p*<=	*r*	*M* Ranks**	*Mdn***
Build	13471	.437	.042	167:175	631:621
Cut-Copy-Paste	13214	.288	.057	165:176	821:880
Debug	12379	.046	.108	159:180	843:1214
Editor	13116	.241	.063	164:177	1457:1637
Exception	13423	.406	.045	166:175	49:61
Save	13575	.508	.036	167:174	808:793
Submit	11716	.006*	.148	154:184	11:14

*N = 342; **All ratios are Experimental:Control, *significant at adjusted α <=.007*

Because build errors were not logged in Study I, time in error state was modified to account only for runtime exceptions. Results indicated that, there was no significant difference between groups. (Table 5.7).

Table 5.7. Study I time in error state comparisons

	U	*p*<=	*r*	*M Ranks***	*Mdn***
Course	13584	.514	.035	167:174	9:11
Assignments	13368	.371	.048	166:175	8:11

*N = 342; **All ratios are Experimental:Control, *significant at p <=.05*

Thus, these results do not support **H2**; the programming activity levels of the two groups did not differ significantly.

5.2.3 Performance - Achievement

Table 5.8 compares student performance with respect to course grades on seven course deliverables. No significant differences were found between the treatment groups, suggesting that **H3** was not confirmed.

www.manaraa.com

Table 5.8. Study I performance comparisons

Performance Metric	*U*	*p*<=	*r*	*M* Ranks**	*Mdn***
Final Course Grade	13620	.541	.033	175:169	.82:.81
Lab Participation	14022	.867	.009	173:171	1:1
Quizzes	13728	.623	.027	175:169	.79:.80
Programming Assignments	13643	.557	.032	168:174	.86:.79
Written Midterms	12705	.103	.088	161:179	.74:.78
Lab Final Exam	12498	.063	.101	183:163	.80:.75
Written Final Exam	13191	.276	.059	165:176	.71:.73

*N = 342; **All ratios are Experimental:Control, *significant at adjusted α <=.007*

5.2.4 Performance - Activity Levels

A Kendall's tau-b correlation was performed in order to determine if there was a relationship between aggregate social activity level (posts, replies, and helpful marks given to replies), OSBLE activity feed interactions, total system activity, and course performance. Kendall's tau-b was chosen over a Spearman's rank order correlation due to a high number of tie rank values in the data set. These tests found a significant weak to moderate positive correlation between course grades and all three metrics of social activity in the Control treatment (Table 5.9).

Table 5.9. Study I Control treatment activity level performance correlations

Performance Metric	Aggregate Social Activity		OSBLE Interactions		Total Events	
	r_τ	$p <$	r_τ	$p <$	r_τ	$p <$
Final Course Grade	.271	**.001**	.358	**.001**	.495	**.001**
Lab Participation	.170	**.001**	.228	**.001**	.360	**.001**
Quizzes	.220	**.001**	.241	**.001**	.361	**.001**
Programming Assignments	.247	**.001**	.348	**.001**	.460	**.001**
Written Midterms	.206	**.001**	.278	**.001**	.397	**.001**
Lab Final Exam	.226	**.001**	.275	**.001**	.382	**.001**
Written Final Exam	.302	**.001**	.369	**.001**	.446	**.001**

N = 201; 1-tailed - significant at adjusted α <=.005

Interestingly, in the Experimental treatment, no significant correlations were detected between any of the performance metrics and aggregate social activity. However, both OSBLE interactions

www.manaraa.com

and total event activity were significantly correlated with all course performance measures (Table 5.10). Correlation levels were weak to moderate, with *r* values ranging from 0.16 to 0.53.

Table 5.10. Study I Experimental treatment activity level performance correlations

Performance Metric	Aggregate Social Activity		OSBLE Interactions		Total Events	
	r_τ	$p <$	r_τ	$p <$	r_τ	$p <$
Final Course Grade	.092	.075	.228	**.000**	.525	**.001**
Lab Participation	.052	.218	.166	**.003**	.264	**.001**
Quizzes	.098	.062	.257	**.000**	.448	**.001**
Programming Assignments	.123	.027	.251	**.000**	.523	**.001**
Written Midterms	.048	.227	.163	**.002**	.445	**.001**
Lab Final Exam	.044	.254	.228	**.000**	.404	**.001**
Written Final Exam	.064	.163	.205	**.000**	.421	**.001**

N = 141; 1-tailed - significant at adjusted α <=.005

5.2.5 Student attitudes

Student attitudinal surveys were used to address **H5-H9**. Only self-sociability, **H8**, was significantly different between treatment groups. Results indicated the Control group had a higher mean ranked self-sociability score, suggesting that they were a more social group of participants. All other attitudinal comparisons, addressing **H5-7** and **H9**, were not significant (Table 5.11).

Table 5.11. Study I attitudinal comparisons

Attitudinal Metric	*U*	*p*<=	r	*M* Ranks**	*Mdn***
Coding Self-Efficacy	3881	.956	.003	47:49	47:47
CCS: Connectedness	3364	.115	.084	83:95	0:1
CCS: Learning	3810	.791	.014	88:90	0:0
CCS: Total	3442	.080	.094	84:97	-1:1
MSLQ: Self-Learning	3795	.758	.017	91:88	-2:-1
MSLQ Peer-Learning	3623	.415	.044	90:90	-1:0
Self-Sociability	4494	**.015**‡	.130	95:116	3.1:3.2
System-Sociability	4892	.130	.033	71:75	-3:-2

N = 178; **All ratios are Experimental:Control, *significant at adjusted α <=.05
†significant at adjusted α <=.017; ‡significant at adjusted α <=.025

www.manaraa.com

Comparison of intent to major showed that the Control treatment lost five CS majors (10 CS majors changed their response in the post survey while five non-CS majors changed their responses to yes) equating to a 9% net loss of CS majors. In contrast, the Experimental treatment gained one CS major (two CS majors changed their response in the post survey while three non-CS majors changed their responses to yes) equating to a 4% net gain of CS majors.

5.2.6 Social Network

To investigate **H10**, the post-reply relationships for each of the treatment groups were analyzed as a social network graph using Gephi ("Gephi - The Open Graph Viz Platform," n.d.). Table 5.12 presents the results of the social network measures for the Control and Experimental groups. These results show that the Experimental group exhibited higher values for two metrics: eigenvector and average path length. All other network descriptive measures were lower. However, none of the differences was statistically significant, indicating that **H10** was not confirmed.

Table 5.12. Study I social network statistics

Descriptives	Control Group	Experimental Group	*U*	*p*<=	*r*	*M* Ranks**	*Mdn***
Number of Messages	1204	775					
Degree	2.898	2.495	5447	.462	.040	105:111	4:4
Weighted Degree	11.093	7.243	5542	.603	.028	106:110	6:6
Closeness	.326	.269	5297	.275	.059	104:112	.262:.277
Betweenness	72.24	68.21	5769	.979	.001	108:108	0:0
Eigenvector	.169	.216	4863	.044	.109	117:100	.241:.132
Density	.027	.024					
Diameter	7	7					
Average Path Length	3.086	3.101					

*Note: N = 215 – not all participants participated in activity feed discussions; *significant at adjusted α <=.01*
***All ratios are Experimental:Control,*

Figure 5.1 and Figure 5.2 present directed graphs of each treatment group's social networks. We can see that the instructors (red) and TAs (green) are the main nodes in both graphs. Nodes in

www.manaraa.com

these graphs are scaled to their degree (connections both in and out), making it clear that the instructors and TAs are more connected overall than students (blue). A visual inspection of these graphs also verifies that, though not significantly different, the Control group was overall more active during the semester (note there are visually more disconnected nodes in Figure 5.2).

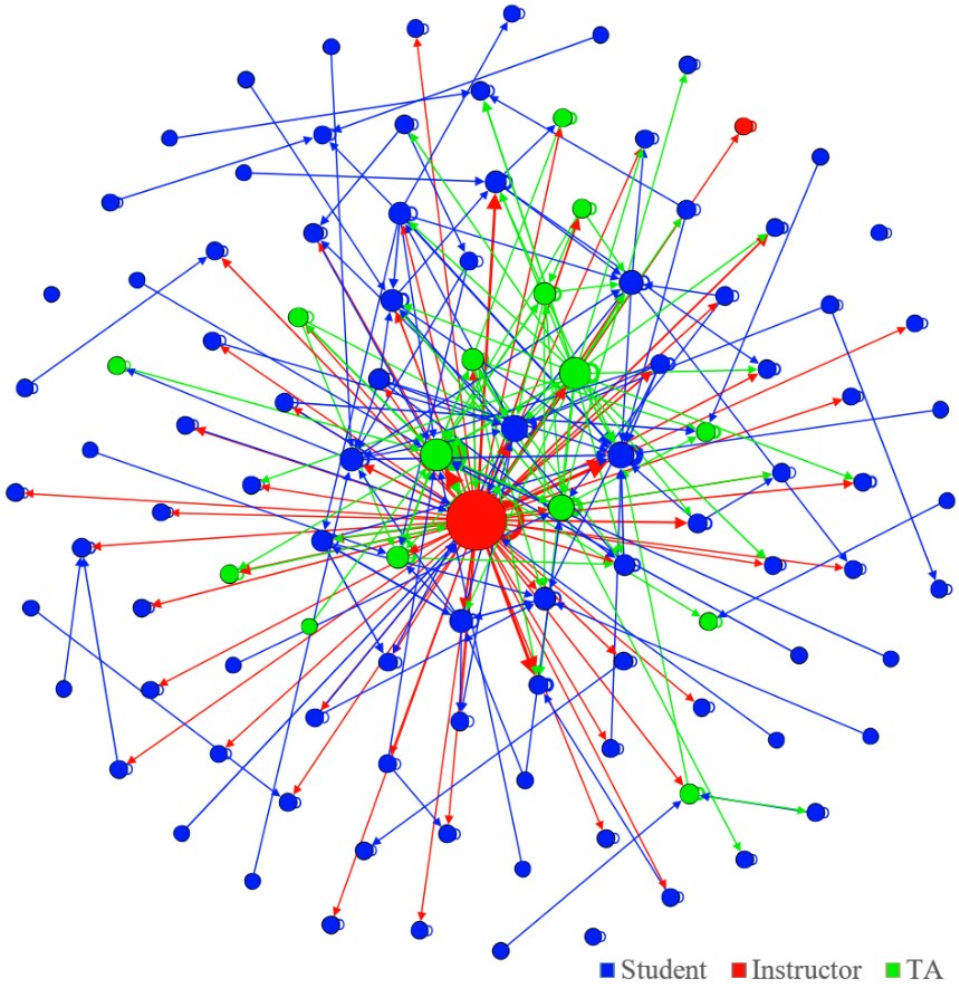

Figure 5.1. Study I Control treatment social network graph

www.manaraa.com

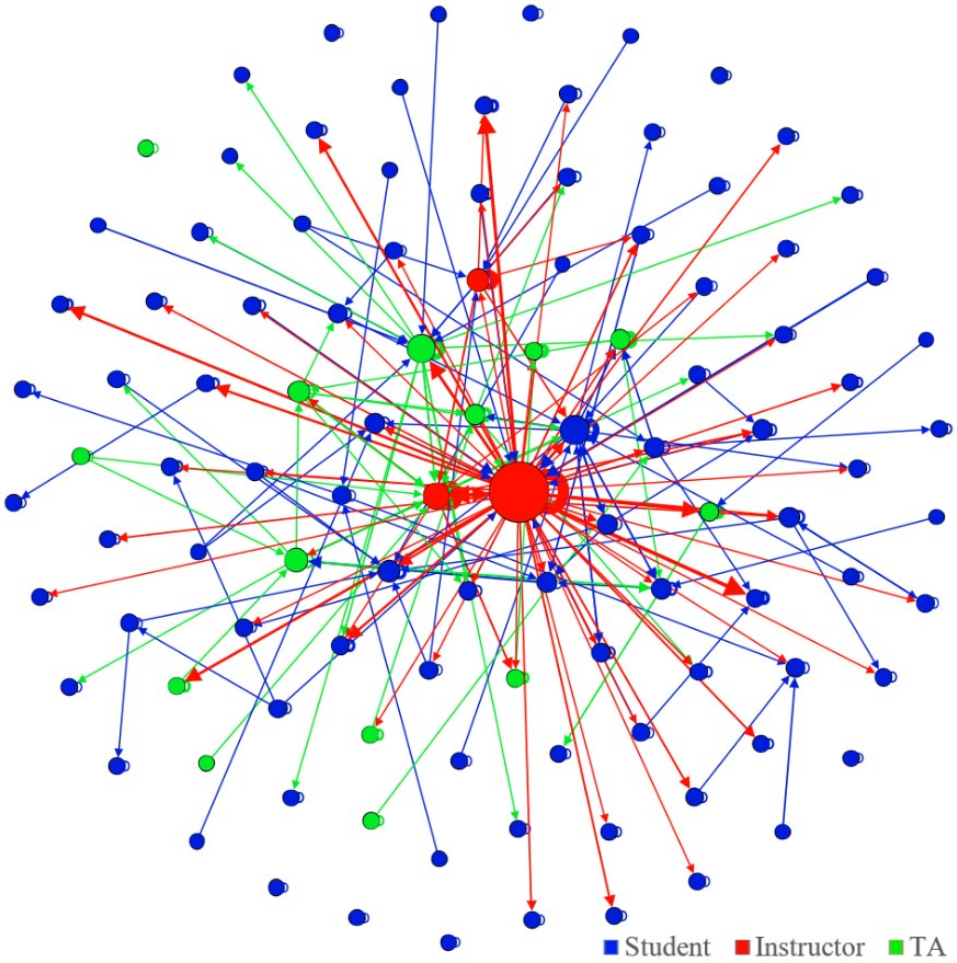

Figure 5.2. Study I Experimental treatment social network graph

5.2.7 System Usability

System usability was measured via the System Usability Scale. Comparisons detected significant differences ($U = 3194$, $p = .014$) between Control ($N = 104$) and Experimental ($N = 78$) with the Experimental treatment having a 23% higher mean ranking (103 versus 83) and small effect size ($r = .132$). The Control group's mean SUS rating was about 65. In contrast, the Experimental group mean SUS rating was about 60. Though the Experimental group provided a lower system usability rating on average, the analysis indicated the Experimental group more often rated the system higher. Nonetheless, both ratings indicate a below average usability score

www.manaraa.com

(anything above 68 indicates an above average score), which suggests users may have encountered usability issues while using the OSBLE+ system.

5.2.8 Experimental Treatment Intervention Analysis

The previous sections looked at the differences between Control and Experimental treatment groups. Focusing on the Experimental treatment, this section considers if there were any significant effects detected based on students' actual use of the interventions.

There was a total of 35,658 interventions generated, with an average of 262 interventions generated per student (see Table 5.13).

Table 5.13. Study I interventions generated per student

Interventions Generated	
Min	1
Max	1,693
Mean	262
Std. Dev.	343
Total	**35,658**

5.2.8.1 Intervention Generation and Interaction Levels

To determine whether a relationship existed between a student's course performance and the number of interventions generated for the student, I used a Kendall's tau-b correlation with a Bonferroni corrected $\alpha <= .007$ (.05/7), 1-tailed.

Results indicated that there was a statistically significant weak to moderate correlation between all course performance metrics and how many interventions were generated (Table 5.14). Additionally, there were weak correlations between final course grade, programming assignments, and midterm performance with intervention interactions (Table 5.14).

www.manaraa.com

Table 5.14. Study I correlations between performance and intervention generation and interactions

	Interventions Generated		Intervention Interactions	
Performance Metric	τ_b	p	τ_b	p
Final Course Grade	.414		.192	**.002**
Lab Participation	.285		.141	.021
Quizzes	.343		.161	.008
Programming Assignments	.441	**< .001**	.187	**.002**
Written Midterms	.329		.181	**.003**
Lab Final Exam	.319		.115	.049
Written Final Exam	.314		.134	.023

*N = 141; 1-tailed; *significant at adjusted α <=.007*

5.2.8.2 Individual Intervention Interactions

To get an idea of which interventions were interacted with, I categorized interaction log data by individual intervention. These counts include logged interactions with the specific intervention windows and exclude general system interactions (Table 5.15). They do not indicate whether students followed through with intervention suggestions. Intervention interactions ranged from 0 to 2 unique interactions. The maximum number of interactions by a single participant was 2, with a total of 22 interactions split between 15 of the 141 participants.

Table 5.15. Study I Experimental treatment intervention interaction counts by intervention

Intervention \ Interaction Count	Sum	Max[1]	Unique	Generated[2]	Follow Up
Runtime Errors: Get Help!	0	0	0	28,744	0
Build Errors: Get Help!	4	2	3	3,674	0
Others Available to Help!	1	1	1	841	1
Help other Students!	7	2	6	969	0
Help Your Classmates!	0	0	0	171	0
You Are Available!	8	2	7	0	0
Make a Post! (Topical)	2	1	2	631	0
Make a Post! (Assignment)	0	0	0	628	1

[1]interactions by single student, [2]total for entire treatment

www.manaraa.com

5.3 Discussion

Based on the study results, it is not possible to answer either of the main research questions or support any of the posed hypotheses. This is likely explained by two factors: (1) significantly lower self-sociability seen in the Experimental treatment group (as seen in 5.2.5) and (2) extremely low adoption and interactions with the intervention tools (as indicated by the nonexistent interaction levels outlined in the previous section). Outside or pre-existing influences may also have affected the usage of the interventions. I consider this possibility further in Section 11.2.2 on limitations.

There does seem to be a positive correlation between interventions generated and performance metrics, indicating a possible correlation between student activity level and course performance. This relationship falls into line with our expectations from the learning theories outlined in Chapter 2. That is, we would expect students who are more active in the course (indicated by the number of interventions generated) to perform better in the course. In general, though, the Experimental group was just less social within the system and rarely interacted with the interventions. In fact, the most frequent level of interaction with the system was zero interactions as illustrated by Table 5.15 and Table 5.16.

Table 5.16. Study I intervention interactions summary

	Max	Mean	Std dev	Mode*
Interventions Generated	1693	262	343	0
Intervention Interactions (click-through)	3	< 1	< 1	0
Interventions / Total System Events	18.46%	3.09%	2.82%	0%
Interactions / Total System Events	< 1%	< 1%	< 1%	0%
Interactions / Intervention Generated	7.14%	< 1%	< 1%	0%

www.manaraa.com

CHAPTER 6

STUDY II

As seen in Table 5.15, student interaction with interventions in Study I was nearly nonexistent. Based on the surprisingly low level of interaction with the interventions and a low quantity of social interaction on the OSBLE+ activity feed throughout the Study I semesters, I decided to run a follow-up study (Study II) in which students were *required* to participate in the activity feed. In addition, in this follow-up study, I switched to a within-subjects design, so that data on both treatments could be collected within a single semester.

The same research questions and hypotheses used in Study I (Chapter 5) were used for this study Additionally, many of the measures remained the same between studies; differences will be identified in the methods section below.

6.1 Methods

Study II took place in the summer, 2017 offering of CptS 121, the same CS 1 course considered in Study I. In contrast to Study I, Study II used a within-subjects design. All students were exposed to both treatments, with each treatment corresponding to a different four-week period within the course, which was delivered in a condensed eight-week semester. The first four weeks of the semester (including half of the assignments and the midterm exam) served as the Control treatment (interventions were turned off in the SPE). The second four weeks of the semester corresponded to the Experimental treatment; the SPE interventions were turned on during this period.

The course involved in Study II had a different instructor (me) from Study I. It covered the same topics using similar materials. Only minor changes were made to the assignments to incorporate different tasks and scenarios.

www.manaraa.com

6.1.1 Participants

All 26 students enrolled in the summer 2017 offering of CptS 121 gave consent to participate in the study. The 26 participants consisted of 20 male and 6 female students. Ages ranged from 18 to 30 (M=22), with 54% of the students majoring in computer science. To stimulate participation, I offered course extra credit instead of a drawing for a gift card.

6.1.2 Materials

The same materials used in Study I were used for Study II (see section 5.1.2).

6.1.3 Log Data

All log data was collected in the same manner in Study II as was done in Study I (see Section 5.1.4).

6.1.4 Independent Variable

The independent variable remained the same as described in Study I section 5.1.3.

6.1.5 Dependent Variables

All dependent variables seen in Study I section 5.1.4 remained the same in Study II, with two exceptions due to the different study design.

Unlike Study I, there were not paired Control-Experimental assignments that could be analyzed; thus, the assignments, quizzes, and exams used in the two treatments varied in topic and complexity. I created a change over time value for each assignment in each treatment period to compare activity levels within each assignment period. For example, a total of zero events in one assignment and a total of one event in the following assignment would equate to a 100% increase in activity. This was calculated for the transition between each assignment.

www.manaraa.com

In addition, there was no way to compare assignments and final course grades directly. Instead, I compared, course performance by treatment instead of by overall course grade. Course performance for each treatment consisted of students' average performance for the four programming assignments, three quizzes, and one written exam.

Assignments, quizzes, and exams varied in topic and complexity, with assignments and exams building on all prior materials. In contrast, quizzes focused on only the prior week's topics.

6.1.6 Procedure

As with Study I (6.1.6), participants were given pre-, mid- and post-surveys during the first week, after the fourth week and during the final week of the semester, respectively. Analysis was performed in the same manner as in Study I, with some exceptions that will be highlighted in the presentation of results.

In order to stimulate social activity on the course website, the course instructor established a post/reply requirement of two posts and replies per week. Students only needed to make a post or reply and were not required to meet a quality requirement. Any posts or replies that did not contain academic or social context and, therefore, did contribute meaningfully to the activity feed (e.g., "Post 1: Obligatory post") were removed from the analysis. The motivation for this requirement was to stimulate some minimal level of interaction on the activity feed to help overcome any initial hurdle to participating in the feed, and is consistent with prior studies of the foundational SPE OSBIDE (Carter, 2016), in which it was found that there was no negative impact on results when a posting requirement was established; social activity remained positively correlated both with and without posting requirements.

www.manaraa.com

6.2 Results

Before presenting the results, I note that I used the same analysis approach as in Study I. For example, I once again used Bonferroni correction to guard against Type I error. There was one key difference, however: in statistical comparisons, Wilcoxon signed-rank tests were used instead of the Mann-Whitney U tests because this study used a within-subjects design, and hence had matched samples.

Table 6.1 presents summary log data for each treatment of Study II. In the following subsections, I use these data as a basis for analyzing treatment differences with respect to each dependent variable.

Table 6.1. OSBLE+ event log data for Study II

		Summer 2017 (Control)	*Students: 26*	*Summer 2017 (Experimental) Students: 26*	
Source	**Event**	**Event Counts**	**Counts/ Student**	**Event Counts**	**Counts/ Student**
IDE	Ask-For-Help	26	1	21	1
	Build	7,830	301	17,416	670
	Cut-Copy-Paste	11,741	452	22,281	857
	Debug	9,159	352	56,796	2,184
	Editor	20,822	801	36,066	1,387
	Build Error	3,777	145	8,654	333
	Build Error Counts	16,851	648	66,314	2,551
	Exception	574	22	2,962	114
	Save	11,540	444	26,267	1,010
	Submit	407	16	855	33
Activity Feed	Helpful Mark	37	1	28	1
	OSBLE Interaction	2,686	103	4,804	185
	Post	219	8	518	20
	Reply	237	9	564	22
***Total Event Counts:**		**65,278**	**2,510**	**168,578**	**6,484**

**Build errors and counts are excluded from the total as each build error event is also a build event and each build error event can have many build errors*

www.manaraa.com

6.2.1 *Participation*

Table 6.2 presents a statistical comparison of the two treatments with respect to the log data relevant to participation. As can be seen, there were significant differences between the two treatments for all three measures, with the Experimental treatment engaging in a higher level of participation. The effect size was high. These results provide empirical support for **H1**.

Table 6.2. Study II participation comparisons

Social Metric	*Z*	*p*<=	*r*	*M* Ranks**	*Mdn***
Aggregate Feed Activity	-4.433	**.001**	.869	14:1	35:16
OSBLE Interactions	-4.254	**.001**	.834	14:4	121:51
Total Events	-4.330	**.001**	.849	14:3	6893:2009

*N = 26; **All ratios are Experimental:Control, *significant at adjusted α <=.0167*

Additional tests were performed to see if differences existed with respect to changes in activity level over time (see Section 6.1.5) for each of the three measures. Results indicated that the changes in activity level over time were not significantly different between the treatment periods (Table 6.3).

Table 6.3. Study II participation change over time comparisons

Social Metric	*Z*	*p*<=	*r*	*M* Ranks**	*Mdn***
Aggregate Feed Activity	-0.211	0.833	0.041	13:10	.025:0
OSBLE Interactions	-0.978	0.328	0.192	11:15	.32:.55
Total Events	-0.013	0.990	0.003	13:15	.76:.52

*N = 26; **All ratios are Experimental:Control, *significant at adjusted α <=.0167*

6.2.2 *Programming Activity*

Table 6.4 presents statistical comparisons of the two treatments with respect to each measure relevant to programming activity. As the table indicates, only build errors for the Experimental treatment ($Mdn_e = 440$) more frequently occurred than build errors for the Control treatment ($Mdn_c = 104$), $Z = -4.356$, $p = .001$.

www.manaraa.com

Table 6.4. Study II Statistical comparison of treatments with respect to programming activity measures

Programming Metric	*Z*	*p*<=	*r*	*M* Ranks**	*Mdn***
Build	-1.295	.195	.254	14:13	590:737
Cut-Copy-Paste	-2.197	.028	.431	10:15	866:906
Debug	-.807	.420	.158	13:13	1651:907
Editor	-2.121	.034	.416	12:14	1723:1658
Build Error	-4.356	**.001**	.854	14:4	440:104
Exception	-1.278	.201	.251	16:10	88:73
Save	-1.486	.137	.291	15:13	728:902
Submit	-2.356	.018	.462	13:13	590:737

*N = 26; **All ratios are Experimental:Control, *significant at adjusted α <=.006*

Analyzing the change over time for each programming activity event indicated significant differences only for exception events with higher mean ranks in the Experimental treatments. This suggests that the Experimental treatment encountered significantly more errors over time (Table 6.5).

Table 6.5. Study II active programming assignment programming activity change over time comparison

Programming Metric	*Z*	*p*<=	*r*	*M* Ranks**	*Mdn***
Build	-.632	.527	.124	13:13	.74:.85
Cut-Copy-Paste	-1.232	.218	.242	13:14	1.01:1.08
Debug	-.821	.412	.161	13:13	1.28:1.30
Editor	-.571	.568	.112	13:14	.44:.50
Build Error	-.029	.997	.006	14:11	1.14:.60
Exception	-3.057	**.002**	.600	14:12	2.33:.00
Save	-.876	.381	.172	12:15	.62:.66
Submit	-2.095	.036	.411	12:13	.61:.27

*N = 26; **All ratios are Experimental:Control, *significant at adjusted α <=.006*

Comparing time in error state between the treatments indicated that there were no significant differences (Mdn_e = 98, Mdn_c = 88, Z = -.605, p = .545). Comparing changes over time in error state yielded a similar conclusion (Mdn_e = 1.44, Mdn_c = .30, Z = -.629, p = .530).

As in Study I, these results indicate that programming activity levels between treatments were similar overall. They do not provide empirical support for **H2**.

www.manaraa.com

6.2.3 *Performance – Achievement*

A comparison of both programming assignment performance and quiz performance between treatments showed significant differences (see Table 6.6). In the case of programming assignments, the Control treatment had a higher mean rank, while for quiz performance, the Experimental treatment had a higher mean rank.

Table 6.6. Study II performance comparisons

Performance Metric	*Z*	*p*<=	*r*	*M* Ranks**	*Mdn***
Programming Assignments	-2.603	**.009**	.510	10:15	.84:.93
Quizzes	-3.505	**.000**	.687	16:6	.73:.58
Lab Participation	-.350	.727	.069	7:9	1.00:.91
Written Exams	-1.334	.182	.262	11:15	.72:.79

*N = 26; **All ratios are Experimental:Control, *significant at adjusted α <=.013*

I additionally compared changes over time for programming assignments and quizzes. Results indicated both performance metrics had a higher mean rank in the Experimental treatment but only quiz performance change over time was determined to be statistically significant (Table 6.7). This is also reflected in the average change over time between treatments (Table 6.8).

Table 6.7. Study II performance change over time comparisons

Performance Metric	*Z*	*p*<=	*r*	*M* Ranks**	*Mdn***
Programming Assignments	-1.968	.049	.368	18:8	.115:.003
Quizzes	-3.873	**.001**	.760	16:5	.476:.001

*N = 26; **All ratios are Experimental:Control, *significant at adjusted α <=.025*

Table 6.8. Study II Control and Experimental average performance change over time by treatment

Treatment	Programming Assignments	Quizzes
Control	-6%	-19%
Experimental	1%	18%

These results suggest that on the quiz performance measures the Experimental treatment significantly outperformed the Control treatment, thus providing empirical support for **H3**.

www.manaraa.com

6.2.4 *Performance – Activity Levels*

To determine if there was a relationship between student activity levels and course performance, I applied one-tailed Kendall's tau-b correlation (with corrected $\alpha = .008$ to account for six comparisons) in the same manner as in Study I. For this measure, analysis of performance was again broken up into eight performance measures; 4 programming assignments, 3 quizzes, and a written exam.

Results indicated no significant correlations in the Control treatment (Table 6.9). In the Experimental treatment, in contrast, results indicated significant weak to moderate correlations between social activity and programming assignments and between OSBLE interactions and both programming assignments and quizzes (Table 6.10). These results suggest that increased social activity was positively correlated with increased performance in the Experimental treatment, thus providing some support for **H4**.

Table 6.9. Study II correlations between course performance and social activity level (Control treatment)

Performance Metric	Aggregate Social Activity		OSBLE Interactions		Total Events	
	r_τ	$p <$	r_τ	$p <$	r_τ	$p <$
Programming Assignments	.288	.021	.287	.020	.292	.018
Quizzes	.330	.010	.322	.011	.371	.004
Written Exam	.126	.188	.087	.268	.167	.117

1-tailed - significant at adjusted α <=.008

Table 6.10. Study II Correlations between course performance and social activity level (Experimental treatment)

Performance Metric	Aggregate Social Activity		OSBLE Interactions		Total Events	
	r_τ	$p <$	r_τ	$p <$	r_τ	$p <$
Programming Assignments	.353	**.005**	.433	**.001**	.462	.001
Quizzes	.251	.037	.355	**.005**	.366	.004
Written Exam	.295	.018	.294	.018	.398	.002

1-tailed - significant at adjusted α <=.008

www.manaraa.com

6.2.5 Student Attitudes

Table 6.11 presents a summary of statistical comparisons between the Control and Experimental treatments with respect to the student attitudinal measures. Unfortunately, due to an oversight, surveys were not administered for all measures at the midpoint of the semester. Since attitudinal surveys were administered only at the start and end of the semester, I could only determine attitudinal shifts over the course of both treatments; attitudinal shifts that may have occurred within the individual treatments were not measured.

As Table 6.11 suggests, students experienced a significant increase in coding self-efficacy from the start to the finish of the semester. No other statistically significant differences in attitudes were detected. Given that no attitudinal survey was administered halfway through the semester, however, Study II is unable to provide any evidence to support or refute the hypotheses regarding student attitudes (**H5-H8**).

Table 6.11. Study II attitudinal comparisons

Attitudinal Metric	*U*	*p*<=	r	*M* Ranks**	*Mdn***
Coding Self-Efficacy	-2.815	**.005***	.552	16:6	180:150
CCS: Connectedness	-.174	.862	.0341	11:11	25:24
CCS: Learning	-1.905	.057	.374	13:11	25:23
CCS: Total	-1.087	.277	.213	12:14	49:47
MSLQ: Self-Learning	-1.257	.209	.247	13:12	42:44
MSLQ Peer-Learning	-1.122	.262	.220	11:9	5:6
System-Sociability	-.731	.465	.143	11:11	3.8:3.5

N = 26; **All ratios are Experimental:Control, *significant at adjusted α <=.05
†significant at adjusted α <=.017; ‡significant at adjusted α <=.025

Viewing changes between pre and post course for intent to major showed that one CS major (out of 14) changed their major at the end of the course. In contrast, one non-CS major (out of 12) changed their response to indicate they were a CS major at the end of the course resulting in no net change in intent to major.

www.manaraa.com

6.2.6 Social Network

Table 6.12 presents the results of the social network measures for the Control and Experimental treatments. These results show that the Experimental treatment did reach significantly higher values for all social network metrics except betweenness and average path length.

Additionally, degree, weighted degree, and closeness were found to be significantly different, with the Experimental treatment values having a higher mean rank (Table 6.12). In the case of average path length, a lower score is better as it indicates closer connections between students. Based on this, we can conclude that these results support **H10**; the Experimental treatment network is more connected than the Control treatment network.

Table 6.12. Study II social network statistics

Descriptives	Control Treatment	Experimental Treatment	*Z*	*p*<=	*r*	*M* Ranks**	*Mdn***
Number of Messages	724	1534					
Degree	5.72	7.64	-3.263	**.001**	.617	15:7	14.00:10.50
Weighted Degree	24.97	54.79	-4.600	**.001**	.869	15:1	69.00:35.00
Closeness	.47	.53	-2.931	**.003**	.554	15:11	.56:.50
Betweenness	31.32	18.82	-2.138	.032	.404	11:15	8.12:11.07
Eigenvector	.45	.50	-1.153	.249	.218	16:12	.49:.44
Density	.20	.28					
Diameter	4	4					
Average Path Length	2.16	1.83					

**significant at adjusted α <=.025; **All ratios are Experimental : Control*

Figure 6.1 and Figure 6.2 show a graphical representation of each treatment's social networks. In each of these figures we can see that the most important actor in the graph is the instructor (red) with minimal interaction from the TA (green) and mixed interactions from students (blue). In each, node and edge sizes are scaled to degree of connection which allows us to see that, overall, the Experimental treatment students formed more connections.

www.manaraa.com

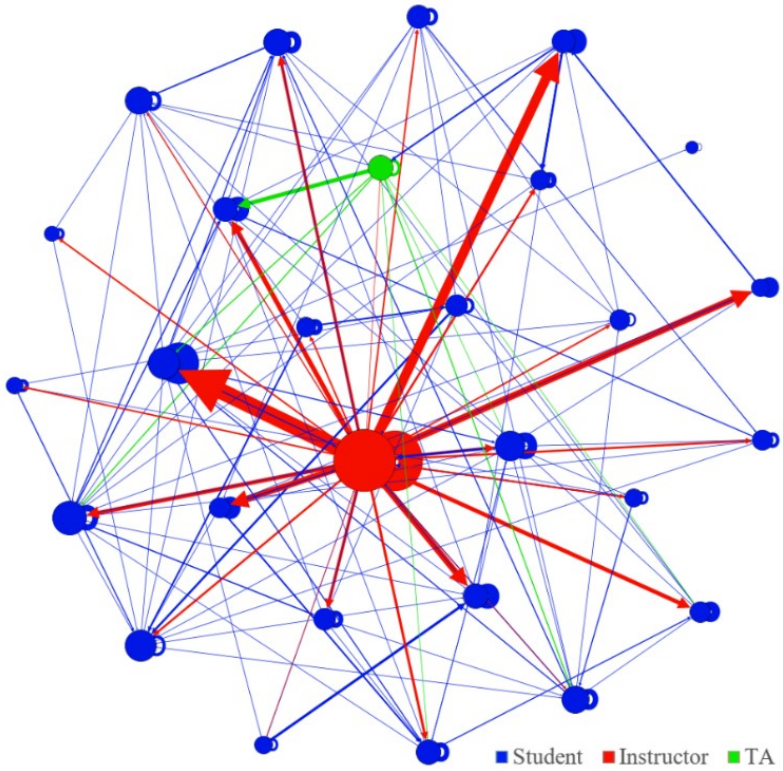

Figure 6.1. Study II social network graph for Control treatment

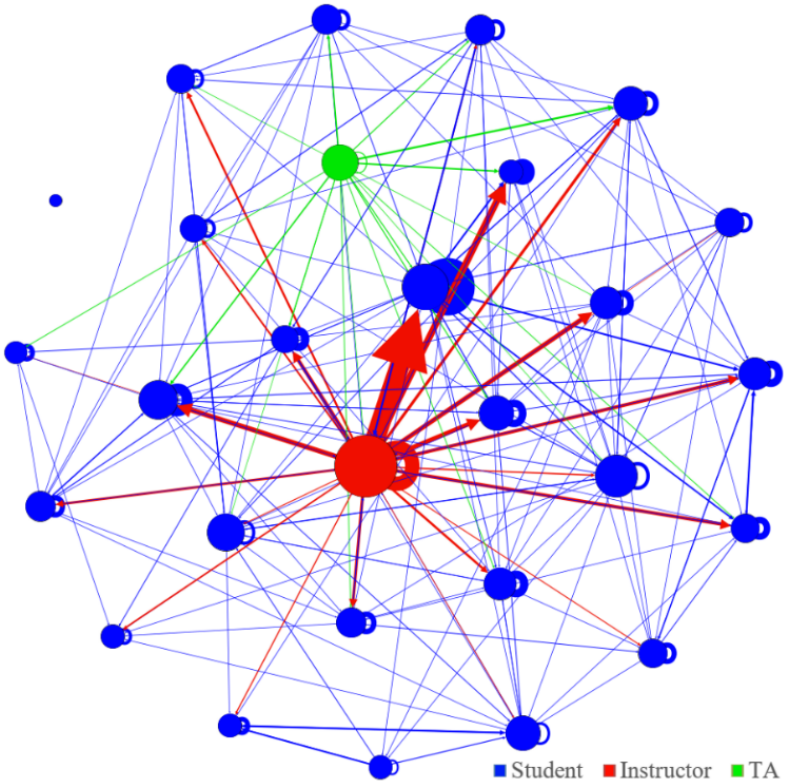

Figure 6.2. Study II social network graph for Experimental treatment

www.manaraa.com

6.2.7 System Usability

Because no surveys were administered at the midpoint of the Study II semester, I was unable to compare the System Usability Scores (SUS) of the Control and Experimental treatment. However, I can report that the average SUS score for all students was just over 68, which is higher than either of the Study I average SUS scores (65 and 60 respectively). This result shows the Study II participants had an above average perceived system usability rating, as opposed to the below average perceived ratings from Study I, thus suggesting that it was less likely users may have encountered usability issues while using the OSBLE+ system in Study II.

6.2.8 Experimental Treatment Intervention Analysis

The previous sections looked at the differences between Control and Experimental treatments. This section looks at the Study II Experimental treatment (2nd half of the semester) to determine if there were any significant effects detected from intervention use.

6.2.8.1 Intervention Generation and Interaction Levels

As in Study I, a 1-tailed Kendall's tau-b correlation was performed with a Bonferroni corrected $\alpha <= .006$ (.05/8). There were 8,488 total interventions generated, with an average of 326 interventions generated per student (see Table 6.13).

Table 6.13. Study II Interventions generated summary

	Interventions Generated
Min	37
Max	1153
Mean	326
Std. Dev.	287
Total	**8,488**

www.manaraa.com

Results indicate a statistically significant weak to moderate correlation between a student's final course grade and the number of interventions generated for that student. Likewise, there was a statistically significant weak to moderate correlation between students' lab participation and their interactions with the interventions, as well as between students' quiz scores and their interactions with the interventions (Table 6.14).

Table 6.14. Study II Correlations between performance and intervention generation and interactions

	Interventions Generated		Intervention Interactions	
Performance Metric	τ_b	***p***	τ_b	***p***
Final Course Grade	.354	**.006**	.306	.016
Lab Participation	.114	.220	.378	**.006**
Quizzes	.348	**.006**	.363	**.005**
Programming Assignments	.348	**.006**	.331	.010
Lab Final Exam	.229	.051	.273	.028
Written Final Exam	.207	.070	.212	.069

*N = 26; *significant at adjusted α <=.006*

6.2.8.2 Individual Intervention Interactions

Table 6.15 presents counts of interactions with each category of intervention. As this table shows, intervention interactions ranged from 0 to 24 unique interactions per intervention (because there were 24 different students in the study who interacted with the interventions), with only the runtime errors intervention not receiving any logged interactions this time. The maximum number of interactions by a single participant was 38, with a total of 353 interactions split between 24 of the 26 participants.

www.manaraa.com

Table 6.15. Study II intervention interaction, generation, and follow up counts by intervention type

Intervention \ Interaction Count	Sum	Max[1]	Unique	Generated[2]	Follow up
Runtime Errors: Get Help!	0	0	0	793	0
Build Errors: Get Help!	1	1	1	6494	0
Others Available to Help!	109	18	24	94	62
Help other Students!	18	3	12	103	4
Help Your Classmates!	2	1	2	128	1
You Are Available!	188	22	24	19	104
Make a Post! (Topical)	10	3	6	95	9
Make a Post! (Assignment)	25	3	13	762	22

[1]interactions by single student, [2]total for entire treatment

Additional analysis of interactions showed that 22 of 26 participants engaged in follow-up social actions (OSBLE Interactions, posts, replies, or helpful marks on replies) within 10 minutes of the intervention interaction. Of the 8 interventions, 6 had follow up interactions; only the runtime and build error interventions did not elicit any follow-up activity. Participants made up to 12 follow-up actions within a 10-minute period of intervention interaction (Table 6.16). An activity feed post was the most common first follow-up action [164 out of the 202 (81%) of first actions following intervention interaction], with an interaction with the OSBLE activity feed being the 2nd most common first action [38 of the 202 (19%) of first actions]. Table 6.16 indicates that the most frequent action taking place after intervention interactions was a feed post. This was followed by interactions with the activity feed and replies to feed posts.

Looking at follow-up activity by intervention shows that two of the availability status related interventions were the most frequently followed up on ("*Others Available to Help!*" Figure 4.14, "*You Are Available!*" Figure 4.19). The assignment submission intervention was the third most followed up intervention ("*Make a Post! (assignment submission)*" Figure 4.21)

www.manaraa.com

Table 6.16. Study II follow up action count and order

Follow Up Action Rank	Post	Reply	OSBLE Interaction*
1st	164	0	38
2nd	16	6	79
3rd	7	15	43
4th	0	24	26
5th	3	3	33
6th	0	1	32
7th	0	6	3
8th	0	0	5
9th	0	1	3
10th	2	0	1
11th	0	0	1
12th	0	0	1

**Counts of follow up interactions on the OSBLE activity feed: posts, replies, replies marked helpful*

6.2.9 System Activity Levels

Looking at the activity levels with the system, we see that the Experimental treatment was more active than the Control treatment across the board, with the Experimental treatment having more than double the number of activity events per student for most events (Table 6.17).

Table 6.17. Study II Experimental activity counts percent difference

Ask-For-Help	Build	Cut-Copy-Paste	Debug	Editor	Build Error	Build Error Counts
0%	123%	90%	520%	73%	130%	294%
Exception	**Save**	**Submit**	**Helpful Mark**	**OSBLE Interaction**	**Post**	**Reply**
418%	127%	106%	0%	80%	150%	144%
Total Event Counts: 158%						

**percent difference – positive indicates Experimental counts higher than Control counts; **per student counts* Table 6.1

6.3 Discussion

Unlike in Study I, in Study II we see several results that support the hypotheses:

www.manaraa.com

- Overall activity levels, along with quiz performance, were significantly higher in the Experimental treatment, providing some support for **H1** and **H3**.
- There were positive correlations between social activity and performance in the Experimental treatment, providing support for **H4**.
- There was a more connected social network in the Experimental treatment, providing support for **H10**.
- The analysis of interventions generated and interacted with identified positive correlations between interventions generated and overall course performance, and interactions interacted with and quiz and lab performance.

This analysis gives one reason to believe that exposure to interventions may positively affect student performance and participation. Participation levels, social network coupling, and performance all showed significantly higher results for the Experimental treatment. In some cases, though, the significantly increased activity was not associated with success. However, this might be explained by the increasing difficulty of tasks in the Experimental treatment, which, due to the within-subjects design of the study, occurred in the second half of the semester, when more difficult course concepts (e.g., arrays and dynamic memory) were addressed.

At the same time, we saw a significant increase in quiz scores in the Experimental treatment. One explanation is that course quizzes focused on only the previously discussed topic and were short (15-20 minutes) in duration. Therefore, more recent learning experiences had more effect on performance and activity levels.

www.manaraa.com

Further, we need to consider that the more closely-coupled social network could have been a result of maturation; students may have been more likely to form closer connections naturally as they spent more time together through the semester.

With respect to **RQ1** (Can interventions promote positive changes in performance and activity?), we might infer from these results that the availability status (help-seeking: "*Others Available to Help!*" and help-giving: "*You Are Available!*") interventions were the most influential, as they had the most interactions and follow up activity. At the same time, the study's mixed results make it difficult to pinpoint the degree to which exposure to the interventions had an influence on student learning and attitudes. To gain more clarity, the study presented in the next chapter takes another shot at evaluating the effectiveness of the interventions with a different set of participants.

www.manaraa.com

CHAPTER 7

STUDY III

Motivated by the more positive results of Study II with respect to this dissertation's hypotheses, I ran a third study (Study III) to collect additional evaluation data. Study III was designed to replicate Study II, while remedying its data collection mistakes by collecting comparative data on student attitudes.

7.1 Methods

Run in conjunction with the summer 2018 offering of CptS 121 at Washington State University, Study III used mostly the same methods as Study II. Any differences are noted below.

7.1.1 Participants

The summer 2018 semester of CptS 121 started with 19 students, of whom 17 were male and 2 were female. Participants' ages ranged from 20 to 28 (M=24) with 42% (eight of the 19) of the participants majoring in computer science. All 19 students gave consent for their data to be used for this study. However, only 15 students completed all surveys: 13 male, 2 females; similar age range and mean; and 33% majoring in computer science. Partial participant data was used from students who did not complete the full survey and where overlapping analysis was required; e.g., social network data for the Control treatment period or pre-post comparisons. All other partial data was excluded as there was no Experimental treatment data for comparison. As in Study II, course extra credit was offered in order to incentivize participation.

7.1.2 Materials

The same materials used in Study I were used for Study III (see section 5.1.2).

www.manaraa.com

7.1.3 Log Data

The same materials used in Study I were used in Study III (see section 5.1.2). Additionally, all course materials (assignments, quizzes, exams, etc.) used in Study II were the same in Study III.

7.1.4 Independent Variable

The independent variable remained the same as described in Study I (see section 5.1.3).

7.1.5 Dependent Variables

All dependent variables seen in Study I and II, section 5.1.4, remained the same in Study III.

7.1.6 Procedure

The Study III procedure was identical to that of Study II (see Section 6.1.6).

7.2 Results

As in Study II, Wilcoxon signed-rank tests were used instead of Mann-Whitney U tests for statistical comparisons. This was because of the within-subjects design, which included matched samples. Table 7.1 presents a summary of the log data collected in the Control and Experimental treatments. Below, I present a statistical analysis of this data for each dependent measure.

www.manaraa.com

Table 7.1. OSBLE+ event log data for Study III

Control treatment			*Students: 19*	*Experimental treatment*	*Students: 15*
Source	**Event**	**Event Counts**	**Counts/ Student**	**Event Counts**	**Counts/ Student**
IDE	Ask-For-Help	25	1	19	1
	Build	8,697	458	13,566	904
	Cut-Copy-Paste	13,890	731	25,902	1,727
	Debug	10,494	552	91,935	6,129
	Editor	20,532	1,081	29,399	1,960
	Build Error	3,840	202	4,353	290
	Build Error Counts	16,948	892	25,857	1,724
	Exception	330	17	1,854	124
	Save	13,291	700	23,507	1,567
	Submit	341	18	540	36
Activity Feed	Helpful Mark	21	1	10	1
	OSBLE Interaction	1,620	85	2,617	174
	Post	109	6	224	15
	Reply	129	7	417	28
***Total Event Counts:**		**69,479**	**4,751**	**189,990**	**12,666**

**Build errors and counts are excluded from the total as each build error event is also a build event and each build error event can have many build errors*

7.2.1 Participation

Table 7.2 presents a statistical comparison of the two treatments with respect to the log data relevant to participation. As can be seen, there were significant differences between the two treatments for all three measures, with the Experimental treatment engaging in a higher level of participation. The effect size was high. These results provide empirical support for **H1**.

Table 7.2. Study III participation comparisons

Social Metric	***Z***	***p*<=**	***r***	***M* Ranks****	***Mdn*****
Aggregate Feed Activity	-3.408	**.001**	.782	8:7	29:16
OSBLE Interactions	-3.010	**.003**	.691	9:4	81:48
Total Events	-3.011	**.003**	.691	8:0	11429:3166

*N = 15; **All ratios are Experimental:Control, *significant at adjusted α <=.0167*

www.manaraa.com

Additional comparisons were performed for activity only during active programming assignment periods using the same method used in Study II (Section 6.2.1). Analysis showed similar results (Table 7.3), except for the OSBLE interactions, which were determined not to be significantly different between treatments. Results indicated that the changes in activity level over time were not significantly different between the treatments.

Table 7.3. Study III active programming assignment participation comparison

Social Metric	*Z*	*p<=*	*r*	*M Ranks***	*Mdn***
Aggregate Feed Activity	-2.480	**.013**	.569	8:4	21:11
OSBLE Interactions	-2.166	.030	.497	9:5	53:45
Total Events	-3.408	**.001**	.782	8:0	10814:5365

*N = 15; **All ratios are Experimental:Control, *significant at adjusted α <=.0167*

Table 7.4. Study III participation change over time comparisons

Social Metric	*Z*	*p*	*r*	*M Ranks***	*Mdn***
Aggregate Feed Activity	-1.506	.132	.346	8:8	.90:.33
OSBLE Interactions	-.170	.865	.039	6:11	.79:.88
Total Events	-.398	.691	.091	9:7	.72:.88

*N = 15; **All ratios are Experimental:Control, *significant at adjusted α <=.0167*

7.2.2 Programming Activity

Table 7.5 presents statistical comparisons of the two treatments with respect to each measure relevant to programming activity. As the table indicates, all metrics except build errors occurred significantly more frequently for the Experimental treatment than events for the Control.

Analyzing the change over time for each programming activity event indicated significant differences only for exception events with higher mean ranks in the Experimental treatment (Table 7.6).

www.manaraa.com

Table 7.5. Programming activity metrics Control-Experimental comparison

Performance Metric	*Z*	*p*	*r*	*M Ranks***	*Mdn***
Build	-3.294	**.001**	.756	8:2	836:456
Cut-Copy-Paste	-3.294	**.001**	.756	8:2	1396:737
Debug	-3.408	**.001**	.782	8:0	4361:843
Editor	-3.294	**.001**	.756	8:2	1771:1305
Build Error	-1.022	.307	.234	8:8	201:202
Exception	-3.409	**.001**	.782	8:0	98:27
Save	-3.124	**.002**	.717	9:3	975:580
Submit	-2.898	**.004**	.665	9:5	39:26

*N = 15; **All ratios are Experimental:Control, *significant at adjusted α <=.006*

Table 7.6. Study III active programming assignment programming activity change over time comparison

Performance Metric	*Z*	*p*	*r*	*M Ranks***	*Mdn***
Build	-1.363	.173	.313	9:8	.52:1.29
Cut-Copy-Paste	-1.931	.053	.443	9:8	.51:1.76
Debug	-.114	.910	.026	10:7	2.27:1.72
Editor	-1.420	.156	.326	9:8	.24:1.08
Build Error	-.284	.776	.065	8:8	1.46:.63
Exception	-3.233	**.001**	.742	8:1	3.31:.00
Save	-1.250	.211	.287	8:8	.58:1.27
Submit	-.284	.776	.065	8:8	.61:.33

*N = 15; **All ratios are Experimental:Control, *significant at adjusted α <=.006*

Comparing time in error state between the treatments indicated that there were no significant differences with the adjusted *alpha* of .006 (Mdn_e = 132, Mdn_c = 100, Z = -2.669, p = .008). Comparing changes over time in error state yielded a similar conclusion (Mdn_e = 1.55, Mdn_c = .42, Z = -.454, p = .650).

As in Study I and II, these results indicate that programming activity levels between the treatments were similar overall. They do not provide empirical support for **H2**.

7.2.3 Performance – Achievement

A comparison of both programming assignment performance and lab performance between treatments showed significant differences (see Table 7.7). In the case of programming

www.manaraa.com

assignments, the Control treatment had a higher mean rank while for lab performance, the Experimental treatment had a higher mean rank.

Table 7.7. Study III performance comparisons

Performance Metric	*Z*	*p*<=	*r*	*M* Ranks**	*Mdn***
Programming Assignments	-2.613	**.009**	.599	4:10	.90:.95
Quizzes	-1.648	.099	.378	7:10	.87:.80
Lab Participation	-3.392	**.001**	.778	9:2	1.00:.89
Written Exams	-.54	.589	.124	6:10	.81:.81

*N = 15; **All ratios are Experimental:Control, *significant at adjusted α <=.013*

I additionally compared changes over time for programming assignments and quizzes. Results indicated change over time was statistically significant, with higher mean ranks in the Experimental treatment (Table 7.8). This is also reflected in the average change over time between treatments (Table 7.9).

Table 7.8. Study III performance change over time comparisons

Performance Metric	*Z*	*p*<=	*r*	*M* Ranks**	*Mdn***
Programming Assignments	-.738	.460	.169	8:8	.010:-.012
Quizzes	-2.329	**.020**	.534	9:5	.100:-.088

*N = 26; **All ratios are Experimental:Control, *significant at adjusted α <=.025*

Table 7.9. Study III Control and Experimental average performance change over time by treatment

Performance Metric	Control	Experimental
Programming Assignments	.82%	-.24%
Quizzes	-4%	12.41%

These results show a similar trend to results as seen in Study II: on quiz performance measures the Experimental treatment significantly outperformed the Control treatment, thus providing empirical support for **H3**.

www.manaraa.com

7.2.4 *Performance – Activity Levels*

To determine if there was a relationship between student activity levels and course performance, I applied a one-tailed Kendall's tau-b correlation (with corrected $\alpha = .008$ to account for six comparisons) on the same eight performance and in the same manner as in Study II. Results indicated no significant correlations in the Control treatment (Table 7.10). Likewise, in the Experimental treatment no significant correlations were detected (Table 7.11), thus providing no empirical support for **H4**.

Table 7.10. Study III Control treatment activity level performance correlations

Performance Metric	Aggregate Social Activity		OSBLE Interactions		Total Events	
	r_τ	$p <$	r_τ	$p <$	r_τ	$p <$
Programming Assignments	.301	.062	.211	.138	.219	.128
Quizzes	-.206	.148	-.135	.243	.356	.033
Written Exam	-.199	.158	-.108	.292	.215	.136

1-tailed - significant at adjusted α <=.008

Table 7.11. Study III Experimental treatment activity level performance correlations

Performance Metric	Aggregate Social Activity		OSBLE Interactions		Total Events	
	r_τ	$p <$	r_τ	$p <$	r_τ	$p <$
Programming Assignments	-.115	.276	-.048	.402	.238	.108
Quizzes	-.340	.041	-.155	.213	.290	.068
Written Exam	-.156	.213	-.126	.259	.379	.026

1-tailed - significant at adjusted α <=.008

7.2.5 *Student Attitudes*

Table 7.12 presents a summary of statistical comparisons between the Control and Experimental treatments with respect to the student attitudinal measures. The table indicates there was a significant difference between the two treatments with respect to coding self-efficacy with the Control treatment experiencing a significantly greater gain in coding self-efficacy. This opposes **H5**'s prediction that the Experimental treatment should promote larger gains in self-

www.manaraa.com

efficacy than the Control treatment. Further, Table 7.12 also provides no support for **H6** and **H8:** no significant differences were detected between the treatments with respect to either sense of community or sociability. However, the results do provide some support for **H7**: as predicted, the Experimental treatment promoted a significantly larger attitudinal shift with respect to peer learning.

Table 7.12. Study III attitudinal comparisons

Attitudinal Metric	***U***	***p*<=**	**r**	***M* Ranks****	***Mdn***
Coding Self-Efficacy	-1.988	**.024***	.584	8:8	.21:.50
CCS: Connectedness	-2.08	.038	.537	8:8	-3.00:.00
CCS: Learning	-1.054	.292	.272	8:8	-2.00:.00
CCS: Total	-1.223	.221	.316	8:8	-3.00:.00
MSLQ: Self-Learning	-.628	.530	.162	7:8	0:0
MSLQ Peer-Learning	-2.263	**.024‡**	.584	8:6	0:-.34
Self-Sociability	-.886	.376	.229	8:9	.00:-.10
System-Sociability	-.503	.615	.130	9:6	.00:-.30

N = 15; **All ratios are Experimental:Control, *significant at adjusted α <=.05
†significant at adjusted α <=.017; ‡significant at adjusted α <=.025

Viewing changes between pre and post course for intent to major showed that one CS major (out of 8) changed their major at the end of the course and two dropped the course at mid-semester. In contrast, none of the non-CS majors (out of 11) changed their response to indicate they were a CS major at the end of the course, though two non-CS majors dropped the course at mid-semester resulting in a net negative change in intent to major.

7.2.6 Social Network

Table 7.13 presents the results of the social network measures for the Control and Experimental treatments. These results show that, although the Experimental treatment did reach higher values for some social network metrics, others were lower than the Control treatment. We see more than double the number of messages between users, as well as higher weighted degree, eigenvector,

www.manaraa.com

network diameter, and average path length values for the Experimental treatment. All other metrics were lower in the Experimental treatment than in the Control treatment. This suggests an increase in activity between fewer users and a less closely-coupled network. Comparisons indicated only weighted degree to be significantly different, with the Experimental treatment having higher values.

Table 7.13. Study III social network statistics

Descriptives	Control Treatment	Experimental Treatment	*Z*	*p<=*	*r*	*M Ranks***	*Mdn***
Number of Messages	449	1093					
Degree	4.7	4.471	-.63	.529	.158	7:8	8:7
Weighted Degree	21.75	64.294	-3.054	**.002**	.764	9:6	66:31
Closeness	.578	.501	-3.054	.049	.491	6:10	.536:.541
Betweenness	12.45	11	-1.086	.278	.272	9:8	2.733:.750
Eigenvector	.322	.427	-1.655	.098	.414	9:7	.440:.335
Density	.247	.279					
Diameter	2	3					
Average Path Length	1.771	1.831					

**significant at adjusted α <=.025; **All ratios are Experimental : Control*

Figure 7.1 and Figure 7.2 show a graphical representation of each treatment's social network. In each of these figures we can see that the most central actor is the instructor (red), who engaged in interactions with students (blue). In these interactions, node sizes are scaled to indicate the extent of the interactions. Inspection of these graphs suggests that some students became less active overall in the Experimental treatment (note the unconnected node in Figure 7.2) which may have contributed to the less connected network.

www.manaraa.com

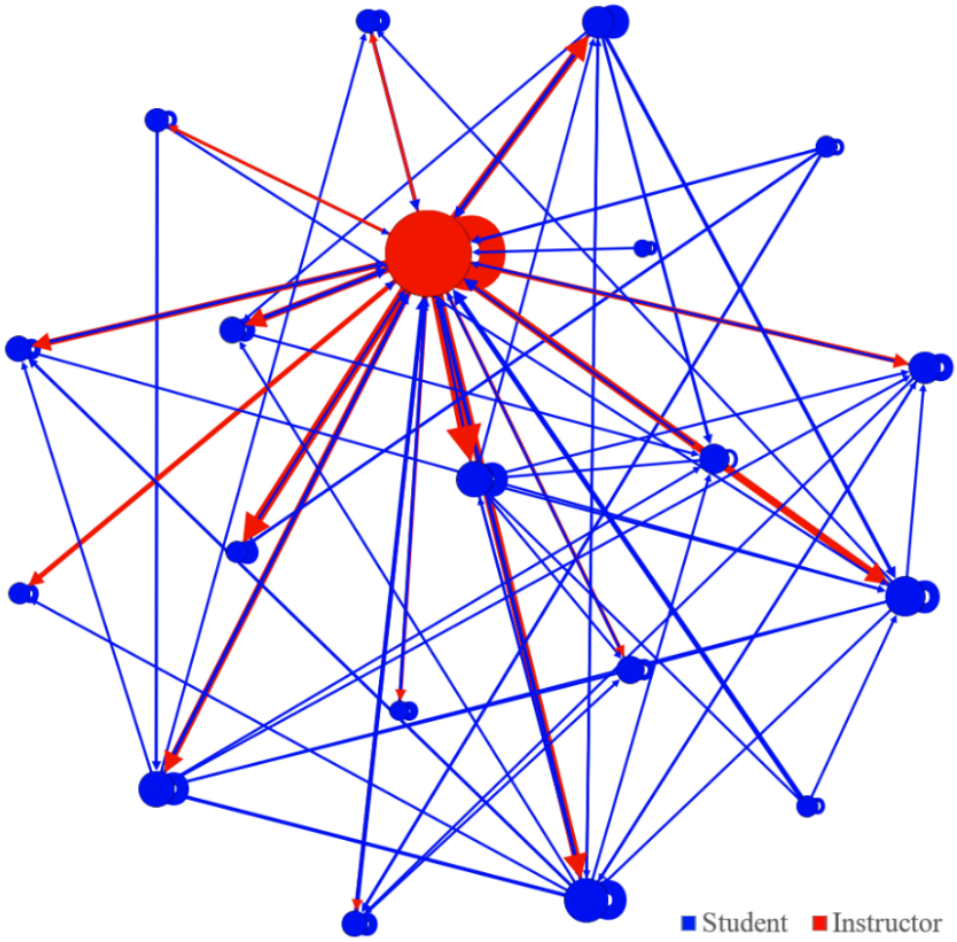

Figure 7.1. Study III (Control) social network graph

www.manaraa.com

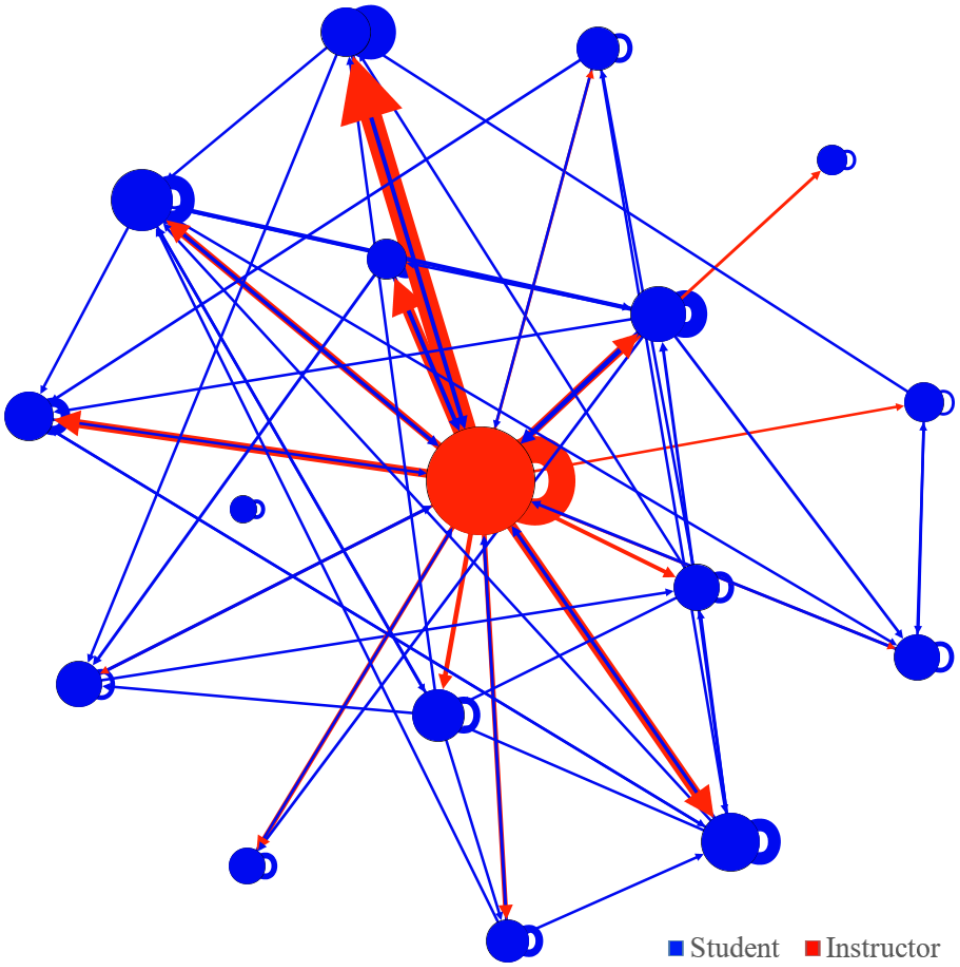

Figure 7.2. Study III (Experimental) social network graph

7.2.7 System Usability

As a means of measuring additional possible confounds, system usability was again measured using the System Usability Scale. The average SUS score for Study III at the midpoint in the semester was 72.8, while the post semester average score was 72.5. Comparison between treatments determined that there was not a significant difference between the two (Mdn_{post} = 77.5, Mdn_{mid} = 75, Z = -1.139, p = .225). Interestingly, this SUS score is even higher than the Study II score of 68, which was significantly higher than either of the Study I average SUS scores (65 and 60 respectively). This trend suggests that the Study III group had an above average perceived

www.manaraa.com

system usability rating, as opposed to the below average perceived ratings observed in Study I. Thus, it was less likely that they encountered usability issues while using the OSBLE+ system in Study III.

7.2.8 Experimental Treatment Intervention Analysis

The previous sections looked at the differences between Control and Experimental treatments. This section looks exclusively at the Experimental treatment to determine if there were any significant effects detected from intervention use.

7.2.8.1 Intervention Generation and Interaction Levels

A total of 4,062 interventions were generated in the Experimental treatment, with an average of 271 interventions generated per student (see Table 7.14).

Table 7.14. Study III interventions generated summary

	Interventions Generated
Min	116
Max	702
Mean	271
Std. Dev.	158
Total	**4,062**

A Kendall's tau-b correlation determined that there were no statistically significant correlations between interventions generated and any of the course performance metrics, nor were there significant correlations between interventions interacted with and course performance (Table 7.15).

www.manaraa.com

Table 7.15. Study III Correlations between performance and intervention generation and interactions

	Interventions Generated		Intervention Interactions	
Performance Metric	τ_b	p	τ_b	p
Final Course Grade	.067	.364	.030	.440
Lab Participation	.070	.374	.117	.301
Quizzes	.193	.160	.142	.240
Programming Assignments	.260	.090	.091	.325
Lab Final Exam	.163	.200	-.192	.169
Written Final Exam	.087	.327	-.143	.240

*N = 15; *significant at adjusted α <=.006*

7.2.8.2 Individual Intervention Interactions

Table 7.16 categorizes interaction log data by individual intervention to determine the degree to which students were presented with and interacted with the interventions. As the table shows, intervention interactions ranged from 0 to 15 unique interactions per intervention, with both runtime and build error interventions having no logged interactions. A total of 173 interactions were split between all 15 of the participants. The maximum number of interactions by a single participant was 24.

Additional analysis of interactions showed that all 15 participants made follow up social actions (OSBLE Interactions, posts, replies, or helpful marks on replies) within 10 minutes of the intervention interaction—a 13% follow up rate by participants. Of the eight interventions, six had follow up interactions. The runtime and build error interventions did not elicit any follow up activity. The quantity of follow up activity ranged between 1 to 8 actions, with posts being the most common first and sometimes second follow up action, followed by interactions with the activity feed and replies (see Table 7.17).

www.manaraa.com

Table 7.16. Study III intervention interaction, generation, and follow up counts by intervention type

Intervention \ Interaction Count	Sum	Max[1]	Unique	Generated[2]	Follow up
Runtime Errors: Get Help!	0	0	0	372	0
Build Errors: Get Help!	0	0	0	2,868	0
Others Available to Help!	71	24	15	109	41
Help other Students!	5	3	3	60	3
Help Your Classmates!	3	2	2	96	3
You Are Available!	77	12	15	12	54
Make a Post! (Topical)	2	1	2	60	1
Make a Post! (Assignment)	15	3	10	485	14

[1]interactions by single student, [2]total for entire treatment

Looking at follow-up by intervention shows that availability status interventions were the most frequently followed up on ("*Others Available to Help!*" Figure 4.14, "*You Are Available!*" Figure 4.19), with the assignment submission reflection the third most followed up interventions (Figure 4.21).

Table 7.17. Study III follow up action count and order

Follow Up:	Post	Reply	OSBLE Interaction*
1st	86	0	30
2nd	11	1	41
3rd	0	17	10
4th	1	4	20
5th	13	3	4
6th	0	2	3
7th	0	1	2
8th	0	0	1

**Counts of follow up interactions on the OSBLE activity feed: posts, replies, replies marked helpful*

7.3 Discussion

The clearest indication that exposure to the interventions might positively influence student behavior can be found in the results relative to **H1** and **H2**: As predicted, the Experimental Treatment promoted significantly more social and programming activity than the Control treatment. However, in this study, increased social activity did not appear to be clearly correlated

www.manaraa.com

with student performance. We instead found was that the results were in conflict: Whereas the Experimental treatment had a significant advantage with respect to quiz performance, the Control treatment had a significant advantage with respect to programming assignment performance. Given these conflicting results, it is difficult to draw clear conclusions from this study regarding the relationship between social activity and student performance.

As in Study II, a plausible explanation for these noisy results relates to the within-subjects design of this study. The Experimental treatment occurred during the second half of the course, when the programming tasks were more conceptually demanding, and the programming solutions required a greater amount of programming effort. This is also reflected in the significant increase in runtime and build errors encountered in the second half of the course (see Table 7.5 and Table 7.6). As a consequence, increased social and programming activity may have been more a result of an increased volume of code needed to be written, and the increased need for help, than a result of the interventions themselves.

Interestingly, in contrast to Study II, the analysis of the relationships between intervention generation and interaction in Study III does not suggest that intervention generation and usage were correlated with course performance. However, as in Study II, we did see that, while build and runtime error interventions were among the top generated interventions, they were among the least interacted with interventions of all. This suggests there may have been deeper problems with the interventions that prevented them from being embraced by students. Even though we saw fewer significant differences between the Experimental and Control treatments in Study III, a key result from Study II was replicated: namely, there was significantly more social activity in the Experimental treatment than in the Control treatment. This key result is summarized in Table 7.18 in terms of the percent differences in activity between the Experimental and Control treatments.

www.manaraa.com

Table 7.18. Study III Experimental activity counts percent difference

Ask-For-Help	Build	Cut-Copy-Paste	Debug	Editor	Build Error	Build Error Counts
0%	97%	136%	1010%	81%	44%	93%
Exception	**Save**	**Submit**	**Helpful Mark**	**OSBLE Interaction**	**Post**	**Reply**
629%	124%	100%	0%	105%	150%	300%
Total Event Counts: 167%						

**percent difference – positive indicates Experimental counts higher than Control counts; **per student counts, see Table 7.1*

Closer analysis of student utilization of specific interventions leads to the same conclusion as in Study II: namely, that the availability status-related interventions (help-seeking: "*Others Available to Help!*" and help-giving: "*You Are Available!*") were likely the most influential in promoting student activity within the SPE.

I conclude this discussion by identifying two additional confounds that may have affected these results. The within-subjects design dictated that the Experimental treatment followed the Control treatment. This introduced a potential transfer-of-training effect that may have confounded the results. Indeed, when students started the Control treatment, they had no prior knowledge of the SPE. By the time they started the Experimental treatment, they had four weeks of experience. This experience could have impacted their awareness of how the system works and their willingness or ability to use the system. Likewise, the fact that the Experimental treatment occurred in the second half of the course meant that students were tackling more conceptually-difficult programming tasks in the Experimental treatment than in the Control treatment (in addition to the varying topics and complexities of assignments between treatments). Thus, the increased activity within the SPE in the Experimental treatment may have been attributed, at least in part, to the need for larger coding solutions and increased levels of help. These concerns will be considered further in the general discussion presented in the following chapter.

www.manaraa.com

CHAPTER 8

GENERAL DISCUSSION OF STUDY I, II, AND III

The preceding three chapters presented a succession of experimental studies designed to address two primary research questions:

RQ1: Can the SPE interventions promote positive changes in students' social and programming behaviors in the OSBLE+ environment?

RQ2: Will these interventions lead to positive changes in students' learning outcomes, and persistence within the computer science discipline?

In this chapter, I take a step back to present a discussion of the results of the three studies. I organize this discussion around the results with respect to each dependent variable and its related hypothesis. For each set of results, I consider three key questions: (1) *What was similar between studies?* (2) *What was different between studies?* and (3) *What could have impacted (positively or negatively) the outcomes of these studies?* I conclude the chapter with a general discussion.

8.1 Participation

With respect to student participation in the SPE, in Study I we saw that both treatment groups were similar with respect to activity levels. In contrast, Study II and Study III yielded significantly more activity in the Experimental treatment than the Control treatment for most metrics. Except for total events in Study II, both Study II and Study III did not show significant differences in changes in participation over time between treatments—i.e. both treatment periods averaged a similar change in activity levels over treatment periods.

www.manaraa.com

One major difference between all three studies was the distribution of post and reply activity between students and instructors (or teaching assistants). We see in Figure 8.1 that most posts and replies were made by both instructors and TAs, with only 34-42% of posts and replies made by students. In contrast, in both Study II and Study III (Figure 8.2, Figure 8.3), the majority of posts and replies were made by students. Across the board we also see that students made more posts than replies for all treatments.

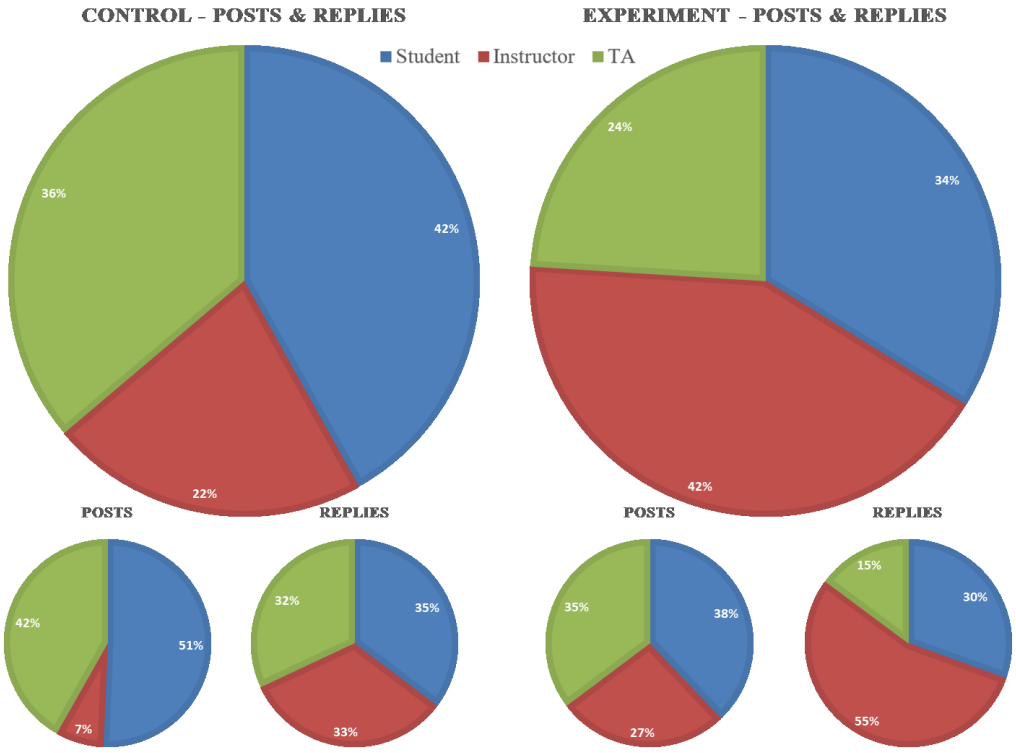

Figure 8.1. Study I post and reply distribution

www.manaraa.com

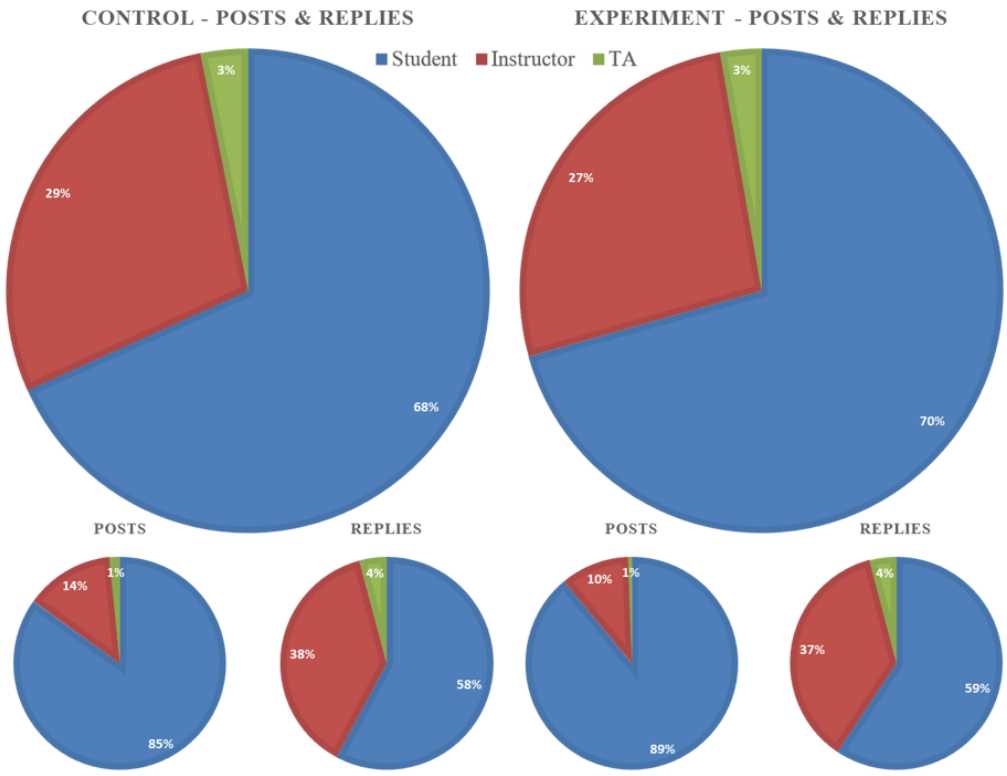

Figure 8.2. Study II post and reply distribution

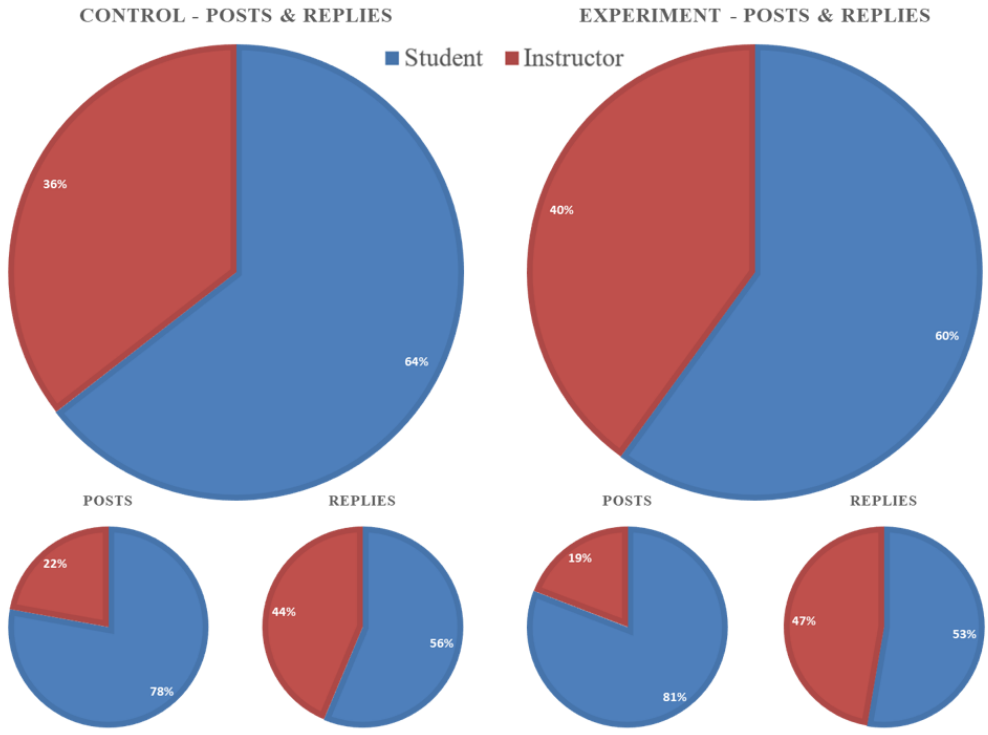

Figure 8.3. Study III post and rely distribution

www.manaraa.com

These results suggest that the level of involvement of teaching personnel may play a role in the extent to which students interact with the SPE, as suggested by Carter (2016). Indeed, we see higher activity levels by instructors in Study I and III during the Experimental treatments, and a corresponding reduction in students' activity level. In contrast, Study II showed a lower activity level by instructors during the Experimental treatment and an overall higher activity level by students.

Another major difference between studies is that students were not *required* to participate in the SPE activity feed in Study I, whereas they were required to do so in Study II and Study III. In Study II and Study III, we saw a 144% to 300% increase in posts and replies in the Control to Experimental treatments respectively. It seems likely that at least part of this increase can be attributed to the new posting requirement. However, since the requirement was held constant in both the Control and Experimental treatments, it does not explain the statistically significant differences in activity levels between the Control and Experimental treatments in Studies II and III.

Based on the above discussion, I believe there is some case to be made that students' exposure to the interventions in Studies II and Study III contributed to their increased levels of activity within the SPE.

8.2 Programming Activity

Programming activity was similar between groups in Study I, with some significant differences in each group. In contrast, both Study II and Study III showed a trend of increased programming activity between treatments, with more activity in the Experimental treatment period. In the case of Study II, only runtime exceptions were significantly higher, while in Study III all programming

www.manaraa.com

activity events were higher. Time in error states for Study II and III was consistent; only exception events significantly increased in the Experimental treatment. The within-subjects design, in which students were exposed to the Experimental treatment in the second half of the semester, likely influenced this result.

We see a clear trend toward increased programming activity in Study II and Study III; however, this trend does not provide support for the hypothesis that exposure to the interventions should lead to should a reduction in "bad" programming activity or time in error states.

8.3 Performance – Achievement

Course performance results were also a mixed bag across all three of the studies. Study I did not show any significant differences between treatments, while both Study II and III showed significant differences in both programming assignment and quiz grades. However, the direction of the differences was not always as predicted. In both Study II and Study III, we saw worse performance for programming assignments in the Experimental treatment, but improved performance on quizzes (see Table 6.8, Table 7.9). These contradictory results may be due to differences in intervention utilization observed between Study II and III. They may also simply indicate that the interventions have a marginal impact on course performance and that we would need a larger sample size to detect a statistically reliable difference.

8.4 Performance – Activity Levels

When analyzing whether there was a correlation between activity levels and course performance, we see a different trend: Studies I and II both show significant weak to moderate positive correlations between activity levels and course performance for programming assignments, quizzes, and exams, while in Study III no significant correlations for any course

www.manaraa.com

performance metric were detected. Based on the motivating learning theories [e.g. see student involvement theory (Astin, 1999)] and academic engagement as a predictor of performance [specifically academic participation; see (Schlenker, Schlenker, & Schlenker, 2013)], we would expect to see higher performance tied to higher activity levels. Thus, across the three studies, we have only limited support for the hypothesis that higher activity levels are correlated with better course performance.

8.5 Student Attitudes

Analysis of student attitudes was mixed across each of the studies. For Study I, I was able to do a direct comparison between treatments due to the between-subjects study design. For Study II, due to an error in executing the study procedure, I failed to gather mid-survey responses to facilitate treatment comparisons. This error was resolved for Study III, in which I was able to perform treatment comparisons.

In Study I, only one significant difference was found in the seven attitudinal measures: self-sociability, which was found to be significantly higher in the Control group. Since self-sociability may serve as a proxy for one's willingness to engage socially online, one might speculate that it could explain differences between treatment groups, and hence should be controlled for in any experimental comparison. To see if a difference in self-sociability might have played a role in the differences observed in Studies II and III, I performed a Mann-Whitney U test on the self-sociability scores of participants in Study II and III. Results indicated that self-sociability was not significantly different in the participants of the two studies ($Mdn_{study2} = 3.1$, $Mdn_{study3} = 3.1$, $U = 179$, $p = .799$), so it does not seem that sociability differences could have influenced the differences in the results observed in Studies II and III.

www.manaraa.com

Study III indicated a significant increase in coding self-efficacy between treatments. However, the significant increase occurred during the Control treatment period of the study. A possible explanation for this is that learners gained significantly more coding confidence during the first half of the semester while learning easier core topics. This is the reverse of what we saw during the Experimental treatment, programming assignments and course topics increased in difficulty. To reduce the possibility that learning effects played a role in this result, we would need to change to a between-subjects experimental design.

The final significant attitudinal difference detected in Study III was in peer learning, with the Experimental treatment experiencing a significantly higher increase. This indicates that students had a higher willingness to initiate peer dialog about course material in the Experimental treatment. While the interventions may have played a role in this significant difference, one cannot rule out the possibility that students' previous experience with peer dialog in the first half of the course also contributed.

Based on these results, we cannot conclude that exposure to the interventions had the positive effect on student attitudes posited by the hypotheses. Rather, we can only confirm that students became more confident in their ability to code, and that their willingness to engage with their peers in dialog increased as they progressed through the course. One significant factor that may have negatively affected Study I was the significantly lower self-sociability detected by the Experimental group. Beyond that, many of the attitudinal measures were not significantly different between treatments and students did not significantly change their intention to major regardless of treatment group.

www.manaraa.com

8.6 Social Network Analysis

Analysis of Study I treatment groups determined that, though the Experimental treatment had an overall less connected network, there were no significant differences between groups.

As mentioned earlier, both Study II and III showed an overall increase in posts by the Experimental treatment, with Study II replies following this trend and Study III showing fewer replies by students (see Figure 8.2 and Figure 8.3).

Results of the social network analysis for Study II showed a marked increase in important network statistics. Specifically, in Study II, treatment groups differed significantly with respect to degree, weighted degree and closeness. Further, participants in Study II had lower average path lengths, indicating a more closely coupled network. This provides support for the posed hypothesis.

In Study III, there was less support and, in fact, a less connected network (although not less connected to a degree that reaches statistical significance) for the Experimental treatment period. In contrast, weighted degree was determined to be significantly higher for the Experimental treatment, indicating that, though less connected, there were some nodes with a higher number of connections. This follows from the Experimental treatment having more frequent communication between fewer people. Inspection of the network visualization (Figure 7.2) suggests that this was due to a higher density of replies from instructor to students.

Based on this analysis, we only see mixed support for the hypothesis that exposure to interventions would result in more closely coupled networks. A confounding factor that must be considered is the natural tendency for more closely coupled networks to occur over time; i.e., shared activities and peers (like those seen in a classroom) greatly increase the likelihood of

www.manaraa.com

individuals becoming more connected (Kossinets & Watts, 2006). Thus, the **H10** garners only mixed support, with the results of Study II providing some support and Study I and III indicating that we should accept the null hypothesis.

8.7 System Usability

Interestingly, Study I system usability was determined to be significantly higher in the Experimental treatment but a lower average rating than the Control treatment. Both treatment groups' mean usability rating was below average.

Study II reported a higher mean rating than Study I and indicated an above average usability score. Again, due to a failure to implement the study procedure properly, a comparison of system usability by treatment was not possible.

Study III showed a further improved system usability score when compared to Study I and II; however, it was determined that there were no statistically significant differences between the two treatments.

This analysis is limited in that it indicates perceived usability issues without attributing these perceptions to possible causes. At the same time, the higher, above average, usability ratings in Studies II and III are consistent with the increased activity levels in those studies. One explanatory factor here could be the instructor's involvement with OSBLE system and intervention deployment in Study II and III. In contrast, in Study I the instructor was not involved with the system and only used it to facilitate assignment submissions.

www.manaraa.com

8.8 Intervention Generation and Interactions

Results across all three studies indicate a range of plus or minus 64 interventions generated per student, with Study I containing the fewest (262), Study II having the second fewest (271) and Study III having the most interventions generated per person (326).

In Study I, all course performance metrics were determined to have a positive correlation with intervention generation and intervention interactions were positively correlated with overall course performance and programming assignment performance. Based on the near nonexistent individual intervention interactions, however, it is hard to explain the significant correlations that were detected.

Study II yielded a positive correlation between interventions generated and overall course performance, while intervention interactions were positively correlated with both quiz performance and the lab final exam performance. Intervention interaction levels were much higher in Study II, lending more credence to these results.

Unlike Study I and II, analysis in Study III showed no significant correlations between intervention generation/interaction and any of the course performance metrics. Based on the inconsistent results across the three studies, it is difficult to make a case in support of the hypotheses.

8.9 Individual Intervention Interactions

In Study I, we saw minimal to no interaction with the interventions. In Study II and III, in contrast, individual students interacted with the interventions up to 38 times and most, participants interacted with the interventions at least once. Similarly, in both Study II and Study III, most participants performed at least one follow-up action after interacting with an intervention. The

www.manaraa.com

most common follow-up activity was a post to the activity feed. Following this trend of increased activity, we also saw that Study II participants perform up to 12 follow-up actions within a 10-minute period of intervention interactions, while Study III participants performed up to eight follow-up actions.

The runtime and build error interventions generated only five interactions in total across all three studies (four in Study I and one in Study II), and no follow up actions. This clearly indicates an issue with either the timing, visibility, or content of these interventions.

In both Study II and III, the availability status interventions were the most frequently followed up on ("*Others Available to Help!*" Figure 4.14, "*You Are Available!*" Figure 4.19), with the assignment submission reflection the second-most followed up on (Figure 4.21). This finding suggests that these interventions had the most effect on students' activities in the SPE. One factor that must be considered in this analysis, however, is that we see the highest level of interaction and follow up activity in those studies in which students received credit for interacting with the activity feed. Clearly this had an impact on students' overall interaction with the activity feed, although the fact that we found significant differences in levels of interaction between the Control and Experimental treatments suggests that the incentivization of student interaction in Studies II and III cannot explain our results.

8.10 Discussion

The goal of the social programming interventions was to increase the amount of social interaction within the problem-solving environment (the learning dashboard, OSBLE, and the programming environment, Visual Studio). Once we modified the study procedure to provide a further incentive to interact within the SPE in Studies II and III, the results clearly show that these

www.manaraa.com

interventions had the desired effect: There was a statistically-significant increase in activity in the Experimental treatments in both Study II and III.

Overall, the three studies have yielded some findings that suggest the potential promise of the interventions in promoting social activity. At the same time there are some mixed results and confounding factors that weaken the empirical case for many of the specific hypotheses. For instance, the decision to incentivize usage of the system through extra credit opportunities may have skewed the results towards some level of interaction with certain interventions. Additionally, the within-subjects design used in Studies II and III set up a situation in which a learning effect could occur between the Control and Experimental treatments, thus potentially contributing to the positive results supporting the hypotheses and making it difficult to disentangle the learning effects from the effects of the interventions themselves.

One issue consistent across all three studies was that it was extremely difficult to elicit participation and interaction with the system, even with monetary (Study I) or extra credit (Study II and III) incentives. Study I was performed over two semesters, with the instructors and teaching assistants only minimally invested in the system. This certainly played a role in students' lack of engagement with the system (Balaji & Chakrabarti, 2010; Ntourmas, Avouris, Daskalaki, & Dimitriadis, 2018; Selim, 2007). In contrast, Studies II and III were both taught by the same instructor (me) who was highly invested in the system. Not coincidentally, we saw a marked increase in interaction with most aspects of the system. Despite the instructor's enthusiasm in Studies II and III, it was still difficult to get students to use the interventions in more than a cursory manner and, unfortunately, many did not interact with some of the interventions (e.g. runtime and build error interventions). In interpreting the results of Studies II and III, then, one must consider that even with strong instructor buy-in, many of the interventions still saw only limited use.

www.manaraa.com

If we look at Experimental treatment intervention generation and interactions for each of the studies (Table 8.1), we see that Study I generated the highest number of interventions by an individual. However, if we consider the extended length of the semester in Study I vs. Studies II and III, we see a reduced number of interventions generated over time in comparison to Study II and III (113 per week versus 176-288 per week). This trend is also present with respect to student intervention interaction and activity levels.

Table 8.1. Intervention interaction summary

Study I				
	Max	**Mean**	**Std dev**	**Mode***
Interventions Generated	1693	262	343	0
Intervention Interactions (click-through)	3	< 1	< 1	0
Interventions / Total System Events	18.46%	3.09%	2.82%	0%
Interactions / Total System Events	< 1%	< 1%	< 1%	0%
Interactions / Intervention Generated	7.14%	< 1%	< 1%	0%
Study II				
	Max	**Mean**	**Std dev**	**Mode***
Interventions Generated	1153	326	287	< 129
Intervention Interactions (click-through)	38	13.58	9.39	8
Interventions / Total System Events	12.30%	5.06%	2.67%	< 3%
Interactions / Total System Events	2.17%	.32%	< 1%	0%
Interactions / Intervention Generated	30%	7%	7.19%	0%
Study III				
	Max	**Mean**	**Std dev**	**Mode***
Interventions Generated	702	271	158	< 238
Intervention Interactions (click-through)	37	11.53	8.59	6
Interventions / Total System Events	13.16%	3.42%	3.63%	< 3%
Interactions / Total System Events	< 1%	< 1%	< 1%	< 1%
Interactions / Intervention Generated	37.84%	8.77%	11.49%	< 3%

**No single mode for all cases. Most frequent percent below threshold shown instead.*

Looking at the most frequent level of interaction with interventions, we see zero for Study I and six and eight for Study II and III respectively. Further, across all studies, participants most

www.manaraa.com

frequently interacted with less than 3% of all interventions generated, with 0% being the most frequent interaction level for both Study I and II. Clearly, a high number of participants in these studies did not use the interventions at all.

Comparing Study I with Studies II and III, we see an increase of over 1000% in interactions with interventions. This leads to the conclusion that, despite some minimal level of interaction, there was not a high enough level of interaction to perform any intervention-specific analysis in Study I. If we look at the most frequent activity levels, we see that interactions were still most frequently less than 3% in Study III and 0% for Study II (due to a split between high activity and low activity students), raising concern as to the strength of the conclusions that can be drawn from the studies.

Based on this analysis of the results across all studies, we conclude that most people did not interact with the interventions. Though there was some support for an increased level of activity and significant positive correlations between intervention activity and course performance, it is not a conclusive result. Because of this, it might make more sense to analyze the differences between those who did use the interventions and those who did not. This analysis approach is taken in the study described in the next chapter.

www.manaraa.com

CHAPTER 9

STUDY IV

Considering the small number of participants who actively used the interventions (as discussed in Chapter 4), this chapter presents a study that takes an alternative approach to analyzing the data collected in Studies II and III. In so doing, it addresses the following research question:

RQ3: Is there a difference in participation, performance, and intent to major between students who used the interventions and those who did not?

In the Chapter 8 conclusions (Table 8.1), we saw that, most frequently, participants interacted with fewer than 3% of the interventions across all three studies. In contrast, some users interacted with as many as 37 percent of their interventions. Thus, while a direct comparison between treatment groups did not produce strong conclusions, a comparison between those who interacted with the interventions at more than a minimal level and those who interacted with the intervention minimally or not at all will be helpful in further understanding the impact of *actual* intervention use on individual learners.

In order to address this new research question, this chapter presents Study IV, which reanalyzes the data collected in Studies II and III. The chapter begins by categorizing participants from Studies II and III into two groups: *low* interaction intervention users and *high* interaction intervention users. Next, the chapter presents an analysis of activity and performance levels similar to the analyses presented for Studies I-III but using the newly categorized intervention use groups instead of treatment groups. The chapter concludes with a discussion of results and their implications.

www.manaraa.com

9.1 Categorization of Intervention Interaction Levels

The first step of this analysis to determine a minimum threshold of intervention activity to be considered for the different interaction intervention groups. For the purposes of this re-analysis, I will focus on participants from Study II and III because the maximum intervention interactions count for any single participant in Study I was just three, with fewer than 10% of participants interacting with any interventions at all. This minimal level of interaction resulted in a highly imbalanced grouping of participants with no means of balancing groups for further analysis. Additionally, as Study II and III employed an identical experimental design, the data sets can be easily combined for analysis.

A k-means cluster analysis (Macqueen, 1967) was performed on the combined study intervention interaction levels variable (i.e. intervention interaction counts by participants) in order to classify participant intervention interaction levels into two distinct groups: *low* interaction level (hereafter referred to as LIL) versus *high* interaction level (hereafter referred to as HIL). Forming two clusters resulted in an imbalanced distribution (LIL: N = 30, HIL: N = 11). To better balance group size, three clusters were formed. This resulted in three groups, low activity (N = 22), medium activity (N = 5), and high activity groups (N = 14). For further analysis, the five participants in the medium activity group were merged into the high activity group, resulting in a more balanced pair of activity groups (Table 9.1).

Table 9.1. Study IV intervention interaction level clustering results

Interaction Level	#Cases	Activity Range*
Low (**LIL**)	22	0-15
High (**LIL**)	19	18-38

**Activity range indicates number of intervention interactions for category*

www.manaraa.com

9.2 Alternative Analysis

The revised analysis performed here employs the same analysis methods used previously, but with the combined Study II and Study III participant data. There were two differences, however. First, the social network analysis was more limited in scope due to the disconnected nature of the new comparison groups, which came from two separate semesters of data. Second, it was possible to do a statistical analysis of differences for system usability scores between groups.

9.2.1 Methods

After grouping participants by activity level, data was re-analyzed to determine differences between groups. Unlike the analysis of Study II and III, which required an alternate means of analyzing activity between treatments due to the within-subjects design, we could perform a direct comparison between individual assignment activity and performance because the focus was only on the Experimental treatment period, when the same course materials were used.

Additionally, hypotheses were slightly modified to address the shift in focus to intervention interaction levels and the RQ posed at the start of this chapter.

Finally, Mann-Whitney U tests were performed throughout the reanalysis for comparison of independent groups, as was done in Study I (5.1). Similarly, Bonferroni correction was used to adjust significance levels in the same manner as previous studies to account for multiple comparisons.

9.2.2 Participants

Participants were those who participated in Study II (6.1.1) and Study III (7.1.1).

www.manaraa.com

9.2.3 Materials

The same materials used in Study I, section 5.1.2, were used for the combined analysis.

9.2.4 Log Data

New log data was not collected for Study IV; analysis was performed on Study II and III data.

9.2.5 Independent Variable

The independent variable remained the same as described in Study I (see section 5.1.3).

9.2.6 Dependent Variables

All dependent variables seen in Study I, section 5.1.4, remained the same in Study IV.

9.2.7 Procedure

This reanalysis of data employed the same analysis methods followed in Chapters 5-7.

9.3 Results

9.3.1 Participation

Table 9.2 presents a statistical comparison of the two groups, LIL and HIL, with respect to the log data relevant to participation. Results indicate two of the three measures, aggregate feed participation and OSBLE interaction, were determined to be significantly different between groups. In all cases, the HIL group had the higher mean rank activity levels.

Table 9.2. Study IV participation comparisons

Social Metric	*U*	*p*<=	*r*	*M* Ranks**	*Mdn***
Aggregate Feed Activity	103.000	**.006**	.433	27:16	68:39
OSBLE Interactions	92.000	**.002**	.478	27:16	231:138
Total Events	207.000	.958	.008	21:21	9795:10574

*N = 41; **All ratios are HIL:LIL, *significant at adjusted α <=.0167*

www.manaraa.com

Similarly, performing the same comparison across only the Experimental treatment assignments, PA5-8, yielded the same significant results (Table 9.3).

Table 9.3. Study IV participation comparisons over active assignment periods

Social Metric	*U*	*p*<=	*r*	*M* Ranks**	*Mdn***
Aggregate Feed Activity	1748	**.000**	.415	104:64	7:3
OSBLE Interactions	1940	**.000**	.363	101:67	20:8
Total Events	3292	.864	.013	83:82	1239:1141

*N = 41; **All ratios are HIL:LIL, *significant at adjusted α <=.0167*

Based on these results, **H1** is clearly confirmed: those who interacted with the interventions maintained a similar total event activity levels but were significantly more active in the social activity feed—i.e., a significantly higher proportion of the HIL group's total event activity was on the activity feed.

9.3.2 Programming Activity

Table 9.4 presents statistical comparisons of the LIL and HIL groups with respect to each measure relevant to programming activity. Results failed to yield any statistically significant differences. Table 9.5 shows the same results for differences over PA 5-8 and change in programming activity over time (Table 9.6).

Table 9.4. Study IV comparison of programming activity

Performance Metric	*U*	*p*	*r*	*M Ranks***	*Mdn***
Build	193	.676	.065	22:20	733:684
Cut-Copy-Paste	203	.875	.024	21:21	1006:1014
Debug	195	.714	.057	20:22	2334:2775
Editor	176	.388	.135	23:20	2049:1706
Build Error	166	.261	.176	23:19	417:256
Exception	205	.906	.018	21:21	91:96
Save	183	.497	.106	22:20	944:717
Submit	186	.539	.096	22:20	35:39

*N = 41; **All ratios are HIL:LIL, *significant at adjusted α <=.006*

www.manaraa.com

Table 9.5. Study IV comparison of programming activity over active assignment periods

Performance Metric	*U*	*p*	*r*	*M* Ranks**	*Mdn***
Build	2929	.171	.107	88:78	133:106
Cut-Copy-Paste	3301	.886	.011	82:83	136:161
Debug	3340	.989	.001	83:82	201:251
Editor	2844	.099	.129	89:77	247:238
Build Error	2874	.121	.121	89:77	39:26
Exception	3160	.539	.048	85:80	11:9
Save	2947	.190	.102	88:78	153:114
Submit	2867	.113	.124	89:77	5:4

*N = 41; **All ratios are HIL:LIL, *significant at adjusted α <=.006*

Table 9.6. LIL-HIL comparison of programming activity change over time

Performance Metric	*U*	*p*	*r*	*M* Ranks**	*Mdn***
Build	194	.685	.063	22:20	.83:.58
Cut-Copy-Paste	135	.053	.302	17:24	.48:1.05
Debug	202	.855	.029	21:21	1.83:1.41
Editor	157	.170	.214	18:23	.34:.55
Build Error	203	.865	.027	21:21	1.20:1.27
Exception	149	.114	.247	24:18	3.38:1.80
Save	195	.705	.059	20:22	.58:.63
Submit	189	.592	.084	22:20	.56:.64

*N = 41; **All ratios are HIL:LIL, *significant at adjusted α <=.006*

As in previous the studies, these results indicate that programming activity levels between groups were similar overall. They do not provide empirical support for **H2**.

9.3.3 Performance – Achievement

Analysis of course performance indicated that, on average, the HIL group performed better across the board for all performance metrics, with up to an 8% increase in all scores (see mean performance column: Table 9.7). However, a Mann-Whitney U test indicated that there was no statistically significant difference between activity level groups.

www.manaraa.com

Table 9.7. Study IV performance comparisons

Performance Metric	*U*	*p*<=	*r*	*M* Ranks**	*Mdn***	*M***
Final Course Grade	177	.395	.133	23:20	.90:.88	.89:.85
Lab Participation	177	.341	.149	23:20	1.00:1.00	.91:.87
Quizzes	176	.381	.137	23:19	.85:.74	.82:.74
Programming Assignments	153	.143	.229	24:18	.92:.84	.84:.76
Lab Final Exam	208	.979	.004	21:21	.79:.80	.79:.79
Written Final Exam	194	.685	.063	22:20	.72:.74	.76:.71

*N = 41; **All ratios are HIL:LIL, *significant at adjusted α <=.008*

9.3.4 Performance – Activity Levels

A Kendall's tau-b correlation was performed in order to determine if there was a relationship between activity levels and course performance. Results indicated no significant correlations on for the LIL group (Table 9.8).

Table 9.8. Study IV LIL activity level performance correlations

Performance Metric	Aggregate Social Activity		OSBLE Interactions		Total Events	
	r_τ	$p <$	r_τ	$p <$	r_τ	$p <$
Final Course Grade	.243	.060	.201	.097	.092	.277
Lab Participation	.225	.091	.130	.219	-.258	.061
Quizzes	.167	.141	.039	.400	-.174	.130
Programming Assignments	.075	.315	.158	.154	-.092	.277
Lab Final Exam	.317	.021	.196	.102	.035	.411
Written Final Exam	.292	.031	.306	.024	.048	.378

*N=41; 1-tailed; *significant at adjusted α <=.006*

When looking at only the HIL group, final course grade, programming assignments, and the written final exam showed significant weak to moderate positive correlations (Table 9.9).

www.manaraa.com

Table 9.9. Study IV HIL activity level performance correlations

Performance Metric	Aggregate Social Activity		OSBLE Interactions		Total Events	
	r_τ	$p <$	r_τ	$p <$	r_τ	$p <$
Final Course Grade	.078	.324	.161	.171	.473	**.003**
Lab Participation	.113	.274	.056	.382	.202	.142
Quizzes	.288	.043	.135	.211	.347	.019
Programming Assignments	-.030	.430	.089	.299	.425	**.006**
Lab Final Exam	.148	.190	.194	.124	.29	.043
Written Final Exam	.191	.130	.261	.061	.513	**.001**

*N=41; 1-tailed; *significant at adjusted α <=.006*

Significant positive correlations were found when comparing HIL group activity levels with several of the performance course performance metrics. These results provide limited support for **H3**, though only in a correlational sense.

9.3.5 Student Attitudes

Table 9.10 presents a comparison of student attitudinal changes experienced for the LIL and HIL groups. The results presented in Table 9.10 provide no support for **H5**, **H6** or **H8**; no significant differences were detected between the groups with respect to coding self-efficacy, sense of community or sociability. However, the results do provide some support for **H7**: as predicted, students in the HIL group experienced a significantly larger attitudinal shift with respect to peer learning. This result suggests the HIL group had a higher willingness to use study groups or friends to help learn.

Viewing changes between pre and post course for intent to major showed that two CS majors (out of 20) changed their major at the end of the course. In contrast, one of the non-CS majors (out of 21) changed their response to indicate they were a CS major at the end of the course, resulting in a net negative change in intent to major.

www.manaraa.com

Table 9.10. Study IV attitudinal comparisons

Attitudinal Metric	*U*	*p*<=	r	*M* Ranks**	*Mdn***
Coding Self-Efficacy	206	.927	.014	22:20	5.59:5.72
CCS: Connectedness	185	.529	.098	20:22	24.00:25.00
CCS: Learning	162	.213	.194	19:23	24.00:27.00
CCS: Total	181	.464	.114	20:22	49.00:51.00
MSLQ: Self-Learning	186	.537	.096	20:22	5.14:5.43
MSLQ Peer-Learning	112	**.011**‡	.400	26:17	5.00:4.00
Self-Sociability	194	.880	.023	21:20	3.10:3.00
System-Sociability	193	.665	.068	22:20	3.80:3.60

N = 41; **All ratios are HIL:LIL, *significant at adjusted α <=.05
†significant at adjusted α <=.017; ‡significant at adjusted α <=.025

9.3.6 Social Network

To investigate **H10**, the post-reply relationships for each of the intervention interaction groups were again analyzed using the social network statistics generated from Gephi. Table 9.11 presents the results of a limited set of the social network measures for both the high and low intervention interaction groups. These were calculated by taking the average of individual metrics for each interaction level group (as discussed in the Chapter 5-7 social network analysis). From this, we see that the HIL group's social network statistics for number of messages and weighted degree were both significantly higher than those of the LIL group, indicating stronger connections between some nodes. We can also see higher values for all reported metrics. suggesting a stronger, more closely coupled network.

www.manaraa.com

Table 9.11. Study IV social network statistics comparisons

Descriptives	LIL	HIL	*U*	*p*<=	*r*	*M* Ranks**	*Mdn***
Number of Messages	449	930	108	**.008**	.413	26:16	36:19
Degree	11.19	13.11	165	.341	.054	22:19	13:1
Weighted Degree	64.29	140.79	99	**.006**	.001	26:16	108:54
Closeness	.504	.522	177	.532	.084	22:19	.54:.52
Betweenness	10.75	9.91	184	.665	.105	21:20	6.03:5.22
Eigenvector	.418	.538	140	.104	.016	24:18	.48:.37

*N = 41; *significant at adjusted α <=.008; **All ratios are HIL:LIL*

9.3.7 System Usability

As a means of identifying a possible confound, system usability was measured via the System Usability Scale. Results of the comparison indicated no significant difference between the LIL (*Mdn* = 73) and the HIL (*Mdn* = 75) with regard to calculated system usability scores, $U = 181$, $p = .454$.

9.4 Analysis of Intervention Interactions by Intervention Type

I now take a brief look at which interventions were used most frequently based on interaction level groups. The aim of this analysis is to shed light on which interventions had the most impact on the observed differences between groups.

Looking at the overall intervention interactions, we see that the HIL group interactions consisted of 74.5% of all interactions with the LIL group producing 25.5% of the interactions, indicating that the HIL group interacted with nearly three times more interventions overall. Figure 9.1 shows the breakdown of the percent of interactions with each intervention type by group; the HIL group had significantly higher interaction levels with all interventions.

www.manaraa.com

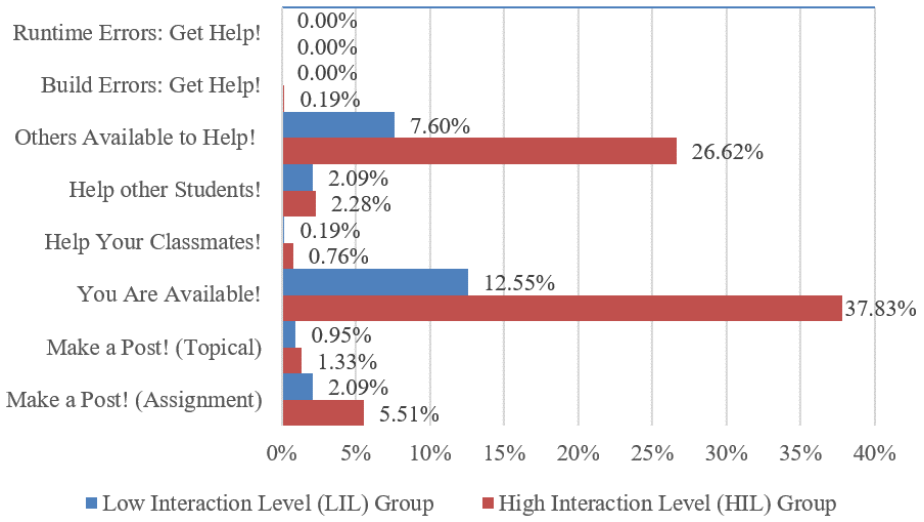

Figure 9.1. Study IV revisited analysis – percent of intervention interactions by group

If we look at the interaction and follow up actions by group, we can see that not only did the HIL group interact more (as expected, as that is how they were categorized into two groups) but they also followed up more often than the LIL group (Table 9.12).

Table 9.12. Study IV revisited analysis – intervention interactions by group

Intervention	Sum		Max[1]		Unique		Generated[2]		Follow up	
	LIL	HIL	LIL	HIL	LIL	HIL	LIL	HIL	LIL	HIL
Runtime Errors: Get Help!	0	0	0	0	0	0	24	33	0	0
Build Errors: Get Help!	0	1	0	1	0	1	225	232	0	0
Others Available to Help!	40	140	4	24	20	19	5	5	22	79
Help other Students!	11	12	3	3	8	7	4	4	4	2
Help Your Classmates!	1	4	1	2	1	3	3	8	1	3
You Are Available!	66	199	6	22	20	19	1	1	34	118
Make a Post! (Topical)	5	7	3	3	3	5	4	4	5	5
Make a Post! (Assignment)	11	29	2	3	9	14	29	32	9	27

[1]interactions by single student, [2]calculated per student – LIL n = 22, HIL n=19

Further, we can see that most of the interactions occurred with the interventions focusing on user availability: "*Others Available to Help!*," Figure 4.14, and "*You Are Available!*," Figure 4.19. In contrast, the runtime and build error interventions received zero interactions across the board. As discussed in Chapter 8, this suggests there were problems with the runtime and build error

www.manaraa.com

interventions. From this, we might conclude that the interventions with the most effect on performance were the availability status-related interventions, with the assignment submission reflection intervention also playing a significant role.

9.5 Discussion

This analysis indicated the HIL group was significantly more active on the activity feed in terms of posts, replies and interactions. Interestingly, the total number of events generated by the HIL group was not significantly different from the LIL group, indicating that for a similar number of total system events, a significantly higher proportion of those in the HIL group were social interaction events (Table 9.2). The strength of these differences is highlighted when we look at active time working on programming assignments (Table 9.3). This provides support for the hypotheses which posit that increased interactions with the interventions will result in increased social activity. With respect to participation activity levels, we did not see significant differences in course performance, but we did see a consistent trend in overall higher performance by the HIL group (Table 9.7). Further, we did detect some weak positive correlations between intervention activity level and some of the course performance metrics (Table 9.9). Though not conclusive, the trend of higher social activity levels being tied to higher course performance is what we might expect based on Astin's student involvement theory (Astin, 1999). It is possible that the weaker correlations between social activity level and course performance may be due to other confounding factors related to the within-subjects study design.

Comparisons of programming activities indicated that there was little to no effect on programming activity based on interaction with interventions. Though not the main focus of this dissertation, it was posited that we should have seen significant differences based on intervention

www.manaraa.com

activity. The lack of differences indicates that interventions based on programming activity had no effect on programming behaviors; i.e., the two interventions that were most likely to affect programming behavior did not meet the immediacy or relevancy needs of the learner (Johnson, 2010). Another possible explanation is that students may have been reluctant to use the intervention to ask for programming error-related help because they were worried about negative judgements by their teachers and/or peers (Ryan, Pintrich, & Midgley, 2001).

With respect to student attitudes, we saw only one significant difference between groups (Table 9.10). This could indicate that the treatment group attitudes were not impacted by interaction with the interventions. It could also indicate that the intervention period was simply too short for measurable attitudinal differences to materialize or that other attitudinal measures were not sensitive to any potential differences that might have been caused by interacting with the interventions. However, we did see significantly higher gains in peer learning scores in the HIL group, which indicates that the HIL group's willingness to work with others could have been one factor that increased activity and performance throughout the course. This higher peer learning could have also played a part in the significantly higher social participation by the HIL group by increasing enactive and vicarious experiences (Bandura, 1997).

Finally, we saw that the HIL group interaction levels directly translated to an increase in follow up activities (Figure 10.1 and Table 9.12). This again aligns with our expectations that those interacting with the interventions more frequently will also be more active overall.

9.6 Conclusions

The main purpose of this study was to determine if exposure to interventions positively affected student performance and attitudes. The results of this study did provide some support for

www.manaraa.com

the idea that interaction with the interventions increases overall activity levels and student performance Indeed, participants who interacted more with the interventions were more likely to be socially active within the SPE, and to achieve higher performance levels during the course. However, we found little evidence to suggest that course performance, programming behavior and attitudes were affected by a higher level of interaction with the interventions.

Though not conclusive, the analysis presented in this chapter provides some support for the hypotheses. At the same time, it suggests the need for further exploration of why certain interventions elicited interaction and others did not. In the following chapter, I consider the implications of the studies presented in this dissertation for the design of software-realized interventions that positively influence student behaviors and attitudes.

www.manaraa.com

CHAPTER 10

IMPLICATIONS FOR DESIGN

Based on the results of the four empirical studies presented in the previous chapter, an obvious question arises: *how can the design of the interventions be revised to increase their educational effectiveness?*

To address this question, this chapter first revisits the successes and failures of the interventions identified in Studies I – IV. Based on those results, the chapter then identifies potential improvements to the design of the interventions. Finally, the chapter considers the next steps in evaluating the intervention design changes and important questions that should be assessed for future intervention deployment.

10.1 Successes and Failures of Interventions

As with Study IV, discussion will focus here on the data collected in Study II and III due to the nearly nonexistent interaction with interventions in Study I. Figure 10.1 shows that the top three interventions eliciting interaction and follow-up actions were the "*Others Available to Help!*," "*You Are Available!*," and the "*Make a Post! (Assignment)*" interventions. All other interventions elicited a minimal amount of interaction or follow up.

www.manaraa.com

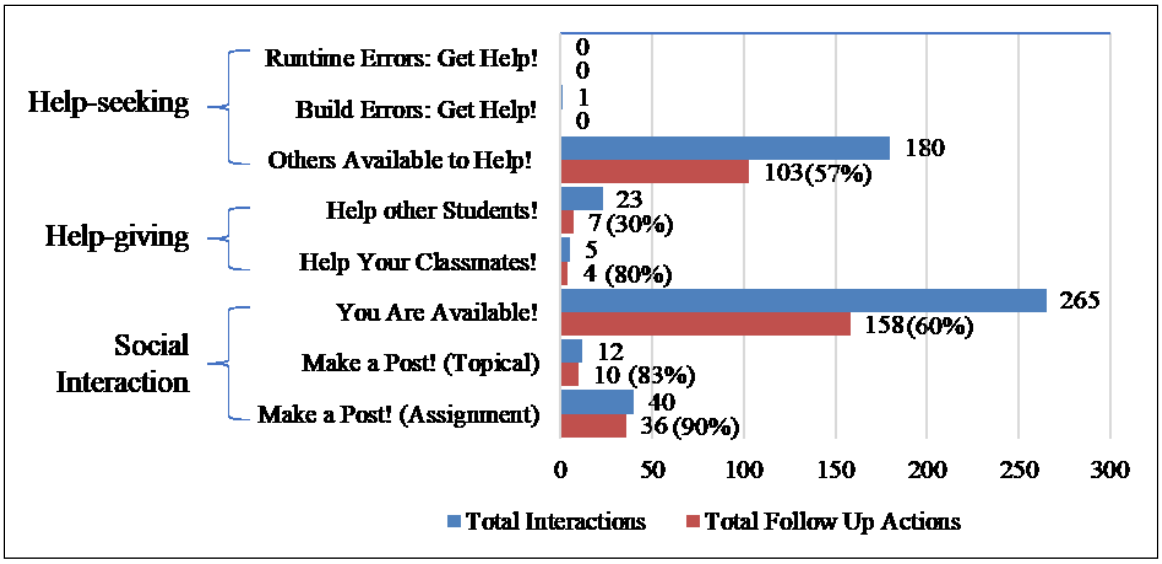

Figure 10.1. Study II and Study III sum of intervention interactions

Looking at each intervention, we see a broad range of follow up rates (between 0 and 90%) with the higher count interactions eliciting a follow-up rate of between 57 and 60%. Figure 10.1 also shows that one intervention in each broad category (Help-seeking, Help-giving and Social interaction) yielded substantially more interaction than the other interventions in the category. This suggests that there is much room for improvement. To explore the potential areas of improvement, I will reexamine the results vis-a-vis the three design dimensions of SPE interventions (Section 3.3); content, timing, and delivery mechanisms.

10.1.1 Content

Recall that the design goal with respect to content was to present only the information salient to the intervention's purpose, and nothing more (Johnson, 2010). Based on the limited feedback provided by participants, it is difficult to determine if this goal was met. Indeed, user feedback such as "Make it a little more prevalent, without being too obstructive…" and "Make it a more prominent feature used in class" suggests the timing and delivery mechanism may have presented a larger barrier to use than the actual content.

www.manaraa.com

While we cannot make firm assertions as to successes or failures of the content of our interventions, we might infer that the more successful interventions provided relevant content (e.g. the top user status-based interventions), while the less frequently interacted with interventions may well have offered relevant content, but had issues with timing or delivery mechanism that prevented that content from being interacted with.

10.1.2 Timing

The *timing* dimension seems particularly relevant in light of the study results. One might classify the interventions developed in this dissertation into two broad categories: *time-sensitive* and *time-insensitive*. Whereas time-sensitive interventions are only relevant during an immediate time-frame, time-insensitive interventions maintain their relevance for an extended period. It is essential that time-sensitive interventions are noticed when they are directly relevant to the learner's immediate goals; they become useless if they are not noticed at the time of need (Johnson, 2010).

An example of interventions that quickly lose their relevance are the runtime and build error interventions. These are generated based on a runtime and build errors, with only the most recent error as the content of the intervention. The delivery is within a 10-minute window of time since the last error (as an attempt to not overwhelm users with notifications). The time at which the intervention has the most impact, in contrast, is the moment at which the user first receives the error—not within 10-minutes afterwards.

In contrast, the most frequently interacted with interventions fall into the time-insensitive category. Indeed, the top two interventions were much more flexible in their relevance: displays

www.manaraa.com

of user statuses, (learners' own and those of others), which can be explored at the user's convenience and still maintain relevance.

Given the observed differences in students' interaction with time-sensitive and time-insensitive interventions, we might conclude that intervention timing played a significant role in the usage or non-usage of the interventions and will consequently need to be redesigned.

10.1.3 Delivery Mechanisms

Table 10.1 shows the most frequent events prior to intervention interaction; interactions directly on the activity feed were the most frequent prior actions. Further, in the exit-survey, students reported most frequently viewing the OSBLE+ suggestions from the OSBLE web site, and not directly from the IDE (see Figure 10.2) suggesting that most of the top activities in Table 10.1 were more likely to occur directly from the LMS and not the IDE. This suggests that our delivery mechanism may need revision; students are likely in the best position to benefit from interventions when they have voluntarily removed themselves from programming activities.

Table 10.1. Study II and III frequency of event prior to intervention interactions

Event Name	Interaction Count
OSBLE Feed Interaction	116
Feed Post*	88
Assignment Submission	83
Code Edit	47
Save	25
Build	17
Feed Reply	5
Runtime Exception	4
Cut-Copy-Paste	2
Debug	1
Reply Marked Helpful	0

**includes "ask for help" event counts*

This observation also motivates a change in the preferred location for intervention delivery. For example, focusing on LMS-based intervention delivery may yield better results than the

www.manaraa.com

current implementation, in which intervention delivery is spread between the IDE and LMS. An alternative plan might be to recognize when a user is browsing the activity feed, using that as a trigger for interventions, rather than displaying interventions right when programming errors occur. Additionally, we might further explore the delivery mechanism on an individual intervention basis to determine which delivery mechanism (or combination) best meets the needs of the learner.

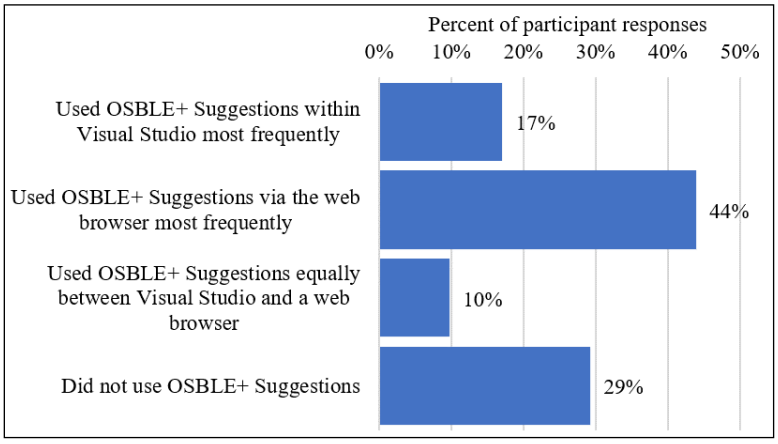

Figure 10.2. Study II and Study III self-reported most frequent intervention interaction location

Finally, we need to consider the *visualization* portion of the delivery mechanism. One possible failure of the delivery mechanism may have been its a lack of visibility (Johnson, 2010; Norman, 2013). In the IDE, users can change the default location or hide the OSBLE+ suggestions window. In contrast, in the LMS, the intervention window is hidden by default. User feedback provides evidence that this may have been a problem. When asked about intervention improvements, some students indicated that the interventions were not sufficiently visible (e.g. “Make it a little more prevalent, without being too obstructive…” and “Make it a more prominent feature used in class”). Thus, design changes are needed to address the issue of intervention visibility.

www.manaraa.com

10.1.4 Conclusions

These successes and failures all suggest deficiencies in the content, timing, and delivery mechanisms of the interventions used in Study I-IV. The timing of programming-based interventions certainly needs revision and we might also suspect that shifting the main intervention focus from the IDE to the LMS could address delivery mechanism shortfalls.

10.2 Considerations for Future Intervention Design and Deployment

The results from Studies II, III and IV suggest that there are weak to moderate positive correlations between activity levels in the SPE and some indicators of course success (assignment grades and overall course grades). Further, two of the three studies showed weak to moderate positive correlations between intervention generation (which is tied to activity levels) and both course success and intervention interaction levels. Therefore, the software-realized interventions studied in this dissertation appear to have some promise that can be harnessed in future work. In this section, I propose a plan to improve the design of the interventions based on what the dissertation studies have taught us. I also suggest changes in intervention implementations based on these findings and reconsider the primary goals of each of the interventions based on what worked and what didn't.

The plan proposes four new intervention categories (Table 10.2). This recategorization of interventions serves a dual purpose: to better focus the interventions on their specific intended outcomes, and to limit evaluation confounds (i.e. interaction of different treatments) by reducing the number of interventions being evaluated at one time, thereby providing more confidence in any resultant conclusions.

www.manaraa.com

Table 10.2. Intervention categories initial revision

Category	Intervention	
1. Programming Behavior	Runtime Errors: Get Help!	Figure 4.12
	Build Errors: Get Help!	Figure 4.13
2. Social Connections – User Status	Others Available to Help!	Figure 4.14
	Help Your Classmates!*	Figure 4.17
	You Are Available!	Figure 4.19
3. Social Interaction – Reflection	Help Your Classmates!*	Figure 4.17
	Make a Post! (Submit)	Figure 4.21
4. Direct Social Interactions – Post-Reply	Help other Students!	Figure 4.16
	Make a Post! (Topical)	Figure 4.20

**Intervention contains elements for multiple categories*

10.2.1 Category 1: Programming Behavior

The first category groups together all programming-based interventions. Both the "*Runtime Errors: Get Help!"* and "*Build Errors: Get Help!"* interventions received the least number of interactions and follow up actions of all interventions (zero and one interaction respectively). As discussed above, these interventions' lack of use appeared to stem from a critical flaw in their timing: they were triggered within a 10-minute window of when the corresponding programming errors occurred, yet they would have had the greatest impact right when the errors occurred—a point at which students may not have been in a good position to receive them.

To increase the effectiveness of these interventions, we might consider the following. First, in the case of build errors, I suspect that prompting students to ask their peers is not the most efficient way to resolve such errors; students can likely resolve the issue more quickly, and with less overhead, simply by thinking about the error for a minute. If they are really stuck, they could then search the internet for the error. Alternately, some students may not be comfortable asking for help for these types of errors due to low self-efficacy or concerns of perceived competence by their peer (Ryan et al., 2001).

www.manaraa.com

In contrast, since run-time exceptions can present more cognitively-complex problems that others in the class may already have learned from, presenting students with social interventions in the case of run-time exceptions may help steer students in the right direction. However, based on my findings and those of past research [e.g. see (Hundhausen & Brown, 2007)], the best timing for such interventions appears to be when students have removed themselves from the flow of programming. For the problem-solving environment studied in this dissertation, this implies that the social interventions should be delivered (a) only in the OSBLE+ LMS, not in the IDE, and (b) only when the learner has switched from the IDE to the LMS, therefore signaling an openness to finding help in the OSBLE+ social environment. However, notice that this approach is in direct conflict with my prior conclusion that a possible cause of the lack of interaction with interventions was due to a disconnect between the immediacy of intervention delivery and the relevance of suggestions to the learner's immediate goals (Johnson, 2010).

10.2.1.1 Proposed Changes

There are two paths that can be taken to address this issue: (1) address the timing and delivery mechanisms of the programming-based interventions as they stand within the IDE by removing the 10-minute window buffer to ensure immediacy and relevance of the intervention or (2) remove the IDE-based component and adjust timing and delivery of interventions to when learners are ready to be engaged in the intervention suggestions in the LMS.

I propose first to assess effectiveness of the programming error interventions through further empirical studies using option (1). The programming-based interventions need to be more accessible in addition to being made immediately available— i.e. they need to be in close proximity to where the user will already be looking for error resolution. In-IDE placement is

www.manaraa.com

currently configured for display in a dockable window with its default position in the lower right, below the activity feed pane. One potential reason this window may not be visible to users is the customizable nature of this pane, as well as its distance from the actual errors. To address this, we would do well to display the intervention directly in the IDE errors message list (Figure 10.3). Doing this will increase the visibility of the intervention, placing it in the user's foveal view when the user first encounters the errors (Johnson, 2010). Further, the time-frame restriction present in other interventions to prevent displaying interventions too frequently will be removed, although we will need to make sure interventions are not considered bothersome and ignored (Teusner, Hille, & Staubitz, 2018). Programming error interventions will be immediately generated for each error, thus increasing the relevance of their content, as recommended by Johnson (2010).

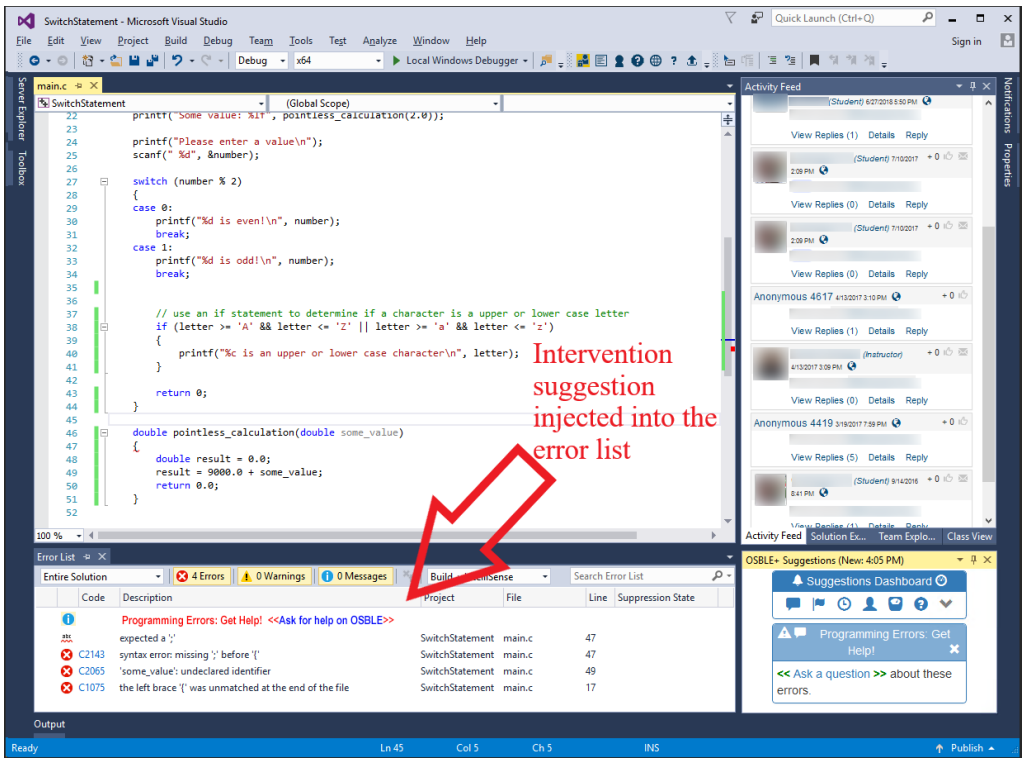

Figure 10.3. Revised category 1: programming error messages in the IDE

www.manaraa.com

An additional adjustment is to present a single programming error prompt, thereby simplifying the number of prompts users will be presented with if they should choose to explore the intervention through the suggestions window (Figure 10.4). Inclusion of a count showing how many other users have received similar errors (runtime or build) should also be added to help stimulate social awareness (Lambropoulos, Faulkner, & Culwin, 2012)

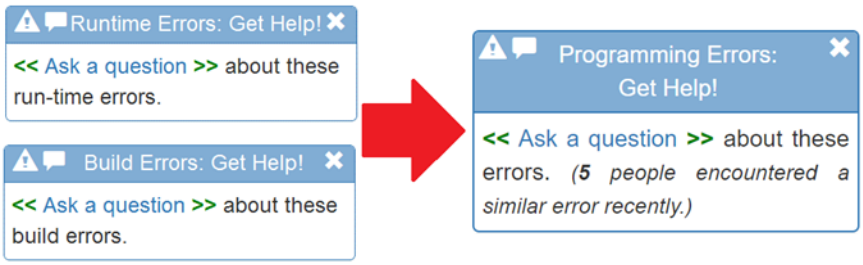

Figure 10.4. Revised programming error prompt

Follow-up empirical studies could provide more insight into the particulars of this proposed redesign. If the proposed change does not accomplish the goal of increasing student interaction, it would be useful to refine the timing of the redesigned interventions, as well as the exact location where they should be presented, by systematically varying those dimensions in additional follow-up studies.

10.2.2 Category 2: Social Connections – User Status

Of the eight interventions, three were focused around the status of users. First, the "*Others Available to Help!"* intervention received the second highest number of interactions and follow up actions, with just under 60% of the interactions eliciting follow up actions. Second, while the "*Help Your Classmates!"* intervention received a minimal number of interactions (the sixth lowest with a total of five) it had the second highest percentage of follow up actions per interaction (80%). Third, the "*You Are Available!"* intervention received the highest number of interactions out of all

www.manaraa.com

studied interventions. Like the "*Others Available to Help!"* intervention, students followed up on about 60% of the interactions it generated.

The rates of interaction and follow up actions generated by these three interventions suggest students may prefer help-seeking behaviors over help-giving behaviors. Such a preference may undermine the intended goal of these interventions: namely, to connect users and thereby increase student involvement (Astin, 1999), opportunities for reflection (Kolb, 1984), and opportunities for vicarious and enactive experiences (Bandura, 1997).

10.2.2.1 Proposed Changes

To refocus the social interaction interventions, I propose to combine the three user status-related interventions into a single intervention. This will not only reduce the overlapping content and the number of entry points into the interventions, but also make it easier to evaluate the effectiveness of the interventions.

To achieve this merging, I suggest that the scaffolded help-seeking template be merged into the generic "*You Are Available!"* intervention. The user status found in both the "*Others Available to Help!" and* "*Help Your Classmates!"* interventions are already present in the "*You Are Available!"* intervention, thus allowing the independent versions of both interventions to be eliminated. The resultant intervention (see Figure 10.6) will contain information focused around connecting users to facilitate social interaction and provide additional scaffolded support for those that may be seeking help. Additionally, displaying a default user status based on activity will make user status more accessible to users. The aim is to promote a minimal level of participation in the feature to help stimulate increased community activity.

www.manaraa.com

To maintain a consistent consolidation of interventions, I propose to merge the three intervention prompts into a single prompt (Figure 10.5). The removal of the overlapping user status portions of interventions should reduce the number of intervention prompts, but also provide additional social awareness through a snapshot of current users online, ideally, prioritized by relevance to the current user, e.g. by past interactions. More changes need to be made to highlight user status features of the LMS; these changes will be discussed later in this section (see 10.2.5).

Figure 10.5. Revised category 2: social connections – user status prompt

To encourage interaction, additional changes could be included in the social interaction process. This requires revisiting the motivation portion of my GAMS framework (Section 3.2). Antin and Churchill (2011) and Nepal et al. (2015), among others, discussed the need for using gamification to affect learner motivation. One method to encourage such interaction could be to integrate a system of badges, reputation points, or achievements for those that ask questions or set their status using the intervention; thus, providing opportunity to positively affect learner motivation to interact by introducing gamified extrinsic motivators to further spread social awareness (Lambropoulos et al., 2012) and academic engagement (Burgess, Riddell, Fancourt, & Murayama, 2018).

www.manaraa.com

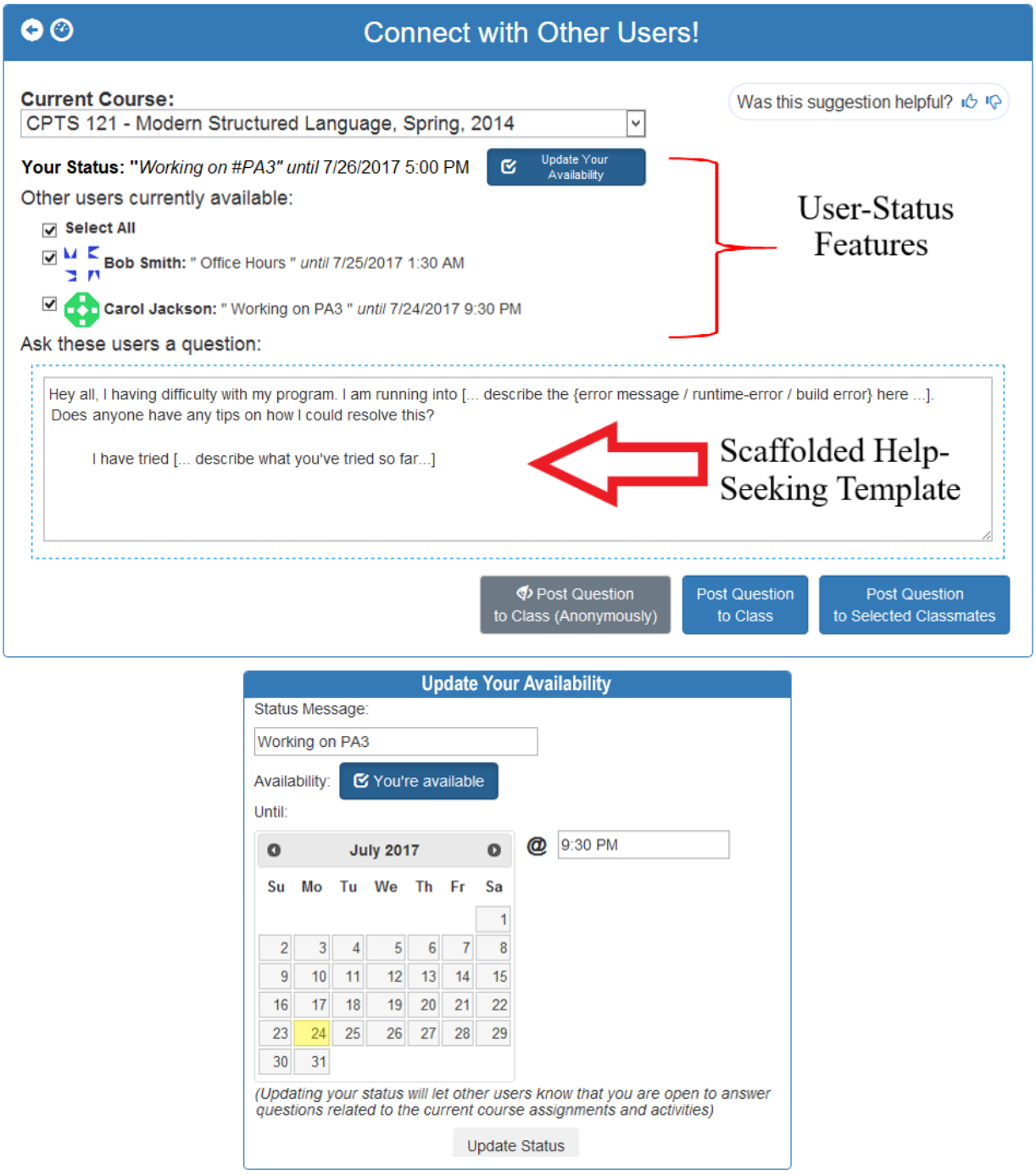

Figure 10.6. Revised category 2: social connections – user status intervention. User-status overview and scaffolded question-asking prompt (top), click-through availability update dialog (bottom)

www.manaraa.com

10.2.3 Category 3: Social Interaction – Reflection

The previously mentioned "*Help Your Classmates!"* intervention also contained a self-reflection component in which users were prompted to (1) reflect on a programming assignment they had recently submitted, motivated by experiential learning theory's supposition that self-reflection is a vital part of the learning process (Kolb, 1984), and (2) post that reflection to the activity feed, motivated by the need to improve interaction and awareness of peer actions in learning environments (Bandura, 1986). I propose to merge the reflection component with the "*Make a Post!" (early assignment submission)* intervention, as they both contain the same content and serve the same purpose; they only differ in the trigger event and minor wording differences. This also has the benefit of eliminating the possibility that both interventions will be generated in some scenarios where it would be confusing—e.g., when a user submits an assignment early followed later by a resubmission not considered early.

Unlike the early submission intervention, "*Help Your Classmates!"* received the third highest number of total interactions (though still six times fewer interactions than the highest interacted with intervention) and the highest number of follow up actions (90%) per interaction. This is likely due to the "early submission" trigger conditions being met less frequently by users; most users do not submit assignments sufficiently early This provides additional reason to merge the two interventions.

10.2.3.1 Proposed Changes

The consolidation of user-status features within the Category 2 intervention (see Section 10.2.2) renders the early submission intervention redundant. The presence of two separate reflection interventions dilutes the intended goal of the intervention by over-exposing learners to

www.manaraa.com

the interventions, potentially resulting in annoyance and loss of intervention significance (Teusner et al., 2018). Thus, I propose to merge the "*Help Your Classmates!"* and "*Make a Post!" (early assignment submission)* intervention components into a new "*Assignment Reflection"* intervention (Figure 10.7). Likewise, I propose to use a single prompt for this intervention (Figure 10.8).

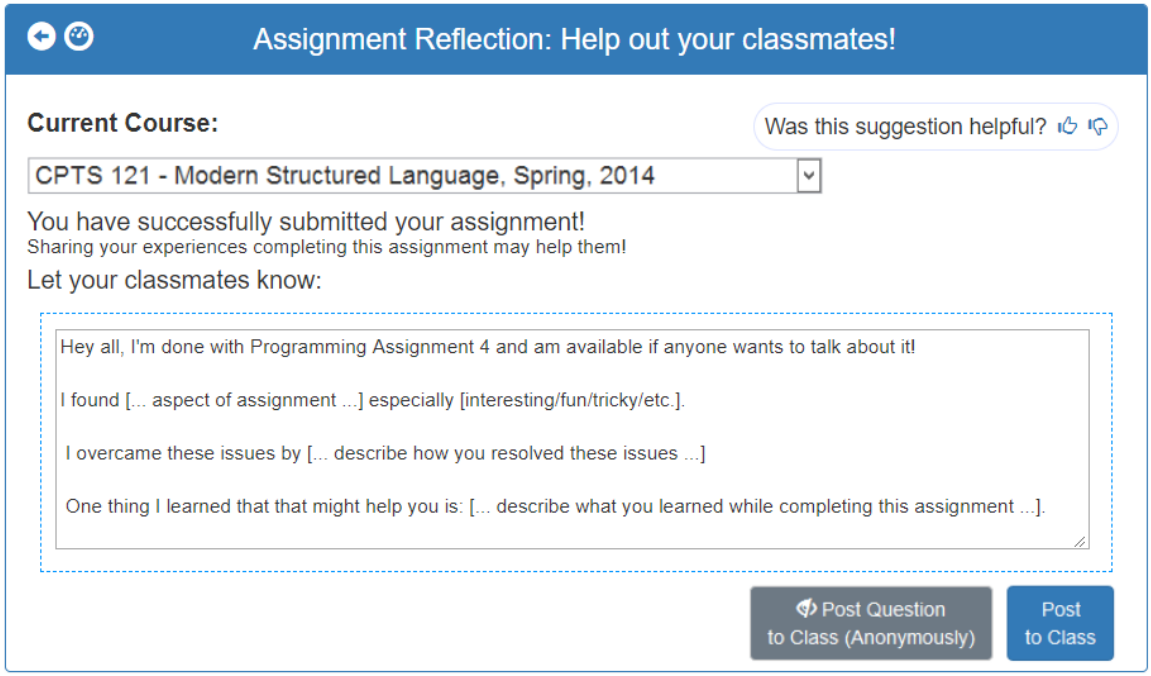

Figure 10.7. Revisions for category 3: social interaction – New reflection intervention

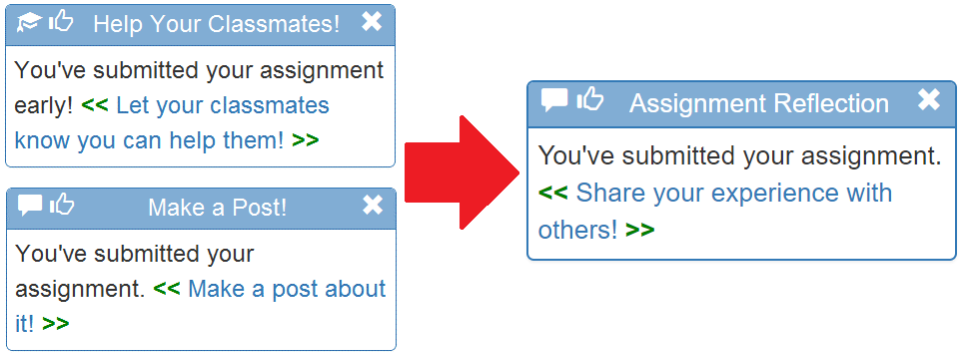

Figure 10.8. Revisions for category 3: social interaction – New reflection prompt

Finally, from a pedagogical standpoint, this category of intervention should be required as part of the programming assignment grade. This will ensure exposure to the interventions and also allow learners to benefit from the reflection process, as encouraged by the experiential learning

www.manaraa.com

theory process as suggested by Kolb (1984), even when they may not have done so without the requirement.

10.2.4 Category 4: Direct Social Interactions – Post-Reply

The final two interventions for Category 4 focus on direct social interaction through posts and replies. One promotes replying to unanswered questions, while the other prompts for general social posting to the activity feed. Both interventions in this category received a low number of total interactions. The "*Help other Students!"* intervention received the 4th lowest number of interactions and the second-lowest follow up action rate (30%), while, the "*Make a Post!" (Topical)* intervention received the 5th lowest number of interactions, with a high rate of follow-up actions (83%).

This low level of interaction with both interventions indicates a need to better entice users to explore the interventions. One approach is to improve social awareness, thus encouraging more social involvement (Astin, 1999) and both vicarious and enactive experiences (Bandura, 1997). The idea behind this is that learners, when made aware of their peers' activity through social awareness cues (Lambropoulos et al., 2012), will be more likely to follow suit with similar behavior [see the social contagion of academic engagement (Burgess et al., 2018)].

10.2.4.1 Proposed Changes

For this intervention category, I propose that the two interventions remain separate but include modified prompts to broaden awareness of social activity (Figure 10.9). The topical social interaction prompt (left) will be amended to include trending topics to give users an at-a-glance idea of current popular topics on the activity feed, while the help-giving (reply-oriented) intervention prompt will include additional social context of recent topics and users who have

www.manaraa.com

asked questions. Both of these changes are geared towards increasing social awareness and involvement (Bandura, 1986; Burgess et al., 2018; Lambropoulos et al., 2012).

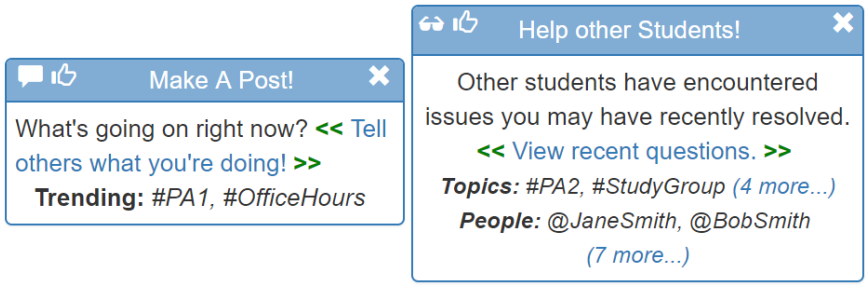

Figure 10.9. Revisions for category 4: direct social interactions – post (left) and reply (right) prompts

The main intervention windows will remain the same as before (see Figure 4.20 and Figure 4.16) as their focus of stimulating post and reply activity remains the same.

Finally, like the user-status intervention category, introduction of badges, reputation points, or achievements could be leveraged to motivate and reward learners to help their peers.

10.2.5 In-LMS Changes

As with the need to adjust the delivery of the programming interventions within the IDE, I propose changes to the LMS in order to make interventions and user status more visible (see Section 10.1.3 motivation). To increase the visibility of user status, I suggest that a "*Users Online*" listing be prominently displayed on the LMS dashboard directly above the Files and Links (Figure 10.10, right). This will provide at-a-glance access to users available to interact with through the OSBLE system. Further, listing of online users should be prioritized to list those that have previous interactions with the current user or close ties to connections. Secondary prioritization could be by similar statuses and user activity level, e.g. users replying more frequently. Additionally, it might be useful to consider extending prioritization of users with similar errors.

www.manaraa.com

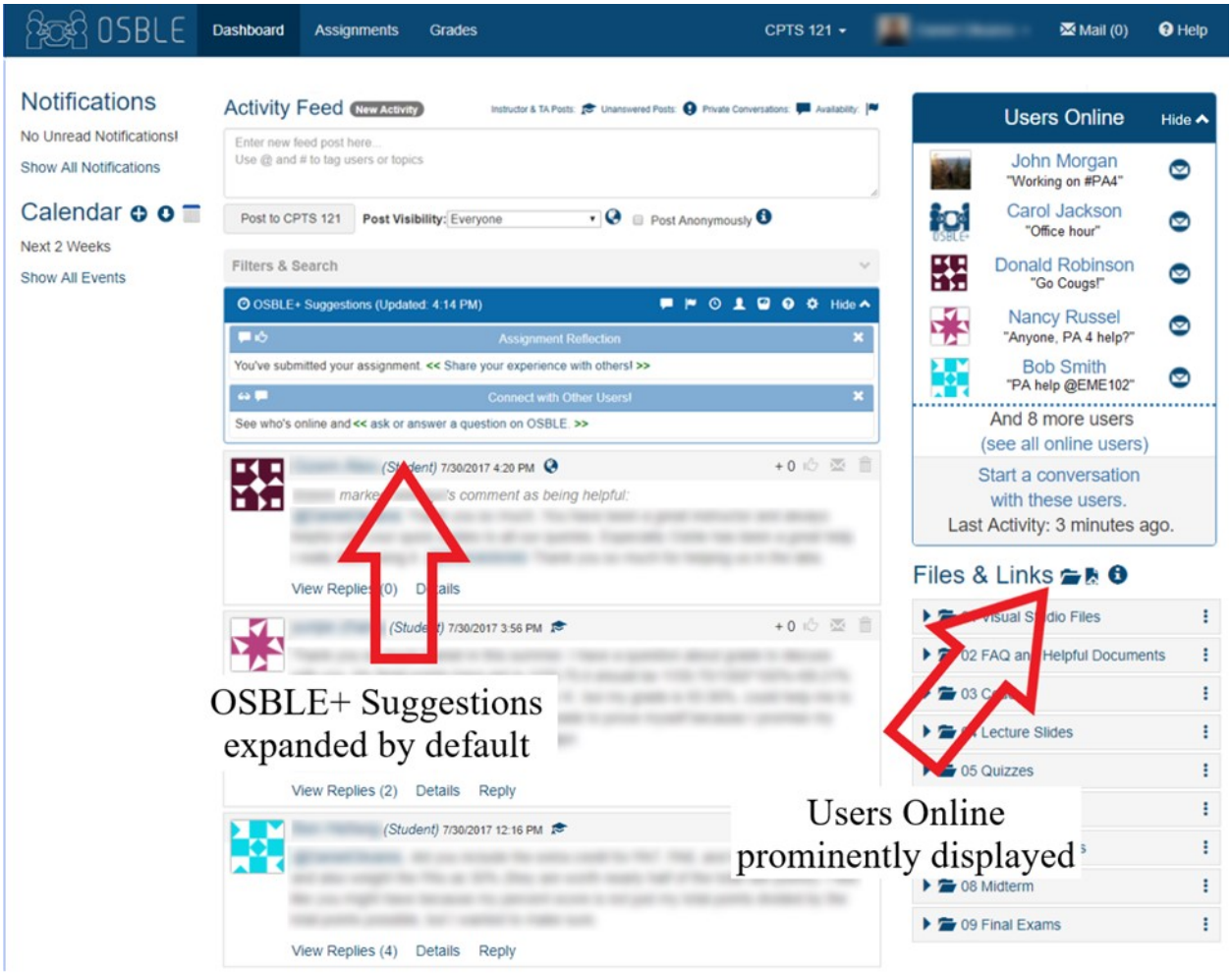

Figure 10.10. Revised user status and OSBLE+ Suggestions dashboard visibility on the LMS dashboard

To increase intervention visibility with the LMS, I propose that the OSBLE+ Suggestions panel remain centrally displayed and be shown by default (Figure 10.10, middle). This will increase the visibility of the interventions and reduce the number of steps required to access them. Users can still choose to collapse the interventions if desired.

A final pedagogical consideration is the possible need to require participation in the social activity feed to stimulate a minimal level of interaction (further discussed in future work, Section 11.3.6). Prior work suggested a minimum post/reply requirement should not have a negative effect impact on results (Carter, 2016) and results of Study II and III did not indicate otherwise.

www.manaraa.com

10.3 Evaluating Effectiveness

To evaluate the redesigned interventions, I propose to perform a series of small scale, short-duration tests for each intervention category, instead of the type of large-scale, semester-long, studies that were previously conducted. The idea is to refine the interventions in each category through iterative testing with a small number of users (3-6 per iteration) before deploying the interventions in a longer-term summative evaluation study.

Table 10.3 summarizes the four categories of interventions described in the previous section. I propose that intervention testing initially be divided into four parts corresponding to these categories.

Table 10.3. Revised interventions and categories for redeployment

Categories	Intervention	
1. Programming Behavior	Program Errors: Get Help!	Figure 10.3, Figure 10.4
2. Social Connections – User Status	Users Online and Statuses	Figure 10.6, Figure 10.5
3. Social Interaction – Reflection	Share How the Assignment Went!	Figure 10.7, Figure 10.8
4. Direct Social Interactions – Post-Reply	Help other Students!	Figure 4.16, Figure 4.20, Figure 10.9
	Make a Post! (Topical)	

10.3.1 Testing Programming Behavior Interventions

This part of the testing plan aims to refine the optimal timing and locations for the programming-based interventions. The triggers for this intervention can also be adjusted. The previous five-error threshold for displaying a programming intervention will likely need to be reduced so that interventions gain relevance by appearing immediately after errors are generated.

Research questions to answer for this phase of iterative refinement are

RQ1: How does changing the timing and delivery of the programming intervention affect user intervention interaction levels?

www.manaraa.com

RQ2: How do intervention interaction levels affect social and programming behavior?

10.3.2 Testing User Status Interventions

This part of the testing plan aims to test the newly revised location for user status in the LMS dashboard (Figure 10.10) and the effectiveness of the merged intervention (Figure 10.6).

It is of interest to determine if the more prominent display of users' online status will have a positive impact on social interaction within the learning community, that is:

RQ3: Does a more prominent user status display stimulate increase social interaction with course users?

10.3.3 Testing Reflection-based Interventions

This part of the testing plan focuses on reflection-based interventions. The goal is to remove the ambiguity between overlapping intervention types and improve the assignment submission reflection intervention.

Research questions that guide this part of the testing plan include the following:

RQ4: Does user reflection increase social interaction among peers?

RQ5: Do students who complete reflections benefit from their reflections?

10.3.4 Testing Interventions to Promote Direct Social Interaction

The final part of the testing plan focuses on interventions intended to encourage direct social interaction (posts and replies to the activity feed). The aim is to determine if these interventions can directly stimulate increased social interaction among the community members:

RQ6: Will exposure to the direct social interaction interventions stimulate increased activity feed posts, replies, and replies marked helpful?

www.manaraa.com

10.3.5 Testing Preferred Intervention Location

After executing the above four-part testing plan that tests each intervention category separately, we also need to determine users' preferred location to access the intervention:

RQ7: Where do users prefer to access intervention notifications (IDE or LMS) for these revised interventions?

This should be done by using both logged data and additional input from users to glean as accurate of an assessment as possible.

10.4 Summary

In this chapter, I explored the implications of Studies II and III for the design of improved interventions. Based on the successes and failures of the individual interventions in those studies, I systematically considered specific ways in which the interventions could be merged, relocated and redesigned to increase the likelihood that they are seen and interacted, with the end goal of having a more significant positive impact on student programming and social behavior. I determined that only two of the eight interventions had a significant number of interactions and follow up actions, while the remaining six interventions had only minimal interaction. Further, based on the lack of impact the interventions had on programming behavior, I concluded that future studies should separate the programming- and social behavior-based interventions into independent categories in order to isolate their potential effects on learner behaviors.

This chapter also revisited the SPE design dimensions in order to better frame the proposed redesign of the individual interventions. Analysis of the study results indicated that most interactions failed to elicit interaction. These results suggest possible problems with the delivery and timing of the interventions, with most users preferring to access the interventions from the

www.manaraa.com

LMS web page instead of directly through the IDE. Due to the low interaction rate, issues with content cannot be fully assessed and should be further explored in future studies.

Finally, based on this analysis, I concluded with a plan to independently test each category of intervention, including the specific research questions that these tests need to address.

www.manaraa.com

CHAPTER 11

CONCLUSIONS

Motivated by learning theories (Astin, 1999; Bandura, 1990; Kolb, 1984; Rotter, 1966; Vygotsky, 1978; Wood et al., 1976) and a projected need for more computer science graduates (Graham et al., 2013), this dissertation has explored the design and effectiveness of automated and personalized interventions delivered by a social programming environment. I hypothesized that exposure to these interventions would lead to better programming and social behaviors as well improved learning outcomes.

Through a series of four studies, I did not find conclusive evidence to support my hypotheses, but did uncover evidence suggesting that, in some cases, there may be a link between intervention interaction levels and success in an introductory computer science course. Specifically, in Study II, III, and IV, those students who interacted with the interventions at a higher level tended to be significantly more active in the learning community and achieved higher grades on some course deliverables. In some cases, students' performance on course deliverables was also weakly to moderately correlated with their levels of interaction.

11.1 Contributions

I now summarize the four primary contributions made by this dissertation.

11.1.1 Theoretically-Derived Framework for Designing SPE Interventions

My primary theoretical contribution of this dissertation, which has already been published in a journal article (Hundhausen, Olivares, & Carter, 2017), is the taxonomy of behaviors (Section 3.1) that can be derived from a continuously-updated stream of learning process data within an

www.manaraa.com

SPE (i.e., an IDE extended with an activity stream). This constitutes the first-ever attempt to systematically describe the space of these behaviors. The taxonomical description provided in this dissertation could form a foundation for future research into learning analytics and educational data mining within computer science education.

In addition, this dissertation contributes the GAMS (Goals, Actions, Motivation, Standing) framework (Section 3.2), which is firmly rooted in relevant learning theory, as well as best programming and learning practices derived from the general education and computer science education literature. By providing a theoretically-motivated framework for intervention design, the GAMS framework also serves as an important foundation for future work in the emerging area of learning analytics for computing education, which shows great promise in effecting positive changes in learners' programming and social behaviors.

The final theoretical contribution of my SPE framework is the design space of SPE interventions presented in Section 3.3. This is the first-ever attempt to systematically characterize this design space for computer science education. These design dimensions served as a vital foundation to the design of interventions used in this dissertation. Further, they provide a foundation for the next steps in refining the interventions proposed in Chapter 10 and set the stage for future research into learning analytics within computer science education.

11.1.2 Novel SPE

A second primary contribution of this dissertation is the key extension of the SPE developed in Adam Carter's dissertation (Carter, 2016). Over the course of multiple years' work, this dissertation designed and implemented the OSBLE+ SPE (outlined in Section 4.1) by extending Carter's OSBIDE SPE in two key ways: (a) by fully integrating it with the OSBLE LMS, thus

www.manaraa.com

making the SPE more accessible to learners and instructors; and (b) by introducing a set of novel interventions (learning dashboards and automated personal interventions) that both present learner and learning community progress and processes, and provide individualized recommendations and guidance toward virtuous learning behaviors. This novel SPE provides the groundwork for future studies into the role and value of a methodology for using data-driven interventions and social programming environments in computing education.

11.1.3 Empirical Study of SPE Effectiveness

A third primary contribution of this dissertation is the quasi-experimental evaluation of SPE interventions, which contributes insights into the role and value of learning dashboards and dynamically-generated notifications in computer science courses. Chapters 5-7 provided some support for the hypothesis that such interventions can increase social interaction within the learning community, while also identifying weak to moderate positive correlations between students' social activity levels and some course performance metrics. Chapter 9 provided stronger evidence in support of the hypothesis that higher levels of interaction with the interventions were positively correlated with course performance. Though the studies by no means provided conclusive results, these studies contribute a better understanding of how to foster learning community interactions and provide additional insight into social and programming intervention design within a SPE. In addition, they provide a compelling basis for future work focused on designing and evaluating the effectiveness of social programming interventions.

11.1.4 Refined SPE Interventions

A final primary practical contribution is a set of SPE interventions whose design has been carefully refined based on the results of the empirical studies presented in this dissertation. Firmly

www.manaraa.com

rooted in both learning theories and empirical results, these interventions provide an additional foundation for future work. The successes of the social interventions highlight the promise of a tightly coupled SPE, while the failures of the programming-based interventions highlight a need to more closely tie interventions to the immediate context in which they are relevant.

11.2 Limitations

This dissertation is limited by the design choices it made, as well as by several other factors behind its immediate control. In retrospect, a flaw in these studies was that they tested a broad range of interventions all at once. This lack of focus may have impaired the ability of the studies to detect and understand the effects of individual interventions. The changes proposed in Chapter 10 attempt to address this shortcoming.

11.2.1 Social Interface

While there are several interfaces into which social interventions can be integrated, this research focused on only one of them—the activity stream used in social media sites such as Facebook. I made this choice for two reasons. First, in prior work, it has shown promise in being an effective tool for promoting social activity (Carter & Hundhausen, 2015). Second, a 2014 poll (Elliot, 2014) of youths age 12-17 shows that Facebook is popular and relevant among future college students. However, a more recent study (Jung, 2018) suggests that younger Facebook users find less satisfaction using Facebook than older users. This indicates that it may be worthwhile to explore alternative social interfaces which may better resonate with future college students (see future work, Section 11.3.1).

There are two additional social interfaces that I might have considered: Stack Overflow's question and answer interface, which allows for the construction of an online reputation, and

www.manaraa.com

Piazza's collaborative interface for building wiki-style repositories of answers to questions. Each of these interfaces has a different way of organizing social content. Table 11.1 shows the dimensions considered important: how top posts are determined, how posts are aggregated, content or reply creation method, level of integration with problem-solving environment, gamification approach, discussion seeds/stimuli, structured or unstructured environment, and source of data.

Because this dissertation focused exclusively on an *activity stream* model integrated into the program solving environment (the Visual Studio IDE), it leaves for future work an exploration of alternate platforms of social interaction in a learning environment. Each of these could be tested individually for their effect on social interaction.

Table 11.1. Social activity interfaces contrast

Dimension \ Social Interface	Activity Stream (This work)	Question & Answer / Reputation (Stack Overflow)	Collaborative Wiki (Piazza)
How top posts are determined	Most recent	User up/down votes	Pinned, most recent
How posts are aggregated	Date	Category, tags, vote total, "accepted" answer	Categories: pinned, week
Content/reply creation	Individual	Individual and collaborative	Individual and collaborative
Level of integration with problem-solving environment	Integrated	Separate	Separate
Gamification approach	None	Reputation, reputation level privileges, badges, achievements	Answer endorsements
Discussion seeds/stimuli	None	Top questions on home page	Topic
Structured / Unstructured	unstructured	unstructured	Unstructured
Source of data	IDE Log + User posts	User posts only	User posts only

11.2.2 Influences Outside of the SPE

As mentioned in section 5.3, it was unavoidable that there were factors outside of the SPE that influenced the effectiveness and usage of the SPE. Additional interactions that occur outside of the SPE include in-classroom discussion, peer-to-peer discussion, or usage of other resources

www.manaraa.com

outside the SPE (e.g. the internet). Unfortunately, these factors, which could have influenced the results of this dissertation's studies, remained outside of my control. Indeed, the dissertation studies could only detect student behaviors that occurred when students interacted within the SPE. This is clearly a significant limitation of the work—one that needs be taken into consideration in all of this work's findings.

11.2.3 Data Gathering Barriers

There were a few barriers to my data collection process. Of those, I briefly discuss the most relevant: course-specific requirements, instructor involvement, student use of the system, software related issues, and administering study procedures.

11.2.3.1 Course-Specific Logistic Requirements

This work was limited by the design and logistics of the courses it studied (CptS 121 at Washington State University). These were, for the most part, out of my control, as they were dictated by the veteran instructor who taught the course. For my dissertation work, it was vital to gain instructor cooperation in requiring the OSBLE+ SPE (which is, in part, a course management system). Though the data collection process of my dissertation did not directly require cooperation by the instructor, usage of the system *did* depend on the instructor agreeing to use the system and to encourage its use by his students.

In order to address this barrier, I worked with the instructor, who ultimately agreed to use the OSBLE+ course management system as a required component of the Study I courses. (In Studies II and III, I served as the course instructor, but was still required to adopt the veteran instructor's curriculum.). Students were required to use the interface as part of the course, though not required to participate in the study. The data logging tools and interface were used throughout the semester

www.manaraa.com

as part of their problem-solving environment. Participant recruitment was voluntary. Despite instructor cooperation in Study I, participant recruitment was only around 60% of the student population for each semester of the study. Study II and Study III were smaller courses with a more invested instructor, which resulted in a 100% recruitment rate.

11.2.3.2 Instructor Involvement and Participant System Usage

In addition to cooperation of the course instructor, recruiting participants *and* having participants use the system was a huge barrier to data gathering. In the case of my dissertation work, there was a huge discrepancy in participant activity levels between Study I, Study II, and Study III. Activity feed participation levels in Study II and Study III, which had a highly invested instructor, were 20 times higher than in Study I, which had a less-invested instructor. In order to test the interventions, I needed a significant number of participants to *actually use* the system. In order to facilitate this participation, I needed a course instructor who was invested in the system and encouraged, if not required, its use as part of the course. Since necessary levels of participation in Study I failed to materialize, I ran Study II and Study III in smaller summer courses over which I, as the course instructor, had much more control. This allowed me to be more involved in the SPE, as well as to require its use as part of the course. Such control and involvement clearly proved crucial in increasing students' use of the interventions.

11.2.3.3 Software Bugs and Defects

Especially in a system as large and complex as OSBLE+ (which has over 30,000 lines of code), issues with software bugs and usability can be a huge barrier to gathering data. In order to prevent this, the system underwent extensive testing to ensure acceptable levels of usability and to minimalize bugs present. This was, however, by no means a definitive answer to the problem.

www.manaraa.com

Usability and software defects were continuously monitored prior to the final deployment and, as part of the development team, I continued to diagnose and resolve issues that arose during the studies. Despite my best efforts, any real or perceived issues with system usability could have influenced study results.

11.2.3.4 Administering Study Procedures

A final data gathering barrier was the occurrence of unforeseen issues that prevented analysis of the anticipated data. Particularly, for this dissertation, mistakes made in administering all surveys in Study II prevented me from performing empirical comparisons of the student attitudinal changes in the Control and Experimental treatments. The only way to address this issue after the fact was to take note of the failure and ensure that it was not repeated in future studies.

11.2.4 Threats to Validity

It is important to consider threats to validity of the data collected and analyses performed on those data, in order to make sure I am making informed conclusions that consider possible confounds. Below I briefly discuss four possible threats to the validity of this research.

11.2.4.1 Statistical

The main concerns regarding the statistical validity of my research relate to *low power* and *unreliable implementation of intervention*. As a starting point for considering statistical threats to validity, I use the categorical definition by Shadish, Cook, and Campbell (2002), who define this category of threat as "quality of inferences that can be made about the correlation/covariation between the treatment and outcome." Most relevant to my research is a ***low power*** threat that can result from a failure to obtain a sufficient sample size. I was constrained by the number of enrolled students and additionally by the number of students who actively participated in the study. As

www.manaraa.com

mentioned in the barriers to data gathering section, I took steps to address this issue in Study II and Study III by requiring the usage of the SPE as part of the course and to incentivize participation in a manner which did not affect the content of the data collected. Despite these efforts Study II and Study III suffered from small enrollments because they were held during the summer semester, which has enrollment restrictions as well as a history of relatively low enrollment.

Additionally, the ***Unreliable Implementation of Intervention*** threat to validity is of concern: i.e., a failure to implement the intervention in a consistent manner across all participants. This threat was minimized by requiring students to use the intervention delivery mechanism (the OSBLE+ SPE) as part of the course and by maintaining parity between course experiences where possible. I was also able to account for individual usage through the collected activity log data and to use such data in my assessment of statistical validity.

11.2.4.2 Internal

This research used a quasi-experimental non-equivalent group design (NEGD). This introduced several internal threats to validity that must be considered. Trochim (2006) indicates that ***threat of selection*** is of concern in NEGD studies. Of primary concern here is that prior differences between groups can affect the outcome of the study. This, unfortunately, could not be controlled for in my studies due to the nature of a self-selecting (i.e. students choose to enroll in the course) participant pool. To mitigate this issue, I did my studies in the same course at the same institution, which increased the likelihood that each course offering would attract similar students. Moreover, I used pre/post differences to compare students in different groups with respect to their attitudes. Despite this effort, there were still numerous differences between groups that cannot be accounted for.

www.manaraa.com

Because I compared two groups of students, I needed to consider the internal validity threat of ***maturation*.** This assumes that the different groups of students will come into the course with differing skill levels and differing rates of learning. According to Trochim (2006), a Control group can address this and help show a natural rate of change. I was able to mitigate this threat by using Control and experimental groups in Study I. For Study II and III though, this is bigger concern due to the within-subjects design: students may have naturally improved their performance over time regardless of the treatment. In consideration of this issue, change over time was used to compare differences between treatment periods and, in Study IV, intervention interaction levels were used to form new groups.

In consideration of possible threats to ***testing*** validity, I wanted to avoid over-assessing individuals. To that end, I limited my use of surveys and assessments to pre, mid, and post course, and I limited repeat participant testing during individual design sessions where possible.

Finally, there is a threat to internal validity with regard to ***experimenter bias***. This is primarily a concern with Study II and Study III, in which the course instructor (me) was highly involved in the running of the studies. Though no intentional bias was introduced to participants during the studies, it is possible that unintentional bias was introduced by the instructor. Future work could address this by replicating the studies with instructors not involved in the creation of the interventions.

11.2.4.3 Construct

Because I collected most data through a single tool (the SPE), my studies were vulnerable to a ***mono-operation bias (one measure)*.** Given the nature of the research, this threat to validity was difficult to avoid. It was partially mitigated through the collection of data from two additional

www.manaraa.com

sources: attitudinal surveys and course deliverables (grades). While the attitudinal surveys were also subject to mono-operation bias, such a bias was reduced based on the observation that the surveys are well-established research instruments.

The ***novel treatment*** threat was also an important consideration. That is, I needed to consider the fact that observed differences between treatments might be due to the novelty of the social programming environment. Unfortunately, it was not possible to specifically control for this issue. However, the fact the treatment spanned a relatively long period of time (a college semester), I believe that this threat was diminished

Finally, ***interaction of different treatments*** was a threat to validity because I implemented multiple interventions but was unable to measure the effects of the individual interventions at the desired level of granularity. The data logging tools were able to tell me which interventions were interacted with, but not the fine details of each user interaction beyond which intervention was used and when the interaction took place. This introduced ambiguity with respect to which interventions were the root cause of significant effects. I was only able to infer a general sense of effectiveness from interaction levels.

11.2.4.4 External

With respect to external threats to validity, I needed to consider the threat of **generalizability**. Will the results of my study be generalizable outside the context of a Washington State University's introductory computer science course? Notably, the significant results detected in this research came from Studies II and III, which had a total of just over 40 participants. While the statistical methods used were appropriate and accounted for the sample size of the studies, one must exercise caution in any attempts to generalize the findings beyond the specific populations

www.manaraa.com

studied. Future studies could address this limitation by running comparative studies at additional universities.

11.2.4.5 Ecological

Ecological validity (Brunswik, 1956) is an external threat to validity that needs to be considered: how *natural* was the study setting. Two factors could have contributed to lower ecological validity: (1) a required minimum participation for Study II and Study III, and (2) the monetary and extra credit incentivization provided for Study I-III. In contrast, the natural setting the studies were performed in gave the studies higher ecological validity. Each of the studies collected data from actual computer science courses during the natural course of a semester. Interventions were introduced as part of the systems that were used for the entire course. Even though the novel SPE used as the data collection tool may have lowered ecological validity, the SPE components likely had a minimal impact on this threat to validity; indeed, many classrooms outside of these studies use online course management systems and the Visual Studio IDE. Thus, students in the studies had experiences that were likely similar to the experiences of students in courses that did not use the SPE.

11.2.4.6 Implementation Fidelity

Finally, I need to consider how implementation fidelity (Fullan & Pomfret, 1977) may have affected my study results. That is, how did the planned use of my interventions compare to the actual use. Based on the analysis and results of Study I, in which instructor participation in the activity feed and promotion of student participation was minimal, this is clearly an issue that needs to be further addressed in future work. Users may not have used all interventions as planned and, in some cases, they may not have been used in the manner intended by the design. These

www.manaraa.com

differences may have influenced study outcomes. Future work might make provide more explicit training to course instructors prior to their use of the interventions in their classes and use additional frameworks or tools to ensure the implementation fidelity of the instructor's use of the SPE, as suggested by the work of Mills and Ragan (2000).

11.3 Future Work

The research presented in this dissertation lays a strong foundation for future work. I conclude by describing six possible directions for this work.

11.3.1 Exploration of Alternate Platforms

As discussed at the start of this chapter and shown in Table 11.1, one avenue for future research is to explore social interaction in different styles of learning environments (activity feed, Q & A, Social Wiki). Each of these could be tested individually for their effect on social interaction and programming behaviors. The results of individual studies could then be compared to identify the ways in which the different environments differ with respect to their ability to promote desirable social interaction and programming behaviors and outcomes.

11.3.2 More Advanced Intervention Triggers

Another avenue for future work is to refine the trigger system that decides when and which intervention to generate for each user. The trigger system used by this dissertation focused on a simple system that relied on counting behaviors and triggering interventions when certain thresholds were met. Future studies that used more advanced algorithms that rely on, e.g. predictive models and states, could determine if these more advanced algorithms could promote increased effectiveness of interventions.

www.manaraa.com

11.3.3 Interaction of Different Treatments

As mentioned in the threats to validity, the interaction of treatments in the within-subjects design of Studies II and III increased the difficulty of interpreting study results. Employing the intervention and category redesigns outlined in Chapter 10 would be one way to address this issue in future research. Reverting to a between-subjects design would be another way. Reducing the interaction between multiple treatments in future studies will allow us to better measure the effects of individual interventions.

11.3.4 Improved Assessment of Intent to Major

A major downside to the method used by this dissertation to determine student intent to major (i.e., persistence) was that we were only able to infer changes of intent to major by comparing students' responses to a question about their intent to major in computer science. A more direct and reliable measurement approach. which should be pursued in future work, would be to observe student persistence in a longitudinal study.

11.3.5 Increased Usage Granularity

One issue that should be overcome in future work is to devise a way to accurately account for where students accessed the SPE's features—in the IDE or on the LMS dashboard. Unfortunately, in this study, we were unable to distinguish these two forms of access and instead had to rely on students' self-reported usage. Unfortunately, as seen in the logged intervention activity cited in Section 10.1, students' self-reported system usage varied significantly from the system logs.

11.3.6 Intervention Use May Need to Be (Initially) Required

Additionally, we found that getting students to interact with both the activity feed and interventions is a difficult task. In order to overcome the initial barrier of use, future work would

www.manaraa.com

benefit from some form of initial required interaction and/or seeded discussion (Miller, Zyto, Karger, & Mazur, 2014) to jump-start system involvement. This is especially important for larger courses where students may be reluctant to make posts and replies if they do not see others making posts and replies. As this dissertation's results underscore, testing the effectiveness of SPE interventions relies on a minimal level of interaction for the interventions to have any chance of positively influencing behaviors and outcomes, and in order to provide a basis for measuring those effects.

11.3.7 Development and Use of Instructor Training Materials

Finally, future work will greatly benefit from the development and use of instructor training materials. Training materials are needed to help jumpstart instructor knowledge of system features and usage; to highlight the importance of having learners use the system and the impact such usage may have on learning outcomes and success; to provide guidelines for active instructor participation in the social feed; and ultimately to ensure that the instructor implements the pedagogy with high *implementation fidelity*. This is especially important considering the huge impact perceived importance can have on instructor buy-in when deciding to adopt new tools and technology (Ni, 2009; Ni et al., 2010). Further, a recent education research report (Lipsey et al., 2012), suggests an important correlation between instructional format, teaching technique, instructor training, and larger treatment effect sizes, thus emphasizing the importance of instructor training and buy-in for future studies.

www.manaraa.com

APPENDIX

A Instructor Survey Report

Programming Metrics

Instructors were asked to rate their perceived importance of the following information about their students' programming activities on a 10-point scale (with 1 being "not important at all" and 10 being "extremely important) and were given nine programming metrics to rate. Of those nine, six (programming constructs used, calendar view of student activity, runtime errors, time-on-task, and debugger use) were most frequently rated 6 and above by 55-90% of participants (see Table 4.2. Instructor survey: programming and social metrics summary) and all but debugger use was given an average score of 6 or higher. Of those six top metrics we can see that time-on-task was the most positively rated metric with 90% of participants rating it a 6 or higher and the highest average rating of 7.68. The next highest average rating was for a calendar view of student activity (6.78).

Social Metrics

Social metrics, overall, did not have high mean ratings. Ratings averaged between 4.78 and 5.48. Despite the low mean, both “Answers given marked helpful” and “Programming answers received” had a mode of 8. Only “Answers given marked helpful” had more than half of the participants (52%) rate 6 or higher (see Table 4.2. Instructor survey: programming and social metrics summary). Thus, although not all social interaction was considered important by instructors, some factors of social interaction were perceived as an important part of assessing student progress.

www.manaraa.com

Open Ended Questions

Instructors were additionally asked five open ended questions:

1. What instructional practices do you presently use to help students who are struggling with programming assignments?
2. What kinds of things would you like to know about what your students are up to as they work on programming assignments?
3. For each item you identified in the previous question, how could you use your knowledge of that item to improve your instructional practices?
4. Do you have any suggestions for other types of metrics or information that you would find helpful in assessing your students' programming progress on individual programming assignments?
5. Please share with us any design ideas you have for an instructor dashboard that could be used to monitor students' programming activities as they work on individual programming assignments.

Each of the open-ended questions was coded into common individual categories. Those categories were then combined into to broader categories (see Table 4.1, Table 11.2-Table 11.6 and Appendix C for the full coded topics counts).

www.manaraa.com

Table 11.2. Combined "desired metrics" category topics (question 2)

Combined Categories	Count	Percent*	Categories
Coding Process	68	42%	code versions, compile frequency, compile history, copy/paste use, correctness, debugging use, progress snapshots, assignment feedback, more detailed process information, test use, task completion, time allocation, time on task, reliance on ide
Learning Process	39	24%	cognitive load, concept knowledge, concept models, aha moments, internal thought process, points of struggle, problem solving approaches, problem understanding, thought process
Design	27	17%	required readings, resources used, starting point
Errors	15	9%	common mistakes, misconceptions, repeated mistakes
Social	12	8%	help seeking behavior, peer interaction

*of combined categories

Coding of these questions (Table 11.2) shows that how students spend their time is important to instructors. “Time on task” is considered the most important individual metric (13% of individual responses), with “time allocation” also important (7% of individual responses) and “how time is spent” (8% of individual responses) also coming up frequently in the open-ended questions showing a combined 28% of responses focusing on what students do with their time. Similarly, if we look back at the Likert responses (Table 4.2), time-on-task was also highly rated with a mean rating of 7.68 across all 15 metrics (programming, social, and instructor) and a mode rating of 7.

Further, instructors showed interest in what students are doing with their time while programming. In question 2, a combined 73% of instructors’ responses (coding process 42%, learning process 24%, social 7%) related to what students are doing. Similarly, 67% of instructors showed interest in having metrics on their students’ progress (coding metrics 45%, social metrics 22%). Instructors also showed some interest in how students spend their time (8%) and their time-on-task (18%).

www.manaraa.com

In contrast to the Likert responses, the importance of social activity to instructors is highlighted much more prominently in the open-ended questions. For question 1, 54% of responses could be considered "social"—meetings and peer interaction that instructors already practice (Table 11.3). For question 2, the social category constituted 8% of responses (Table 11.2), while question 4 had "social metrics" in 22% of the responses (Table 11.4).

Additionally, details of the programming process are seen to be important: 42% of the responses to question 2, 28% of the responses to question 3, and 45% of the responses to question 4 were related to student programming data. This is consistent with the Likert-type rating questions, which showed student programming data as the higher-rated metrics. The top-rated metric was "programming constructs used": most frequently rated a 9 out of 10 for importance with 70% of participants rating it a 6 or higher (Table 11.4).

Table 11.3. Combined "current practices" category topics (question 1)

Combined categories	Count	Percent*	Categories
Meetings	77	46%	group meeting, office hours, lab time help, mentoring, tutors, ta, solo meeting w/instructor,
In-Class intervention	32	19%	in-class activities, in-class discussion, in-class questions, discussions, class discussion, active learning, encourage question asking, example problems,
Additional guidance	20	12%	additional guidance, additional instruction, additional practice work, additional resources, instructional videos, code-walkthrough, concept review, video-lecture, published notes, step by step tutorials, extra days to 'recover',
Online interventions	18	11%	online forum, online materials, online-materials, email, forum answers
Peer interaction	13	8%	pair-programming, peer collaboration, peer comparison, think-pair-share
Design structuring	5	3%	design recipes, informal algorithm to code, verify requirements, written code practice, planning
Feedback	4	2%	assignment feedback, design feedback, weekly assessment, student result data,

*of combined categories

www.manaraa.com

Additionally, details of the programming process are seen to be important: 42% of the responses to question 2, 28% of the responses to question 3, and 45% of the responses to question 4 were related to student programming data. This is consistent with the Likert-type rating questions, which showed student programming data as the higher-rated metrics. The top-rated metric was "programming constructs used": most frequently rated a 9 out of 10 for importance with 70% of participants rating it a 6 or higher (Appendices B and C).

Additionally, details of the programming process are also seen to be important: 42% of the responses to question 2, 28% of the responses to question 3, and 45% of the responses to question 4 were related to student programming data. This is consistent with the Likert-type rating questions, which showed student programming data as the higher-rated metrics. The top-rated metric was "programming constructs used": most frequently rated a 9 out of 10 for importance with 70% of participants rating it a 6 or higher.

Table 11.4. Combined categories for additional design metrics (question 4)

Combined Categories	Count	Percent*	Categories
Coding Metrics	23	45%	code complexity, code similarity, class design, completion of parts, correctness, error locations, number of variables, end time, programming session length, start time, test cases, test cases run, typing code, unit test use, wasted time on task, copy/paste use, correctness of ide suggested fixes, shared code, internet guidance usage
Understanding	17	33%	assignment reflection, historical data, in-class exercise success, patterns of misunderstanding, persistent errors, progress, self-reported frustration, test scores, time in error state, variable misuse, direct expert observation and feedback, how time is spent
Social Metrics	11	22%	comments, help feedback, help seeking data, identity of outside help, informal peer discussion, number of answers (social), number of questions (social), office hours attended, shared debug time, social activity

*of combined categories

For question 3, instructors were asked how they would use their proposed (desired) metrics to improve instruction. The most significant category for interventions leans towards providing some

www.manaraa.com

measure of additional intervention to the learner (Table 11.5) with 64% falling into this category. The largest contributors to this category being interventions which try to address the issues (24%) or an understanding check (15%).

Table 11.5. Combined categories for interventions based on desired metrics (question 3)

Combined Categories	Count	Percent*	Categories
Additional intervention	59	64%	additional discussion, additional examples, additional resources, additional review, address issues, adjust assignments, adjust deadlines, be more explicit in instruction, initiate help, create examples, focus instruction, refine teaching, targeted improvement, encourage students
Check Code	26	28%	check programming methods, check progress, detect integrity issues, measure progress, progress notification, false solution paths, track time management/progress, check understanding,
Share Data	5	5%	share data with other instructors, inform students, prompt break from coding, source of help, compare scores
Preparation	2	2%	force design, knowledge of student prep

*of combined categories

Further, in question 5 instructors were prompted for additional design suggestions on which features might be useful from student programming data. For this question, we see that information about students was the most desired feature for a learning dashboard with 67% of the responses fitting this category (Table 11.6). The highest individual topic reported for this category is a desire to see detailed student progress (~18%).

Table 11.6. Combined categories for instructor dashboard feature suggestions (question 5)

Condensed Categories	Count	Percent*	Categories
Class/Student Statistics	30	67%	class averages, class details, class overview, custom student groupings, customizable stats, heat map (latest changes), help seeking stats, individual details, individual student progress, lines of code counter, live student code, monitor incremental development, pre-coding design, problem > solution history, reports, show student progress (detailed), time on task stats, visualizations, unit test use, completed code prediction
Hardware/Software	10	22%	android and web support, cross-platform daemon, LMS plugin, Markus† integration, MOSS‡ integration, scalable, support java and python, set assignment notifications
Ease of Use	3	7%	easy to use, keep it simple, no "cost" to use
Preparation/Resources	2	4%	predefined milestones, error fix knowledgebase

*of combined categories †http://markusproject.org/ ‡https://theory.stanford.edu/~aiken/moss/

www.manaraa.com

B Instructor Survey

Instructional Practices Related to Individual Programming Assignments in Early Computing Courses

Section 1: Overview of instructional practices relative to individual programming assignments

Introductory prose: "We would like to get an idea of the role of individual programming assignments in your early computing courses you teach. In addition, we would like to know what you do not know about your students' activities as they work on these assignments."

1. How many years have you been teaching computer science courses? <Range from 0 to 30>
2. What early programming courses do you presently teach, or have you taught in the past?
    - CS 0
    - CS 1
    - CS 2
    - CS 3
    - Other (please specify):_____
    - Other (please specify):_____
    - I don't teach early computing courses
3. How many individual programming assignments do you typically assign in your early computing courses?
    - CS 0: <0-12> individual assignments
    - CS 1: <0-12> individual assignments
    - CS 2: <0-12> individual assignments
    - CS 3: <0-12> individual assignments

www.manaraa.com

- Other Course (please specify)_____: <0-12> individual assignments
- Other Course (please specify)_____: <0-12> individual assignments
- Other Course (please specify)_____: <0-12> individual assignments

4. How do you monitor what your students are up to (their programming progress, their struggles, their success) as they work on individual programming assignments? <Open ended>
5. On a scale of 1 to 10, with 1 being "not at all confident" and 10 being "extremely confident," how confident are you in your knowledge of what your students are up to as they work on individual programming assignments?

   <Radio buttons from 1 to 10 on Likert-style scale>
6. What instructional practices do you presently use to help students who are struggling with programming assignments? <Open-ended>
7. What kinds of things would you like to know about what your students are up to as they work on programming assignments? <Open-ended>
8. For each item you identified in the previous question, how could you use your knowledge of that item to improve your instructional practices? <Open ended>

Section 2: Content of Instructor dashboard

Introductory prose: "Imagine that you have access to an *instructor dashboard* that provides you with a window into your students' programming activities as they work on individual programming assignments. We'd like to get your feedback on some ideas about what information such a dashboard might include."

www.manaraa.com

1. On a 10-point scale, with 1 being "not important at all" and 10 being "extremely important," please rate the importance of the following information about your students' programming activities:
    a. Amount of time students are working on the assignment
    b. Calendar view of students' programming activities (so activities can be seen in relation to due date)
    c. Lines of code written
    d. Number of build errors encountered (including the types of errors encountered)
    e. Number of run-time errors encountered (including types of errors encountered)
    f. Programming constructs used in program
    g. Number of comments present in program
    h. Frequency of debugger use
    i. Number of methods written
    j. Average method length
    k. Number of programming questions asked (in an online discussion forum linked to your class)
    l. Number of answers received (in an online discussion forum linked to your class)
    m. Number of answers the student has marked helpful (in an online discussion forum linked to your class)
    n. Number of answers given to other students' programming questions
    o. Number of answers given to others that were marked helpful

www.manaraa.com

2. Do you have any suggestions for other types of metrics or information that you would find helpful in assessing your students' programming progress on individual programming assignments? <Open ended>

Section 3: Design and use of Instructor dashboard

Introductory prose: We'd like to solicit your ideas for an instructor dashboard that could be used to monitor your students' activities as they work on individual programming assignments.

1. On a scale of 1 to 10, with 1 being "not likely at all" and 10 being "extremely likely," rate the likelihood that, if readily available, you would use an instructor dashboard to monitor your students' programming activities as they work on individual programming assignments. <1 - 10 scale>
2. Please share with us any design ideas you have for an instructor dashboard that could be used to monitor students' programming activities as they work on individual programming assignments. <Open-ended>

www.manaraa.com

C Instructor Survey: Additional Open Question Coding Results

Table 11.7. Coding of instructor "desired metrics" categories (question 2)

Desired Metrics Counts	#		#
Time on task	21	Capability	1
Resources used	16	Change between runs	1
Points of struggle	12	code versions	1
Time allocation	12	Cognitive load	1
Progress snapshots	11	Common mistakes	1
Help seeking behavior	10	Compile frequency	1
Internal thought process	10	compile history	1
Design prep	9	Concept models	1
problem solving approaches	9	Help requests	1
misconceptions	7	More detailed process information	1
Unit test use	7	Peer interaction	1
Common bugs	6	Problem understanding	1
Debugging use	5	reliance on IDE	1
Concept knowledge	2	Repeated mistakes	1
Copy/Paste use	2	Required Readings	1
Correctness	2	Starting point	1
aha moments	1	Task completion	1
Assignment Feedback	1	Thought process	1
Total			**161**

www.manaraa.com

Table 11.8. Coding of current instructional practices categories (question 1)

Current Practices Counts	#		#
office hours	19	assignment feedback	1
solo meeting w/instructor	19	code-walkthrough	1
example problems	17	concept review	1
Tutors	15	Design feedback	1
Lab time help	12	design recipes	1
online forum	10	discussions	1
TA	10	Encourage question asking	1
additional guidance	7	Extra days to 'recover'	1
pair-programming	7	group meeting	1
email	5	in-class discussion	1
in-class questions	5	Informal algorithm to code	1
in-class activities	4	mentoring	1
Peer collaboration	4	Peer comparison	1
online materials	3	Planning	1
additional resources	2	published notes	1
class discussion	2	Student result data	1
instructional videos	2	think-pair-share	1
Step by Step Tutorials	2	Verify requirements	1
active learning	1	video-lecture	1
Additional instruction	1	weekly assessment	1
Additional practice work	1	Written Code Practice	1
Total			**169**

Table 11.9. Coding of instructor use of "desired metrics" to improve instruction (question 3)

Improve Instruction Counts	#		#
Address issues	22	Be more explicit in instruction	1
Check understanding	14	Check progress	1
Additional examples	8	Compare scores	1
Targeted improvement	7	Detect integrity issues	1
Refine teaching	5	Encourage students	1
Track time management/progress	4	False solution paths	1
Additional resources	3	Focus instruction	1
additional review	3	Force design	1
initiate help	3	Inform students	1
Check Programming methods	2	Knowledge of student prep	1
Create examples	2	progress notification	1
measure progress	2	prompt break from coding	1
Additional discussion	1	Share Data with other instructors	1
Adjust assignments	1	Source of help	1
Adjust deadlines	1		
Total			**92**

www.manaraa.com

Table 11.10. Coding of metrics categories instructors would like to see in a learning dashboard (question 4)

Suggested Metrics Counts	#		#
How time is spent	4	in-class exercise success	1
unit test use	3	informal peer discussion	1
historical data	2	internet guidance usage	1
identity of outside help	2	Number of answers (social)	1
Progress	2	Number of questions (social)	1
start time	2	Number of Variables	1
test cases run	2	office hours attended	1
assignment reflection	1	patterns of misunderstanding	1
Class design	1	persistent errors	1
Code complexity	1	Programming session length	1
Code similarity	1	self-reported frustration	1
Comments	1	Shared code	1
Completion of parts	1	Shared debug time	1
Copy/Paste use	1	Social activity	1
Correctness	1	test cases	1
correctness of IDE suggested fixes	1	test scores	1
Direct expert observation and feedback	1	time in error state	1
end time	1	Typing code	1
error locations	1	Variable misuse	1
Help feedback	1	wasted time on task	1
Help seeking data	1		
Total			**51**

www.manaraa.com

Table 11.11. Coding of design suggestion features instructors would like in a learning dashboard (question 5)

Design Ideas	#		#
Show Student Progress (detailed)	8	Keep it simple	1
Custom student groupings	3	lines of code counter	1
Customizable stats	2	Live student code	1
LMS Plugin	2	Markus integration	1
Moss integration	2	Monitor incremental development	1
Android and web support	1	No "Cost" to use	1
Class averages	1	Pre-coding design	1
Class details	1	Predefined milestones	1
Class overview	1	problem > solution history	1
Completed code prediction	1	reports	1
Cross-platform daemon	1	Scalable	1
Easy to use	1	Set Assignment Notifications	1
Error fix knowledgebase	1	Support java and python	1
Heat map (latest changes)	1	Time on task stats	1
Help seeking stats	1	Unit test use	1
individual details	1	Visualizations	1
Individual student progress	1		
Total			**45**

www.manaraa.com

D Student Survey and Prototyping Report

Prototype Overview

The interventions were designed to be within an "OSBLE+ Community" window where users were presented with an array of interventions generated using the GAMS model to guide their designs:

- The ***Goals*** dashboard (Figure 11.1) is designed to situate the user around their *Goals and Actions* based on Rotter's (1966) *locus of control* theory.

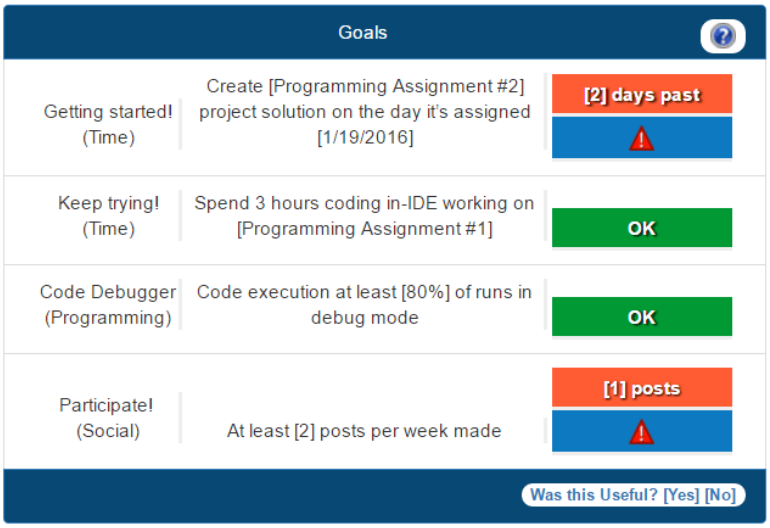

Figure 11.1. Goals overview intervention

- The ***Community Standing*** and ***Goals at a Glance*** modules (see Figure 11.2 and Figure 11.3) are centered on the *Standing* aspect of the GAMS model. Per Bandura's (1997) and Astin's (1999) social learning theories, their goal is to make users aware of where they are in relation to their community and goals and enable them to observe others' learning behaviors within the community and access their own learning progress in that same community (thus supporting *vicarious* and *enactive experiences*).

www.manaraa.com

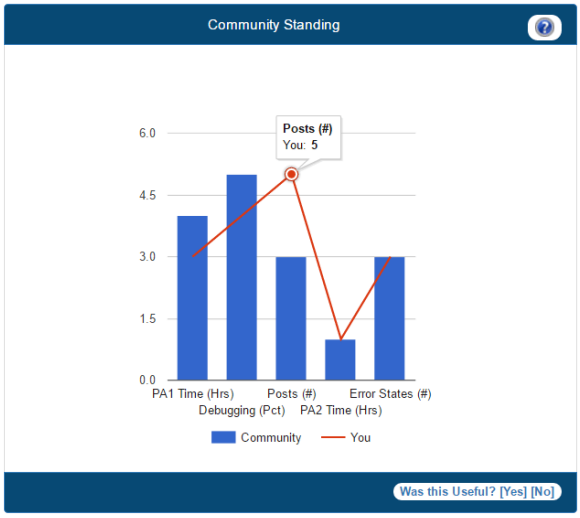

Figure 11.2. Community (where do I stand?) intervention

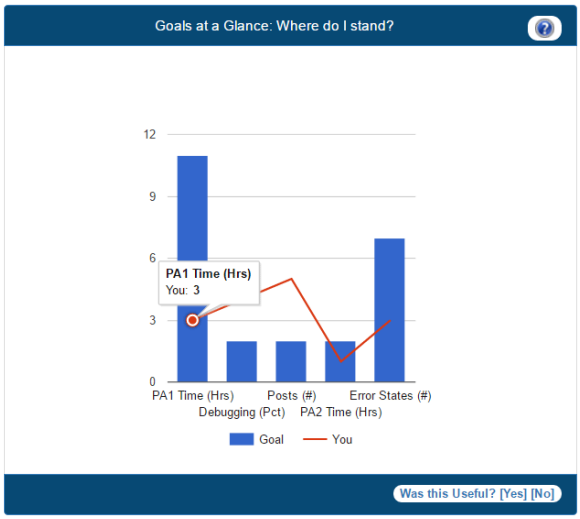

Figure 11.3. Personal goals at-a-glance (where do I stand?) intervention

- The ***Who's Online?*** module (see Figure 11.4), though not fitting explicitly into the GAMS model, might be seen as contributing to *motivation* by promoting awareness of community activity and providing a means for users to initiate social interaction through private messages or the activity feed.

www.manaraa.com

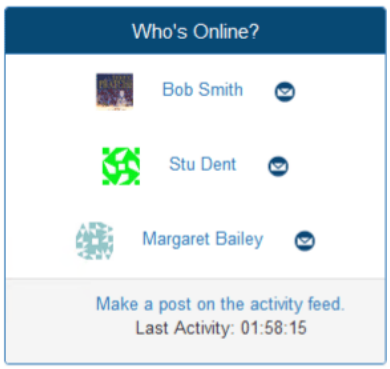

Figure 11.4. Who's online? (community awareness)

- The ***Recent Activity*** module (Figure 11.5), though not explicitly part of the GAMS model, can be seen be as supporting both *Motivation* and *Standing* by letting learners know where they stand in relation to their community within the context of current activity) and also by providing *motivation* to take action (e.g. to post a message to the activity feed based on recent activity).

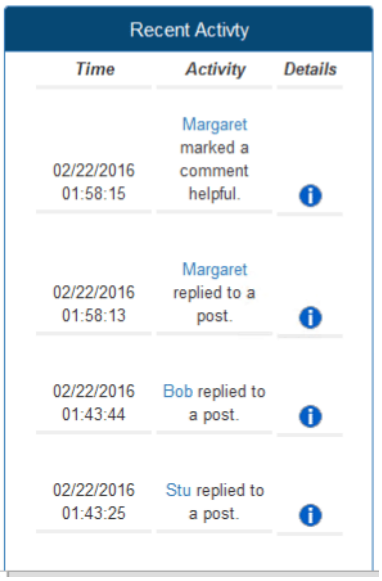

Figure 11.5. Recent activity (community awareness)

Figure 11.6 shows an overview of the above interventions in the context of the Visual Studio development environment as students would see them. Upon starting Visual Studio, users see the activity feed, from the OSBLE+ learning dashboard, docked on the right-hand side in the same

www.manaraa.com

pane as the Solution Explorer tab. In the main window tab, an "**OSBLE+ Community**" pane is prominently displayed along with the most recent intervention information.

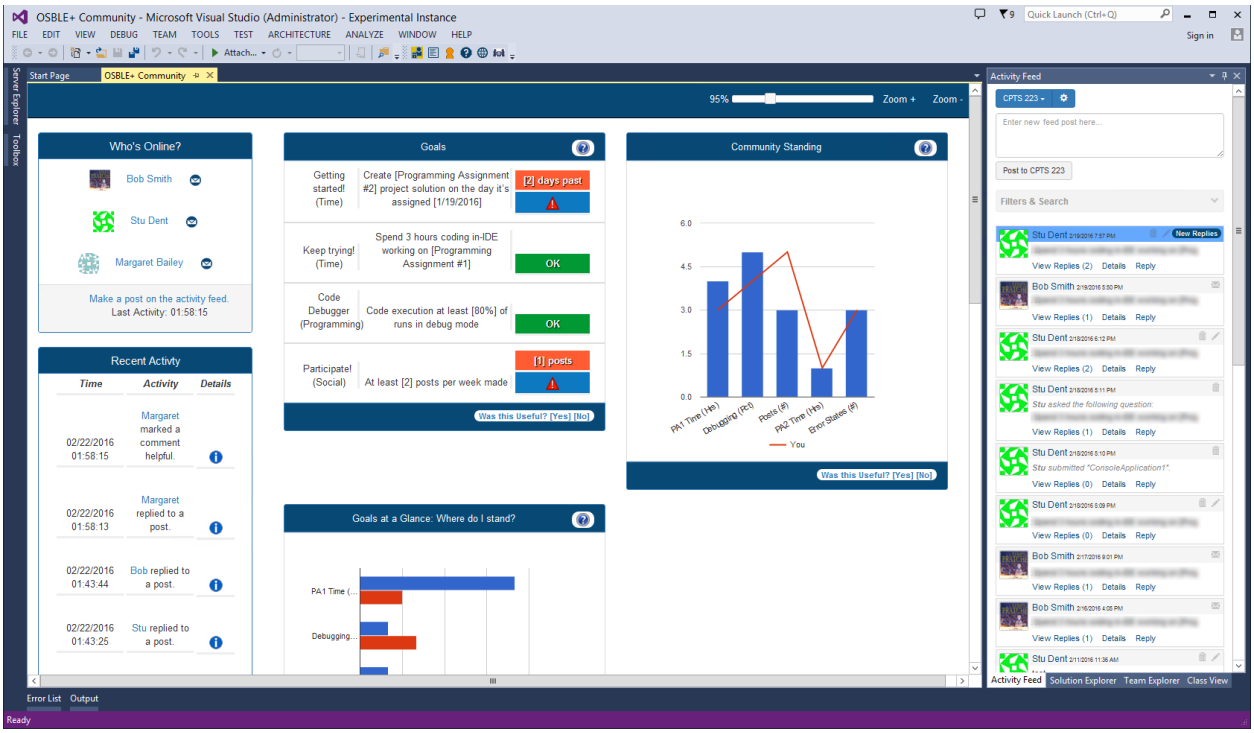

Figure 11.6. OSBLE+ Community Social Interface

Procedure

Participants were walked through a series of questions (see Appendix E). First, they were shown images of the individual intervention panels in order and asked to click a feature they would most likely use and to describe how they expected the interface to respond, what they believed that feature would do, and to rate their perceived usefulness (1, not at all useful to 7, very useful) of the intervention. Click interactions were tracked for each intervention along with their responses. Participants were next shown the overview of all community panels and were then asked to rank the features in order of most and least important to them and to rate the likelihood (1, not likely at all to 7, very likely) and frequency (never to 3+ times per day) with which they expected they would use the intervention.

www.manaraa.com

Summary

Results suggest a moderately positive reception of these interventions and their use. Overall, we saw a positive response from participants, with most responses on average being at worst neutral to positive (average responses 4-5 out of 7); the only exceptions being the frequency of use responses for the ***Community Standing*** and ***Goals at a Glance*** interventions though this can be expected as frequency of use uses a time-scale instead of a negative to positive rating. Additional open-ended feedback indicated that participants thought the ability to "*connect through the IDE would greatly benefit the class*" because it could "*help you over your hump in a part of a program that may be just a simple problem.*" The results discussed here were used to inform design decisions for the following intervention studies.

www.manaraa.com

E Prototyping Survey (Spring 2016)

An initial design prototype survey given to students with minimal or no OSBLE+/Visual Studio Plugin experience.

Part 1: Design Feedback

The following section will provide you with component images from a community dashboard integrated into the Visual Studio IDE. You will be asked to rate usefulness of the component and provide design feedback as a CptS 121/122 student. Please use the provided printouts to sketch any design suggestions you may have.

[Screenshot of a 'community' module. Modules shown: "Who's Online," "Recent Activity," "Goals Overview," "Goals at a Glance," and "Community Standing," and All components together in the Visual Studio IDE]: Click on the image "feature" the you'd use most.

Figure 11.4. Who's online? (community awareness), Figure 11.5. Recent activity (community awareness), Figure 11.1. Goals overview intervention, Figure 11.3. Personal goals at-a-glance (where do I stand?) intervention, Figure 11.2. Community (where do I stand?) intervention, Figure 11.6. OSBLE+ Community Social Interface

[For the combined image, the following text was displayed]: Looking at all components together, integrated into the Visual Studio IDE, please answer the following questions. (Right click image and select view/open in new tab/window to see image at full resolution) Click on the image "feature" the you'd use most.

- What "feature" did you click and what action/response did you expect to see from the interface?

www.manaraa.com

- How useful would you rate the above component? (rated 1-7 not at all useful to very useful)
- How likely would you make use of the above component? (rated 1-7 not at all likely to very likely)
- How frequently would you use of the above component?
    - Never, 1-2 times per semester, 1-2 times per month, 1-2 times per week, 3+ times per week, 1-2 times per day, 3+ times per day
- Which feature (or features) of the above component are MOST important to you?
- Which feature (or features) of the above component are LEAST important to you?
- What would you change about or add to the above component? Reply below and/or use the provided printout to sketch any changes.
- How interested would you be in having the above component integrated into your programming environment (e.g. the Visual Studio IDE)? (rated 1-7 not at all interested to very interested)

[For the combined image, the following text was displayed instead of the previous question]:

- How interested would you be in having the above dashboard integrated into your programming environment (e.g. the Visual Studio IDE)?

Please rank components by usefulness (drag and drop) where an item ranked 1 is considered "more useful" compared to an item ranked 2 and item ranked 2 is "more useful" than an item ranked 3, etc...

www.manaraa.com

_____ Who's Online? _____ Recent Activity _____ Goals _____ Community Standing

_____ Goals at a Glance _____ Activity Feed _____ Other component suggested

_____ Other component suggested

Please rank components by usefulness in the category of would or would not use (drag and drop) where an item ranked 1 is considered "more useful" compared to an item ranked 2 and item ranked 2 is "more useful" than an item ranked 3, etc...

USEFUL components	I would NOT use
_____ Who's Online?	_____ Who's Online?
_____ Recent Activity	_____ Recent Activity
_____ Goals	_____ Goals
_____ Community Standing	_____ Community Standing
_____ Goals at a Glance	_____ Goals at a Glance
_____ Activity Feed	_____ Activity Feed
_____ Other component suggested	_____ Other component suggested
_____ Other component suggested	_____ Other component suggested

Part 2: Open Discussion / Any Further Feedback

Thank you for your time! If you have any further feedback, please leave your comments below. This feedback can be for the survey you just completed, the designs you've seen, or your usage of OSBLE+ and the Visual Studio Plugin if you have used either. You will have another opportunity towards the end of the semester to give additional feedback on your usage of OSBLE+ and the Visual Studio Plugin. **<Open-ended>**

www.manaraa.com

F Design Iteration 1 Report

Evaluation Study

Prototype Design Feedback Sessions

During this session, 5 participants were recruited to answer a brief survey and answer questions related to their help-seeking and help-giving behavior during the programming process. Participants were age 18 to 27 (mean 21), male, consisted of 2 computer science and 3 non-computer science majors. All participants had previously taken the CptS 121 offering at WSU as part of their major requirements. This small pilot study was designed to elicit initial design feedback, gauge student opinion of the designs, and elicit their help-seeking, help-giving, and social behaviors while programming.

Participants were given a programming scenario followed by three hypothetical events: (1) a build error, (2) a runtime error, and (3) an additional syntax, build and/or runtime error. For each of the events we asked the user to assume that they don't immediately understand the cause of the error. Each event was followed by questions asking participants to rate their help-seeking frequency and likelihood of seeking help from instructors, teaching assistants, and classmates as well as their approach to solving the problem provided for the scenario. All rating questions followed a 7-point Likert scale with 1 representing never or not likely at all and 7 representing always or very likely. The three help-seeking events were followed by a fourth event in which participants were told that they had resolved all previous errors and that their code performs as expected. Participants were then asked to rate their help-giving and social behaviors: sharing coding success with others, helping others with coding issues, frequency of helping others, and a rationale for their help-giving behaviors.

www.manaraa.com

Following the above help-giving, help-seeking, and social interaction questions, participants were asked to revisit the same scenario and events but were also shown early prototype designs of the newly refocused interventions. Prior to the individual prototype window questions, participants were shown an overview of the Visual Studio IDE with interventions situated in the lower right of the IDE (see Figure 11.7). For each of the event scenarios, participants were asked to describe what the prototype window's purpose was and to rate their perceived usefulness and frequency of use. Additionally, participants were asked for design feedback and likelihood of the prototype helping them resolve the hypothetical issues.

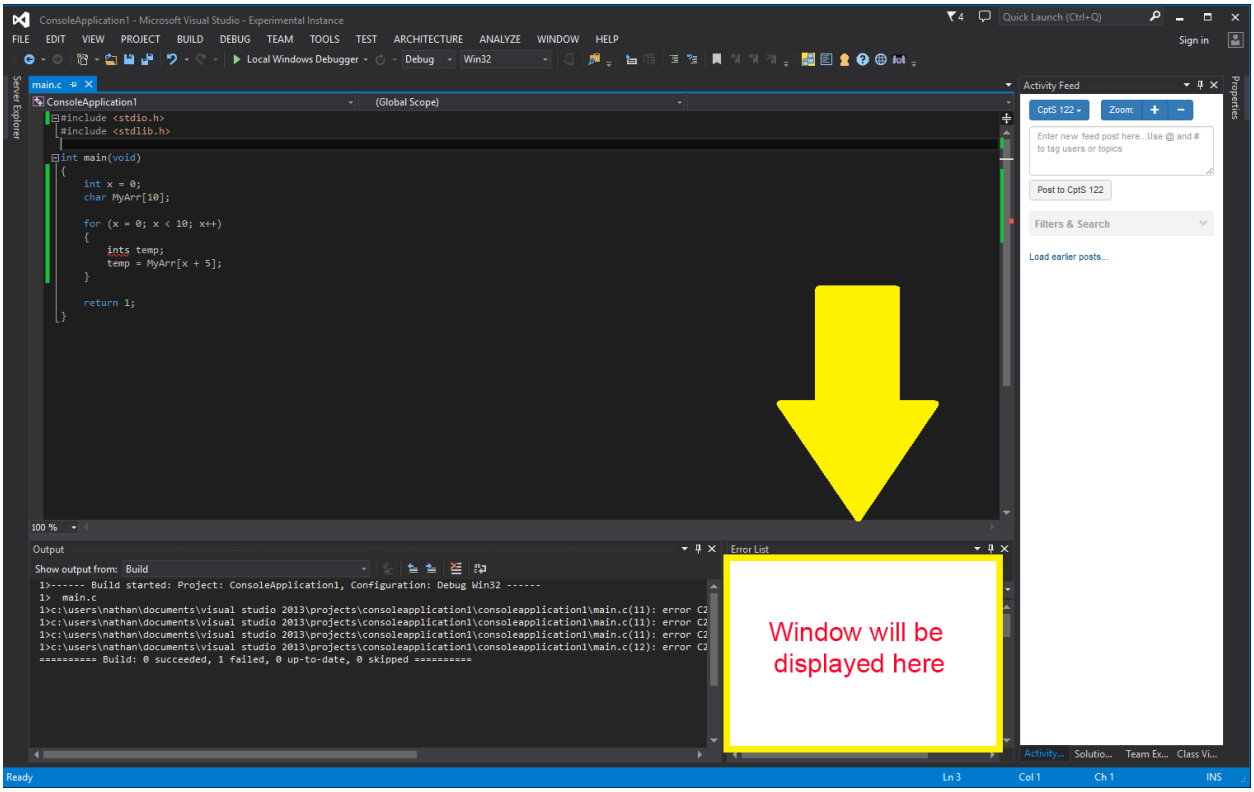

Figure 11.7. Prototype intervention location

www.manaraa.com

Help-Seeking Interventions

The first three intervention designs were focused on learner help-seeking behavior (events 1 through 3) and were designed as prompts to be displayed in the above described location (Figure 11.7).

For event 1 (build error) participant responses indicated an even spread in frequency of asking for help with one response each between the frequency ratings of "rarely" (less than 10% of occurrences) to "Usually" (about 90% of occurrences) and no responses for "never" or "every time" with 60% of the responses indicating they would ask for help at least half. Responses also indicated that participants were most likely to ask their TA for assistances (M=4.8) and then instructors (M=4) but least likely to ask their classmates for help (M=3.2) when running into a build error.

When presented with the event 1 scenario and the intervention window (Figure 11.8), all participants were easily able to determine the intended usage of the window (post coding questions to their class). Additionally, participants responded positively to the usage of the intervention indicating that, on average (M=5.2), they would make use of the intervention at least 1-2 times per week or greater (60% of responses) and that the intervention was likely to help them resolve their issue (M=5.8).

www.manaraa.com

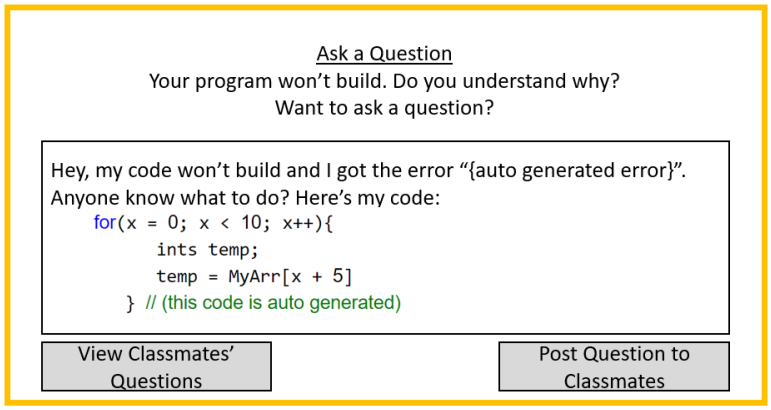

Figure 11.8. Build error intervention prototype (event 1)

Event 2 (runtime errors) participant responses indicated a slightly negative skewed spread in frequency of asking for help with 60% of responses falling below the 50% frequency rating of asking for help. Like the event 1 responses, no responses indicated "never" or "every time" but only 40% of responses indicated they would ask for help at least half of the time. Again, like event 1 responses, responses indicated that participants were most likely to ask their TA for assistances (M=4) and then instructors (M=3.4) but least likely to ask their classmates for help (M=2.4) when running into a runtime error.

When presented with the event 2 scenario and the intervention window (Figure 11.9), all participants were again easily able to determine the intended usage of the window (post coding questions to their class) and also inferred that their problem was similar to other classmates' problems. Participant responses for event 2 were more neutral in nature, though still skewed positive: Ratings of perceived intervention usage averaged 5 and responses indicated they would make use of the intervention at least 1-2 times per week or greater (60% of responses) and that the intervention was likely to help them resolve their issue (M=5.2).

www.manaraa.com

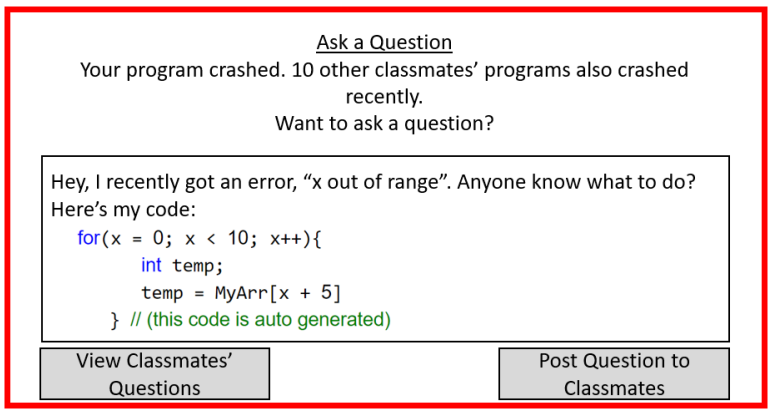

Figure 11.9. Runtime intervention prototype (event 2)

For event 3 (mixed error scenario), participant responses indicated a negatively skewed spread in frequency of asking for help, though in this case there was a higher average response (M=4) than in the event 2 scenario. Still, 60% of responses fell below the 50% frequency rating of asking for help. Similar to the event 1 responses, no responses indicated "never" but 1 participant did indicate that they would ask for help "every time" when they had multiple undiagnosable problems with their code. Like event 2, only 40% of responses indicated they would ask for help at least half of the time. Following the pattern seen in the first two events, responses indicated that participants were most likely to ask their TA for assistances (M=4.6) and then instructors (M=4.2) but least likely to ask their classmates for help (M=2.4) when multiple undiagnosable problems with their code.

When presented with the event 2 scenario and the intervention window (Figure 11.10), again, all participants were easily able to determine the intended usage of the window (post coding questions to their class) and also inferred that their problem was similar to other classmates' problems. Participant responses for event 2 were more neutral in nature, though still skewed positive, with intervention usage averaging 5 and they would make use of the intervention at least

www.manaraa.com

1-2 times per week or greater (60% of responses) and that the intervention was likely to help them resolve their issue (M=5.2).

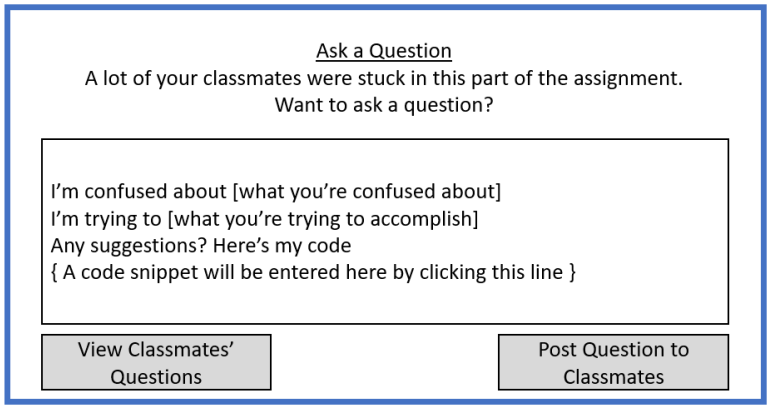

Figure 11.10. Help-seeking intervention prototype (event 3)

Help-Giving and Social Interaction Intervention

The last prototype intervention design was focused on stimulating help-giving and social interaction behaviors and consisted of a slightly different set of questions. Participants were presented with a scenario in which they had fixed their previous errors and the program now ran as expected. Four questions inquired as to their likelihood of sharing their coding success with others, likelihood of helping others, the frequency with which they have historically helped others, and their rationale for helping (or not helping) others.

Event 4 (help a classmate) participant responses also indicated a slightly neutral to positive skewed spread, with 80% of the responses indicating that they were neutral to very likely to share their coding success with others, with a 50-50 split between neutral and very likely (M=4.8). The likelihood of helping someone else with a coding issue was also skewed positive, with 80% of responses indicating a 5 or higher rating and all responses lower were neutral rating. Historical helping of others on coding problems was split down the middle, with 40% of responses indicating that they have helped with similar problems 70% of the time or greater and 40% only rarely helped

www.manaraa.com

(less than 10% of the time) with the remaining responses falling in the neutral category of helping only 50% of the time. Rationale for these help rates showed a lack in confidence in their ability to help and not feeling like others needed help ("*It depends … on how confident I am*," "*most of them probably don't need my help*," "*I have very rarely been asked for assistance*") but also showed a willingness to help if they thought they could ("*If they ask for help I'll try and help as much as I can*," "*I would try to help them*," "*I am always ready to assist in whatever way I can*").

When presented with the event 4 scenario again and the intervention window (Figure 11.11), all participants were easily able to determine the intended usage of the window (help others who might have similar problems). Participant responses for event 4 here were again split between positive and negative, with a skew toward negative. Intervention usage averaged 5.6 with 80% of responses providing ratings of 5 or higher. Participants rated frequency of use more in the neutral to negative range with only 20% indicating that they might use the intervention 3 or more times per week and the remaining responses split evenly between 1-2 times per week and month (M=3.8.)

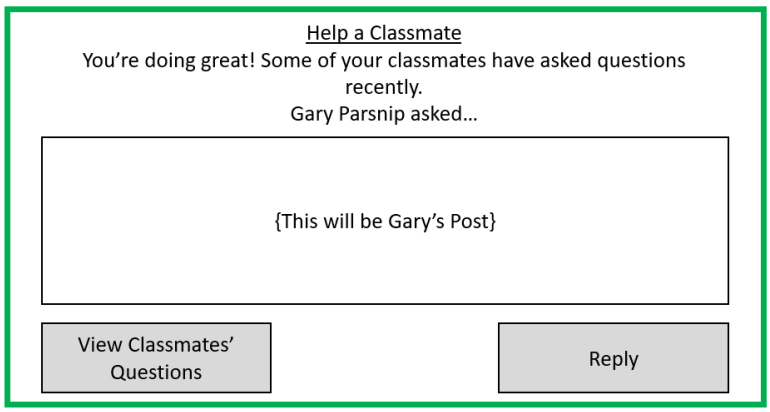

Figure 11.11. Social interaction intervention prototype (event 4)

www.manaraa.com

G Design Iteration 2 Report

Interactive Intervention Walkthrough

For the interactive portion of the pilot study, participants were given 5 tasks to complete. For each task, they were given a programming problem scenario followed by a series of events which gave directions to inspect the intervention suggestion prompts and to "think aloud" while describing how they would interact with the suggestion window. Participants were then directed to click the suggestion link and again describe how they would interact with the intervention window that appeared.

Task 1

Like the summer 2016 prototype session, the first task focused on ***help-seeking*** behaviors via 3 event scenarios; build error, runtime error, and help is available. This task directed participants to "*Ask a Question!*" and "*Get Help!*" interventions for the first two events (Figure 11.12 and Figure 11.13). For the third event, participants were directed to a "*Help is Available!*" prompt and the resultant window showing users available and a scaffolded question prompt (Figure 11.14).

www.manaraa.com

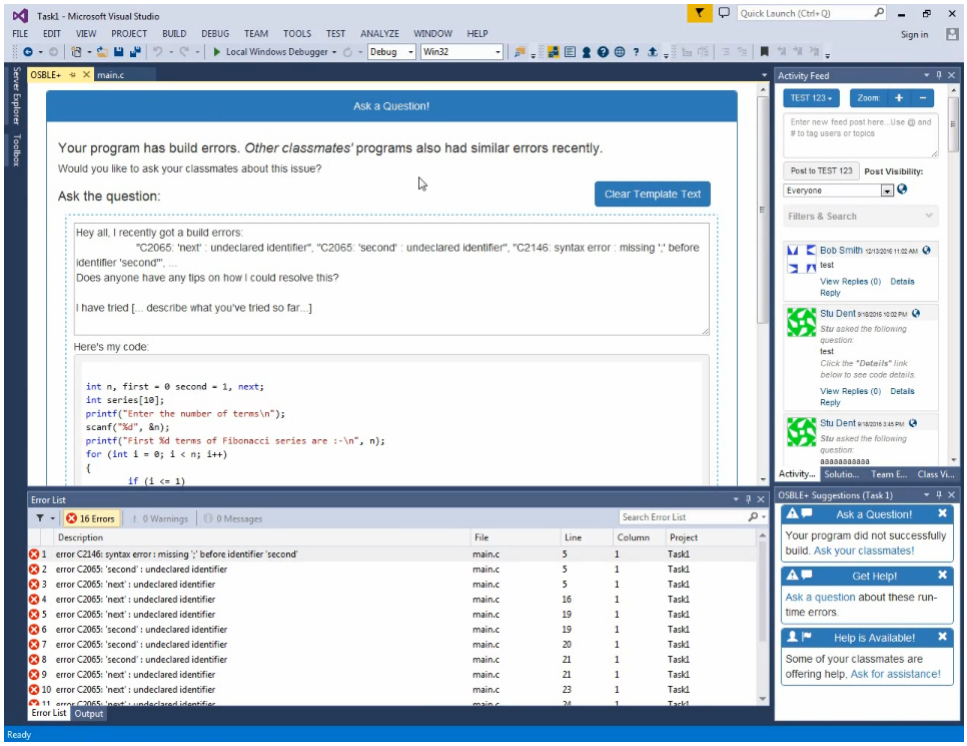

Figure 11.12. OSBLE+ Suggestions final prototype task 1a

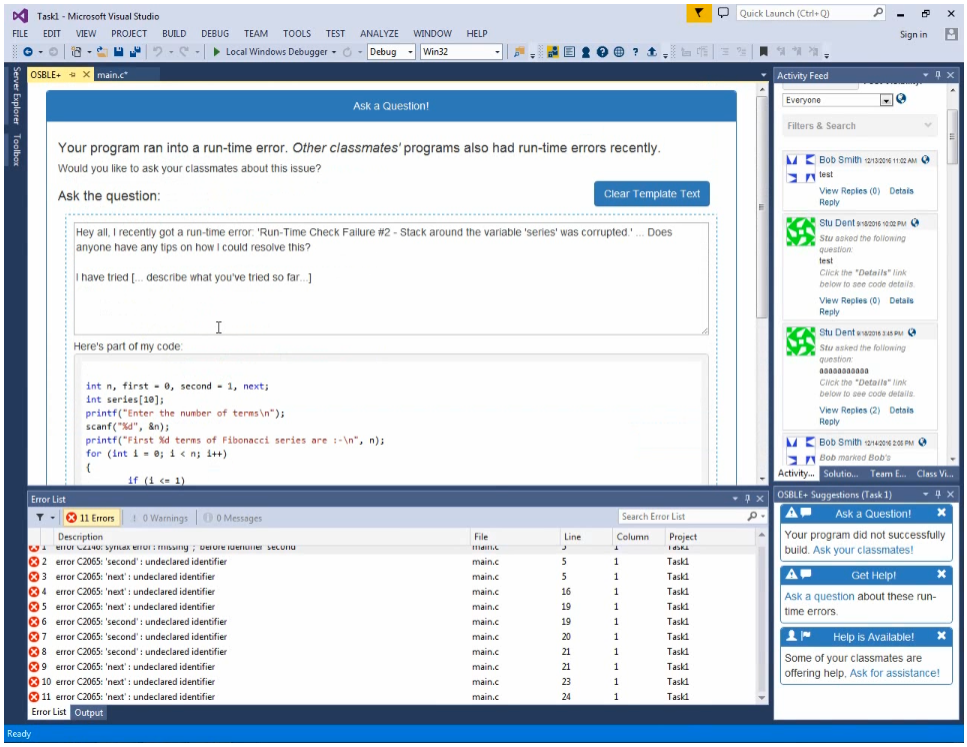

Figure 11.13. OSBLE+ Suggestions final prototype task 1b

www.manaraa.com

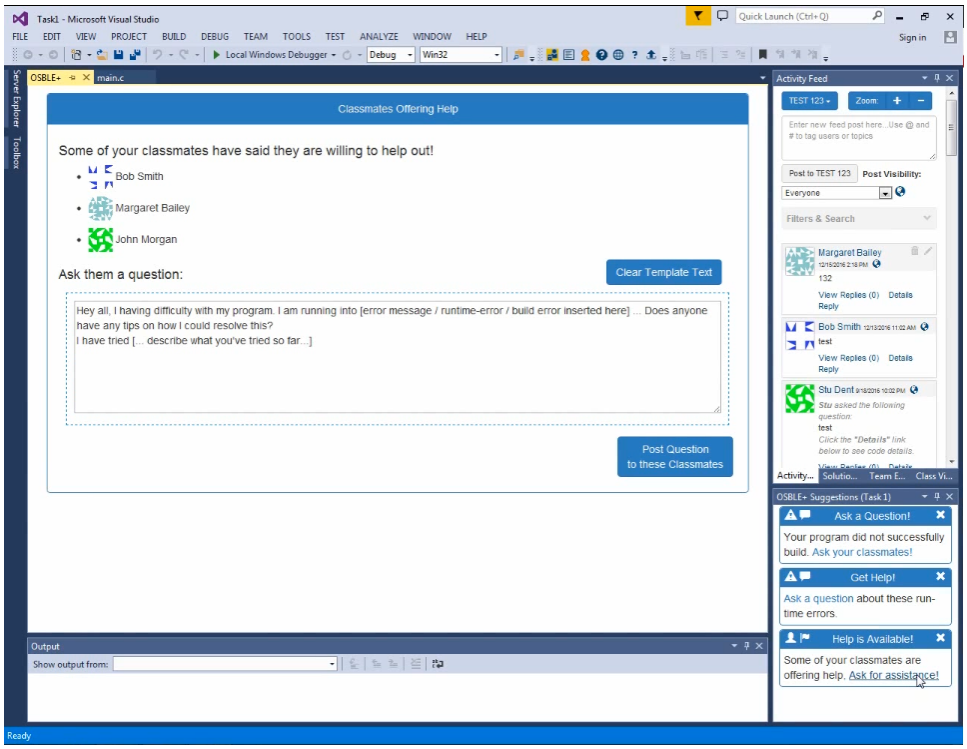

Figure 11.14. OSBLE+ Suggestions final prototype task 1c

Task 2

Task 2 walked the participant through a ***help-giving*** intervention scenario. As before, participants were given a brief scenario to situate them around the intervention actions. Task 2 provides a scenario that they just submitted their assignment early and elicits the user to describe their interactions with a "*Help Your Classmates!*" suggestions and the resulting intervention window (Figure 11.15).

www.manaraa.com

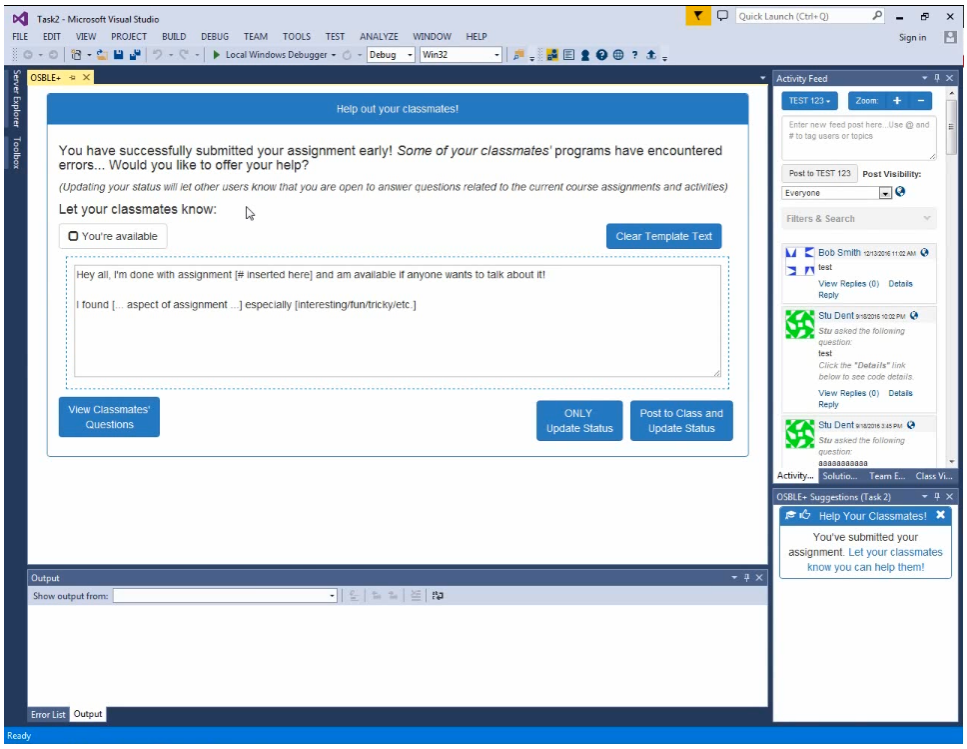

Figure 11.15. OSBLE+ Suggestions final prototype task 2

Task 3

Task 3 also focuses on the ***help-giving*** and ***social interaction*** intervention categories. For this task, participants are told that they have set their status to available and are directed to a "*You Are Available!*" prompt in the OSBLE+ Suggestions window. After clicking the prompt link, users are directed to their availability details page (Figure 11.16) and directed to describe their interactions with the intervention.

Task 4

In task 4, participants are walked through two ***help-giving*** event scenarios. The first prompt of "*Help Another Student!*" and the second of "*Help Other Students!*" both of which lead to the same unanswered questions prompt and its resultant intervention window seen in Figure 11.17.

www.manaraa.com

Task 5

Task 5 is focused on ***social interaction*** interventions. In this scenario event, participants are told that they have completed their work and have just submitted the assignment. They are then directed to two OSBLE+ Suggestions prompts; the "*Make a post!*" social and assignment submission (not considered early). The first prompt directs users to the topical social interaction prompt as seen in (Figure 11.18) and the second to the assignment submission prompt (Figure 11.19).

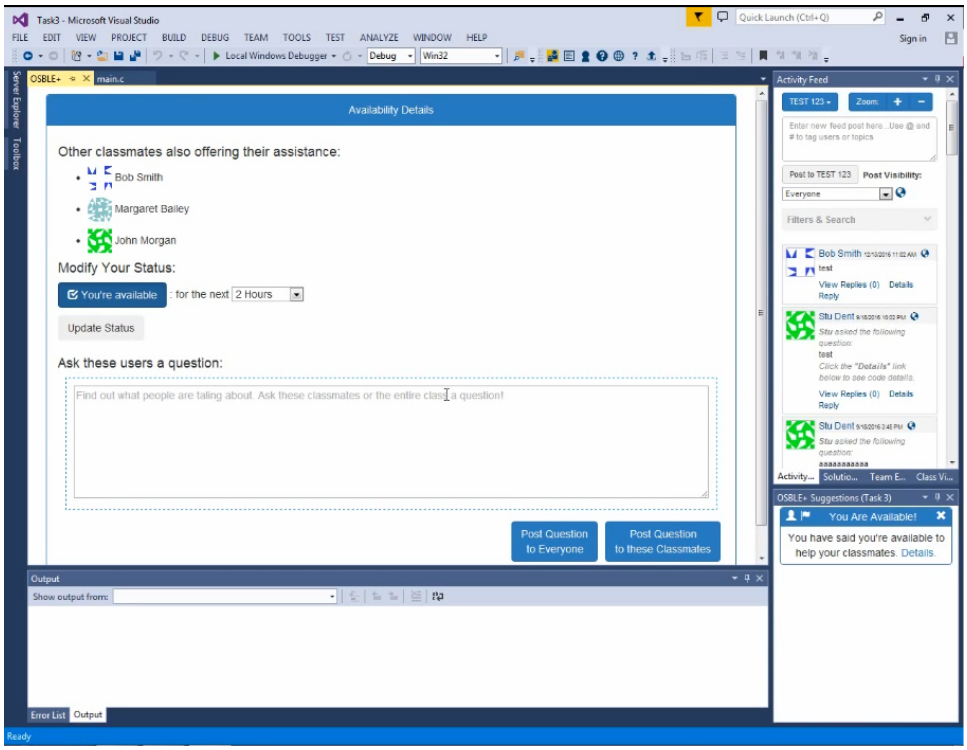

Figure 11.16. OSBLE+ Suggestions final prototype task 3

www.manaraa.com

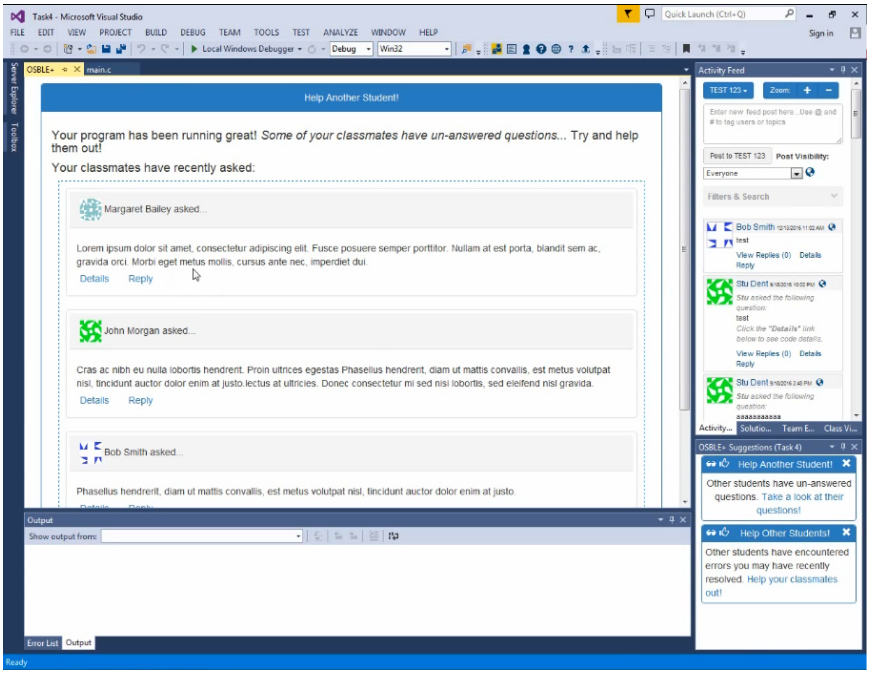

Figure 11.17. OSBLE+ Suggestions final prototypes task 4

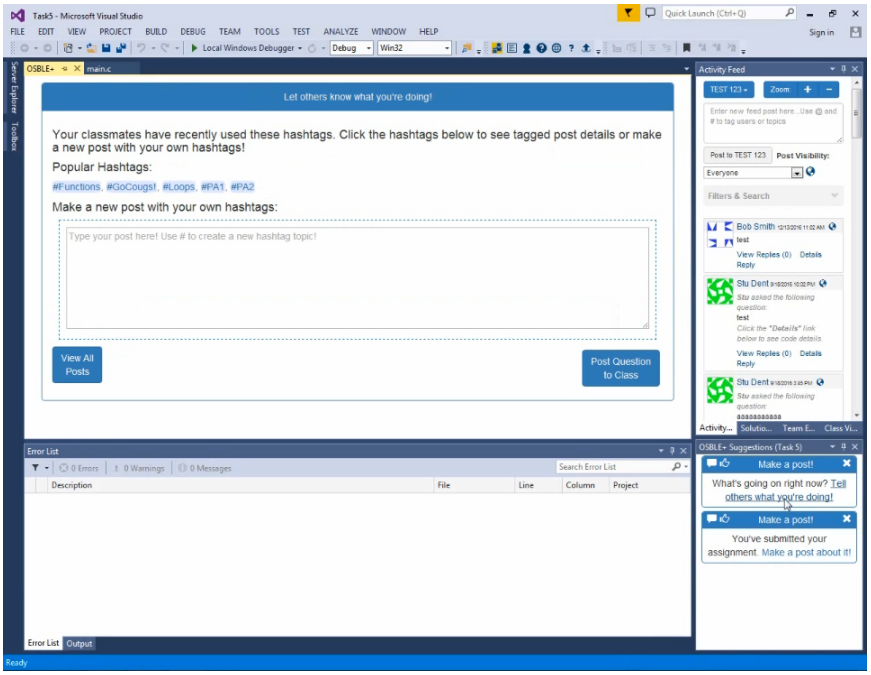

Figure 11.18. OSBLE+ Suggestions final prototype task 5b

www.manaraa.com

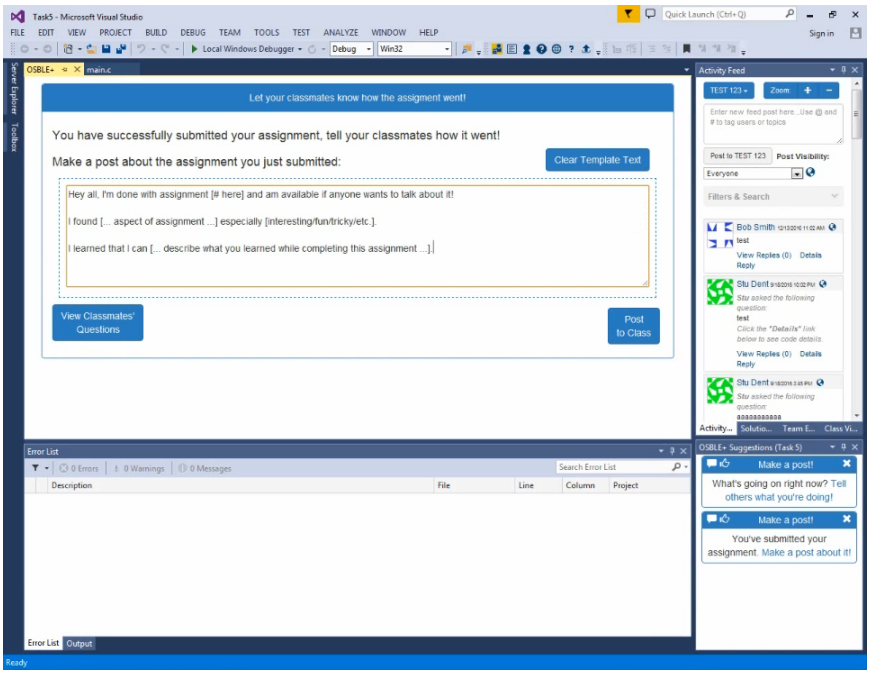

Figure 11.19. OSBLE+ Suggestions final prototype task 5c

Survey Responses

When asked how often they should be presented with the intervention prompts, students felt they would like to be presented with the 'ask for help' intervention with 10 minutes or less being the most frequent response (75%). When asked how many build or runtime errors they felt they should have before being presented with the 'ask for help' intervention all participants indicated 10 or less errors would be their preference with 75% of responses indicating that they felt the intervention should be presented when 5 or fewer errors have occurred.

Questions regarding the social interaction aspects of the interventions indicated that students felt they would offer assistance on assignments between 30-70% of the time when they had the opportunity with half of the participants indicating they would offer assistance 50-70% of the time. Additionally, when asked how often participants would help their peers after submitting an assignment, we see responses increase to a range of 30-100% with greater than 80% of participants indicating a willingness to offer help more than half of the time. All 6 participants stated that they

www.manaraa.com

would be interested in some social topics in the activity feed with the most responses indicating that they would like to see computer science and course related topics of some manner, e.g., study groups, lab/exam discussion, CS clubs, other CS topics though some participants did express concern that too off-topic posts could be distracting.

"Think Aloud" Feedback

The think aloud sections of the prototype study were designed to elicit participant interpretation of the intervention interfaces and potential usage. Analysis of the verbal feedback during participant interaction with the prototypes indicated a positive response to many of the intervention features and provided feedback on areas of the design that did not align with user interpretation. The feedback was used to adjust the both intervention delivery and design before deploying during the summative study. Below is a summary of key comments which affected the final prototype designs.

During task 1 participant feedback indicated that some users might be confused about or misinterpret what the "*Ask a Question!*" intervention (Figure 11.12 and Figure 11.13) was showing them, e.g. in task 1a and 1b one participant stated "I can look at their code and compare it to mine" when looking back over the build and runtime intervention suggestion indicating that they thought the code snippet was of someone else's code and not from their own code as intended. Additionally, during task 1c when looking at the "*Help is Available!*" intervention (Figure 11.14) one participant asked "Are they always willing to help out? Or is this just a time thing? … next few hours? Semester long?" indicating that just showing who is currently available might not be enough information. An additional comment of interest which indicated a possible change in the intervention interface was also during task 1c where a participant stated "I'd be more willing to

www.manaraa.com

ask for help from someone making themselves available for a time they decided" which further supported a need to include more information on the list of users available to help.

Task 2 feedback indicated interest in helping others where one participant stated "I'd do that … yeah, if I'd finished early and I had nothing to do I'd probably click that and see if I could help" showing an interest in helping others through the intervention but also said "this has to make noise" indicating a need for some sort of cue tied to the intervention prompts. There was some confusion between the difference between the update status and post to class and update status buttons indicating a need to better differentiate the two.

Further, two points of confusion expressed by users during Task 3 indicated a need for more information displayed on the availability details page (Figure 11.16) with participants making statements like "I'm just not sure who I'm detailing my availability for," "Maybe you could … show their activity?" and expressions of uncertainty as to what they should be inferring from the list of users. Parallel to the availability details intervention shown in task 3, one participant stated "It's nice to have that kind of information" in reference to the embedded activity feed in the IDE and wanting to know what was going on in the class when people were setting their status to available.

Task 4 showed a problem with the similarity between the two intervention prompts and their resulting intervention window (Figure 11.17) showing the same content (unanswered questions) with statements such as "The first [intervention] is plenty, they say they are unanswered questions, if you can answer them you might" and "oh, these are the same thing?" when looking at the two similar intervention prompts. Task 4 also produced additional positive feedback, e.g. "This is a good way to help … this is pretty good" and "I like the help other students window because I think

www.manaraa.com

it's pretty cool the computer can figure out that" when looking at the listing of unanswered questions.

In task 5, the topical intervention ("*Make a Post*": *What's going on right now?*) produced the feedback "It would be cool because then people that might not know about the events would actually hear about them.," "It would be kind of cool if things like [study groups] would be talked about in the activity feed," and "It brings more of a social aspect, it's not over the computer, kind of face-to-face" indicating additional positive reception of the intervention. Further, this intervention seemed to pull on familiarity with users of other social networking systems, e.g. one participant said "Oh! Hashtags … This is like Facebook!."

For the second assignment submission intervention ("*Make a post!*") in task 5, feedback provided supported the intent behind the motivating learning theories (Experiential Learning Theory) when a participant said "this is actually a good way to think about what you just programmed and then maybe reflect on it and also that would be a good way to help others in the community" which was the exact intent behind the scaffolded reflection prompted by the intervention. Finally, the positive responses were also seen in the last task with one user saying that "It would be pretty neat if OSBLE updated [the OSBLE+ Suggestions window] in real-time" instead of the fixed prompts experienced during the prototype study.

Summary

Some changes to the final versions of the interventions were based on good design principles overlooked during the prototype design but noticed prior to the deployment during the summative study but other primary changes implemented in the final version from the feedback provided during these sessions.

www.manaraa.com

For task 1a-1b, there was some confusion about the difference between the build and runtime error intervention prompts. The suggestion headers were modified in the final version to include the text build and runtime error to help differentiate the two as separate prompts. Additionally, there were indicators that there might be some confusion for the code excerpt from their own code. This was made more explicit in the final design by changing the code box from "Here's part of my code:" to "Here's part of my code: (edit)" to indicate that they can modify the code snippet (prompted by users attempting to modify the code while interacting) as well as adding opening and closing comment blocks to indicate it is a code excerpt.

Task 1c feedback also prompted a re-wording of "who" was willing to help out as, with a large class including multiple teaching assistants, it would be nice to know more than just the name of people in the list. The scaffolding of question asking was further supported by participant feedback, e.g., during task 1 a participant stated "When I ask someone for help, I don't know how to describe to them the issue that I'm having" indicating that they liked having the template text as an option to start from when asking questions.

The verbal feedback provided for task 2 (as mentioned above) prompted additional changes to the availability in the prototype to include additional information beyond the toggle as well as adding more detail to user status and additional descriptive text to help remove uncertainty between different window features. Task 3 brought up the issue that users wanted to be able to see who else was asking questions or in need of help and what their status was because just listing people alone was not enough detail. The availability details page was updated to include the additional details changed in the task 3 prompt, specifically allowing for a more refined adjustment of availability.

www.manaraa.com

The task 4 intervention prompts were found to be too similar and were merged into a single intervention as indicated by multiple instances of feedback and some comments mentioned uncertainty of whether or not there were any current replies to questions. These were addressed in the final design by merging the too similar interventions and by adding additional explanatory text to the intervention header as well as making sure there was parity in appearance between the posts listed in the unanswered questions list and the activity feed. It was also noted on task 5 that there was no way to navigate back from clicking one of the hashtag topics. In response to this, all intervention pages were changed to include a back button in the upper left corner of the window to return the user to the referring page.

Finally, all intervention prompts were also modified to include a post anonymously option after this prototype session.

www.manaraa.com

H Prototyping Survey (Summer 2016)

Summer 2016 Design Feedback

Scenario: You are a student in Computer Science 121 at WSU. Your professor has instructed you to create a program that will determine the fairest way to slice a pizza. The program will ask the user for the number of pizzas, the amount of people, and the number of slices each person gets. The program will then calculate how many slices each pizza needs to be cut into.

Part 1

Event 1: You have been working on coding this program for a while and are having trouble getting your program to build successfully. Assuming you don't immediately understand the cause of the error:

1. How often would you ask for assistance when having a programming problem like this?
    - Never (1)
    - Rarely, in less than 10% of the chances when I could have (2)
    - Occasionally, in about 30% of the chances when I could have (3)
    - Sometimes, in about 50% of the chances when I could have (4)
    - Frequently, in about 70% of the chances when I could have (5)
    - Usually, in about 90% of the chances I could have (6)
    - Every time (7)
2. What steps would you take to resolve the error? <OPEN ENDED>
3. How likely are you to ask your INSTRUCTOR for help?
    - (rated 1-7 Not Likely at All to Very Likely)
4. How likely are you to ask your TA (Teaching Assistant) for help?

www.manaraa.com

- (rated 1-7 Not Likely at All to Very Likely)

5. How likely are you to ask your CLASSMATES for help?
    - (rated 1-7 Not Likely at All to Very Likely)

Event 2: You have managed to fix your code and it now builds successfully. Unfortunately, your program now crashes after running. Assuming you don't immediately understand the cause of the error:

1. How often would you ask for assistance when having a programming problem like this?
    - Never (1)
    - Rarely, in less than 10% of the chances when I could have (2)
    - Occasionally, in about 30% of the chances when I could have (3)
    - Sometimes, in about 50% of the chances when I could have (4)
    - Frequently, in about 70% of the chances when I could have (5)
    - Usually, in about 90% of the chances I could have (6)
    - Every time (7)
2. What steps would you take to resolve the error? <OPEN ENDED>
3. How likely are you to ask your INSTRUCTOR for help?
    - (rated 1-7 Not Likely at All to Very Likely)
4. How likely are you to ask your TA (Teaching Assistant) for help?
    - (rated 1-7 Not Likely at All to Very Likely)
5. How likely are you to ask your CLASSMATES for help?
    - (rated 1-7 Not Likely at All to Very Likely)

www.manaraa.com

Event 3: You have managed to fix the issue causing your code to crash. Unfortunately, your program now has additional errors. For example, you keep running into the same error, you keep getting multiple errors, or you are unable to successfully build, etc. Assuming you don't immediately understand the cause of the error:

1. How often would you ask for assistance when having a programming problem like this?
    - Never (1)
    - Rarely, in less than 10% of the chances when I could have (2)
    - Occasionally, in about 30% of the chances when I could have (3)
    - Sometimes, in about 50% of the chances when I could have (4)
    - Frequently, in about 70% of the chances when I could have (5)
    - Usually, in about 90% of the chances I could have (6)
    - Every time (7)
2. What steps would you take to resolve the error? <OPEN ENDED>
3. How likely are you to ask your INSTRUCTOR for help?
    - (rated 1-7 Not Likely at All to Very Likely)
4. How likely are you to ask your TA (Teaching Assistant) for help?
    - (rated 1-7 Not Likely at All to Very Likely)
5. How likely are you to ask your CLASSMATES for help?
    - (rated 1-7 Not Likely at All to Very Likely)

Event 4: You have managed to fix your previous errors and now your code builds and runs as expected.

1. How likely are you to share your coding success with someone else?

www.manaraa.com

- (rated 1-7 Not Likely at All to Very Likely)

2. How likely are you to help someone else with a coding issue?
   - (rated 1-7 Not Likely at All to Very Likely)
3. Have you ever helped anyone else with a coding problem after you overcame a similar problem?
   - Never (1)
   - Rarely, in less than 10% of the chances when I could have (2)
   - Occasionally, in about 30% of the chances when I could have (3)
   - Sometimes, in about 50% of the chances when I could have (4)
   - Frequently, in about 70% of the chances when I could have (5)
   - Usually, in about 90% of the chances I could have (6)
   - Every time (7)
4. What were your reasons for choosing to help (or not help) someone else? (e.g. you did not feel comfortable helping or were told not to share your code.) <OPEN ENDED>

Part 2

We'll now revisit the previous four events. Imagine you see the following windows displayed next to your debug/error output window in the Visual Studio IDE (pictured below).

Figure 11.7. Prototype intervention location

Event 1: You have been working on coding this program for a while and are having trouble getting your program to build successfully.

Figure 11.8. Build error intervention prototype (event 1)

1. What do you think this window does? <OPEN ENDED>

www.manaraa.com

2. How likely would you make use of the above window?
    - (rated 1-7 Not Likely at All to Very Likely)
3. How frequently would you use of the above window?
    - Never (1)
    - 1-2 times per semester (2)
    - 1-2 times per month (3)
    - 1-2 times per week (4)
    - 3+ times per week (5)
    - 1-2 times per day (6)
    - 3+ times per day (7)
4. What would you change about or add to the above window? <OPEN ENDED>
5. How likely do you feel that this window might help resolve your issues?
    - (rated 1-7 Not Likely at All to Very Likely)

Event 2: You have managed to fix your code and it now builds successfully. Unfortunately, your program now crashes after running. Imagine you see the following window displayed next to your debug/error output window.

Figure 11.9. Runtime intervention prototype (event 2)

1. What do you think this window does? <OPEN ENDED>
2. How likely would you make use of the above window?
    - (rated 1-7 Not Likely at All to Very Likely)
3. How frequently would you use of the above window?
    - Never (1)

www.manaraa.com

- 1-2 times per semester (2)
- 1-2 times per month (3)
- 1-2 times per week (4)
- 3+ times per week (5)
- 1-2 times per day (6)
- 3+ times per day (7)

4. What would you change about or add to the above window? <OPEN ENDED>
5. How likely do you feel that this window might help resolve your issues?
    - (rated 1-7 Not Likely at All to Very Likely)

Event 3: You have managed to fix the issue causing your code to crash. Unfortunately, your program now has additional errors. For example, you keep running into the same error, you keep getting multiple errors, or you are unable to successfully build, etc. Imagine you see the following window displayed next to your debug/error output window.

Figure 11.10. Help-seeking intervention prototype (event 3)

1. What do you think this window does? <OPEN ENDED>
2. How likely would you make use of the above window?
    - (rated 1-7 Not Likely at All to Very Likely)
3. How frequently would you use of the above window?
    - Never (1)
    - 1-2 times per semester (2)
    - 1-2 times per month (3)
    - 1-2 times per week (4)

www.manaraa.com

- 3+ times per week (5)
- 1-2 times per day (6)
- 3+ times per day (7)

4. What would you change about or add to the above component? <OPEN ENDED>
5. How likely do you feel that this window might help resolve your issues?
    - (rated 1-7 Not Likely at All to Very Likely)

Event 4: You have managed to fix your previous errors and now your code builds and runs as expected. Imagine you see the following window displayed next to your debug/error output window.

Figure 11.11. Social interaction intervention prototype (event 4)

1. What do you think this window does? <OPEN ENDED>
2. How likely would you make use of the above window?
    - (rated 1-7 Not Likely at All to Very Likely)
3. How frequently would you use of the above window?
    - Never (1)
    - 1-2 times per semester (2)
    - 1-2 times per month (3)
    - 1-2 times per week (4)
    - 3+ times per week (5)
    - 1-2 times per day (6)
    - 3+ times per day (7)
4. What would you change about or add to the above window? <OPEN ENDED>

www.manaraa.com

5. How likely do you feel that this window might help resolve your issues?

    - (rated 1-7 Not Likely at All to Very Likely)

Survey Completed!

Thank you for your participation.

www.manaraa.com

I Final Intervention Prototype Pilot Study Survey (Fall 2016)

The following sections will include 5 tasks. Please read the scenario aloud and follow the prompt given for each question.

At the start of each task, you will open the corresponding Task solution file. Please use the 'Task Solutions' folder on the desktop to open each solution file.

Once you have read the scenario and task, feel free to interact with the IDE as you're describing your intended interactions.

As you participate in today's study, please keep the following points in mind:

- The software tool, and not you, is the entity under test! Your interaction with our software will help us to better understand our software's strengths and weaknesses, so that we can ultimately improve the software's design.
- You are free to take a break at any time. Before beginning each exercise, please read all written instructions aloud.
- Please read the problem aloud before you begin. As you work through each exercise, please "think aloud." Let me know what you are up to by verbalizing your thoughts and actions. In addition, please share any opinions, questions, or concerns that come to mind. If, at any point, you become silent, I will remind you to continue thinking aloud.
- You may skip reading aloud any 'Reminder:' text on this survey.
- Have fun!

Scenario: You are a student in Computer Science 121 at WSU. Your professor has instructed you to create a program that will calculate the Fibonacci series for the first N terms (decided by user input) and store those N values in an array.

www.manaraa.com

Task 1

1. Open Task1.sln

2. Briefly look over the code in main.c.

Event 1a: You have been working on coding this program for a while and are having trouble getting your program to build successfully. Assuming you don't immediately understand the cause of the error... You notice the first notification, "Ask a Question!" in the OSBLE+ Suggestions window.

Reminder: Please "think aloud." Let me know what you are up to by verbalizing your thoughts and actions. In addition, please share any opinions, questions, or concerns that come to mind. Describe how you would interact with the "Ask a Question!" suggestion.

If you have not already done so, click the "Ask your classmates!" link located in the "Ask a Question!" suggestion.

Reminder: Please "think aloud." Let me know what you are up to by verbalizing your thoughts and actions. In addition, please share any opinions, questions, or concerns that come to mind

Describe how you would interact with this new "Ask a Question!" window.

Event 1b: You have managed to fix your code! You discovered that you had a missing comma after initializing your variable 'first' ('first=0' => 'first=0,') and it now builds successfully. Unfortunately, your program now crashes after running for certain user input values (you entered

www.manaraa.com

'15'). Assuming you don't immediately understand the cause of the error: You notice the second notification, "Get Help!" in the OSBLE+ Suggestions window.

Reminder: Please "think aloud." Let me know what you are up to by verbalizing your thoughts and actions. In addition, please share any opinions, questions, or concerns that come to mind.

Describe how you would interact with the "Get Help!" suggestion.

If you have not already done so, click the "Ask a question" link located in the "Get Help!" suggestion.

Reminder: Please "think aloud." Let me know what you are up to by verbalizing your thoughts and actions. In addition, please share any opinions, questions, or concerns that come to mind

Describe how you would interact with this new "Ask a Question!" window. Are there any changes to these pages you feel would make them more useful and/or more likely to be used? Please elaborate.

Event 1c: During either of the previous problems, you noticed the last notification, "Help is Available!" in the OSBLE+ Suggestions window. Assuming you don't immediately understand the cause of the error:

Reminder: Please "think aloud." Let me know what you are up to by verbalizing your thoughts and actions. In addition, please share any opinions, questions, or concerns that come to mind

Describe how you would interact with the "Help is Available!" suggestion:

www.manaraa.com

If you have not already done so, click the "Ask for assistance!" link located in the "Help is Available!" suggestion.

Reminder: Please "think aloud." Let me know what you are up to by verbalizing your thoughts and actions. In addition, please share any opinions, questions, or concerns that come to mind

Describe how you would interact with the "Classmates Offering Help!" suggestion: Refer to the OSBLE+ Suggestions (Task 1) window for the following questions: Feel free to "think aloud" and elaborate on any of your responses.

How much time being 'stuck' (i.e. you have unresolved errors) do you feel should pass before seeing an 'ask for help' notification?

1. None: show immediately (1)
2. 5 minutes (2)
3. 10 minutes (3)
4. 30 minutes (4)
5. Other (specify): (5) _____

How many build and/or runtime failures do you feel should occur before seeing an 'ask for help' notification?

1. None: show immediately (1)
2. 3 (2)
3. 5 (3)
4. 10 (4)
5. Other (specify): (5) _____

www.manaraa.com

Assuming you were not immediately able to solve your problem, how often would you ask for assistance using these suggestions when having a programming problem?

1. Never (1)
2. Rarely, in less than 10% of the chances when I could have (2)
3. Occasionally, in about 30% of the chances when I could have (3)
4. Sometimes, in about 50% of the chances when I could have (4)
5. Frequently, in about 70% of the chances when I could have (5)
6. Usually, in about 90% of the chances I could have (6)
7. Every time (7)

Reminder: Please "think aloud." Let me know what you are up to by verbalizing your thoughts and actions. In addition, please share any opinions, questions, or concerns that come to mind.

Are there any changes to the "OSBLE+ Suggestions (Task 1)" window that you feel would make it more useful and/or more likely to be used? Please elaborate.

Task 2

1. Open Task2.sln

Event 2: You figured out that you were entering a number out of bounds of your array, luckily your assignment prompt specifies that you do not have to validate user input, so no further changes are needed! You go ahead and submit your assignment 3 days early! You notice a new notification, "Help Your Classmates!" in the OSBLE+ Suggestions window.

www.manaraa.com

Reminder: Please "think aloud." Let me know what you are up to by verbalizing your thoughts and actions. In addition, please share any opinions, questions, or concerns that come to mind

Describe how you would interact with the "Help Your Classmates!" suggestion:

If you have not already done so, click the "Let your classmates know you can help them!" link located in the "Help Your Classmates!" suggestion.

Reminder: Please "think aloud." Let me know what you are up to by verbalizing your thoughts and actions. In addition, please share any opinions, questions, or concerns that come to mind

Describe how you would interact with the "Help out your classmates!" suggestion: Are there any changes to this page that you feel would make it more useful and/or more likely to be used? Please explain:

Referring to the OSBLE+ Suggestions (Task 2) Window and the "Help Your Classmates!" suggestion: Feel free to "think aloud" and elaborate on any of your responses.

How often would you offer your assistance after submitting an assignment?

1. Never (1)
2. Rarely, in less than 10% of the chances when I could have (2)
3. Occasionally, in about 30% of the chances when I could have (3)
4. Sometimes, in about 50% of the chances when I could have (4)
5. Frequently, in about 70% of the chances when I could have (5)

www.manaraa.com

6. Usually, in about 90% of the chances I could have (6)
7. Every time (7)

How often would you offer your assistance when you have not yet submitted your assignment, but your program is currently error free?

1. Never (1)
2. Rarely, in less than 10% of the chances when I could have (2)
3. Occasionally, in about 30% of the chances when I could have (3)
4. Sometimes, in about 50% of the chances when I could have (4)
5. Frequently, in about 70% of the chances when I could have (5)
6. Usually, in about 90% of the chances I could have (6)
7. Every time (7)

Task 3

1. Open Task3.sln

Event 3: A few minutes ago, you submitted your assignment and decided to mark yourself as 'available' for the next 30 minutes via the 'OSBLE+ Suggestions' window. You notice a 'You are Available!' notification in the window suggestions window.

Reminder: Please "think aloud." Let me know what you are up to by verbalizing your thoughts and actions. In addition, please share any opinions, questions, or concerns that come to mind

Describe how you would interact with the "You Are Available!" suggestion

If you have not already done so, click the "Details." link located in the "You Are Available" suggestion.

www.manaraa.com

Reminder: Please "think aloud." Let me know what you are up to by verbalizing your thoughts and actions. In addition, please share any opinions, questions, or concerns that come to mind

Describe how you would interact with the "Availability Details" suggestion: Are there any changes to this page that you feel would make it more useful and/or more likely to be used? Please explain:

Task 4

1. Open Task4.sln

Event 4: You have just started a new programming session. You notice there are new suggestions in the 'OSBLE+ Suggestions' window.

Reminder: Please "think aloud." Let me know what you are up to by verbalizing your thoughts and actions. In addition, please share any opinions, questions, or concerns that come to mind

Describe how you would interact with the "Help Another Student!" and "Help Other Students!" suggestions

If you have not already done so, click the "Take a look at their questions!" or "Help your classmates out!" links located in the suggestions window.

Reminder: Please "think aloud." Let me know what you are up to by verbalizing your thoughts and actions. In addition, please share any opinions, questions, or concerns that come to mind.

Describe how you would interact with the "Help Another Student!" suggestion: Are there any changes to this page that you feel would make it more useful and/or more likely to be used? Please explain:

www.manaraa.com

Task 5

Event 5: You have managed to fix your previous errors and now your code builds and runs as expected. You have also submitted your assignment and are done for the week! You notice there are new suggestions in the 'OSBLE+ Suggestions' window.

Reminder: Please "think aloud." Let me know what you are up to by verbalizing your thoughts and actions. In addition, please share any opinions, questions, or concerns that come to mind.

Describe how you would interact with the "Make a Post" suggestions.

If you have not already done so, click the "Tell others what you're doing!" link located in the suggestions window.

Reminder: Please "think aloud." Let me know what you are up to by verbalizing your thoughts and actions. In addition, please share any opinions, questions, or concerns that come to mind

Describe how you would interact with the "Let others know what you're doing!" suggestion: Are there any changes to this page that you feel would make it more useful and/or more likely to be used? Please explain:

If you have not already done so, click the "Make a post about it!" link located in the suggestions window.

www.manaraa.com

Reminder: Please "think aloud." Let me know what you are up to by verbalizing your thoughts and actions. In addition, please share any opinions, questions, or concerns that come to mind

Describe how you would interact with the "Let your classmates know how the assignment went!" suggestion: Are there any changes to this page that you feel would make it more useful and/or more likely to be used? Please explain:

Are there any general topics (formal or informal, e.g. coding topics, campus activities, entertainment, etc.) you feel would have sparked interesting conversation on the activity feed during the CptS 121 semester?

Which topics would you have found interesting to see posted on the activity feed?

Which topics do you think your classmates might have found interesting to see posted on the activity feed?

Feel free to also "think aloud" as you consider possible topics. Please list below: **<OPEN ENDED>**

www.manaraa.com

J Study I-III Online Surveys

General Demographic Data

1. Name
2. WSU ID Number
3. Age
4. Gender
5. Ethnicity
6. Class Standing
7. Are you presently a computer science major?
    a. (If no) Do you plan to become a computer science major?
        i.(If no) What is your intended major? How likely are you to switch your major to computer science?
    b. (If yes) How likely are you to switch to a major that is not computer science?
8. What is your reason for taking CptS 121?
9. How likely are you to enroll in the next computer science in this sequence (CptS 122)? (CptS 122 Data Structures C/C++: Advanced programming techniques: data structures, recursion, sorting and searching, and basics of algorithm analysis)

www.manaraa.com

Adapted C++ Self-Efficacy Survey

Rate your confidence in doing the following C programming related tasks using a scale of 1 (not at all confident) to 7 (absolutely confident). If a specific term or task is totally unfamiliar to you, please mark 1.

1. Write syntactically correct C statements.
2. Understand the language structure of C and the usage of the reserved words.
3. Write logically correct blocks of code using C.
4. Write a C program that displays a greetings message.
5. Write a C program that computes the average of three numbers.
6. Write a C program that computes the average of any given number of numbers.
7. Use built-in functions that are available in the various C libraries.
8. Build my own C libraries.
9. Write a small C program given a small problem that is familiar to me.
10. Write a reasonably sized C program that can solve a problem that is only vaguely familiar to me.
11. Write a long and complex C program to solve any given problem as long as the specifications are clearly defined.
12. Organize and design my program in a modular manner.
13. Understand the procedural programming paradigm.
14. Identify the data types in the problem domain and declare, define, and use them.

www.manaraa.com

15. Make use of a pre-written function, given a clearly labeled declaration of the function.

16. Make use of a data structure that is already defined, given a clearly labeled declaration of the data structure.

17. Debug (correct all the errors) a long and complex program that I had written, and make it work.

18. Comprehend a long, complex multi-file program.

19. Complete a programming project if someone showed me how to solve the problem first.

20. Complete a programming project if I had only the language reference manual for help.

21. Complete a programming project if I could call someone for help if I got stuck.

22. Complete a programming project once someone else helped me get started.

23. Complete a programming project if I had a lot of time to complete the program.

24. Complete a programming project if I had just the built-in help facility for assistance.

25. Find ways of overcoming the problem if I got stuck at a point while working on a programming project.

26. Come up with a suitable strategy for a given programming project in a short time.

27. Manage my time efficiently if I had a pressing deadline on a programming project.

www.manaraa.com

28. Mentally trace through the execution of a long, complex, multi-file program given to me.

29. Rewrite lengthy confusing portions of code to be more readable and clear.

30. Find a way to concentrate on my program, even when there were many distractions around me.

31. Find ways of motivating myself to program, even if the problem area was of no interest to me.

32. Write a program that someone else could comprehend and add features to at a later date.

Classroom Community Scale

Below, you will see a series of statements concerning a specific course or program you are presently taking or have recently completed. Read each statement carefully and select the statement that comes closest to indicate how you feel about the course or program. There are no correct or incorrect responses. If you neither agree nor disagree with a statement or are uncertain, select the neutral area. Do not spend too much time on any one statement, but give the response that seems to describe how you feel. Please respond to all items.

1. I feel that students in this course care about each other

2. I feel that I am encouraged to ask questions

3. I feel connected to others in this course

4. I feel that it is hard to get help when I have a question

5. I do not feel a spirit of community

6. I feel that I receive timely feedback

www.manaraa.com

7. I feel that this course is like a family

8. I feel uneasy exposing gaps in my understanding

9. I feel isolated in this course

10. I feel reluctant to speak openly

11. I trust others in this course

12. I feel that this course results in only modest learning

13. I feel that I can rely on others in this course

14. I feel that other students do not help me learn

15. I feel that members of this course depend on me

16. I feel that I am given ample opportunities to learn

17. I feel uncertain about others in this course

18. I feel that my educational needs are not being met

19. I feel confident that others will support me

20. I feel that this course does not promote a desire to learn

Motivated Strategies for Learning Questionnaire

The following questions ask about your motivation for and attitudes about this class. Remember there are no right or wrong answers, just answer as accurately as possible. Use the scale below to answer the questions. If you think the statement is very true of you, circle 7; if a statement is not at all true of you, circle 1. If the statement is more or less true of you, find the number between 1 and 7 that best describes you.

1. I believe I will receive an excellent grade in this class.

www.manaraa.com

2. I'm certain I can understand the most difficult material presented in the readings for this course.

3. I'm confident I can learn the basic concepts taught in this course.

4. I'm confident I can understand the most complex material presented by the instructor in this course.

5. I'm confident I can do an excellent job on the assignments and tests in this course.

6. I expect to do well in this class.

7. I'm certain I can master the skills being taught in this class.

8. Considering the difficulty of this course, the teacher, and my skills, I think I will do well in this class.

The following questions ask about your learning strategies and study skills for this class. Again, there are no right or wrong answers. Answer the questions about how you study in this class as accurately as possible. Use the same scale to answer the remaining questions. If you think the statement is very true of you, circle 7; if a statement is not at all true of you, circle 1. If the statement is more or less true of you, find the number between 1 and 7 that best describes you.

1. When studying for this course, I often try to explain the material to a classmate or friend.

2. I try to work with other students from this class to complete the course assignments.

3. When studying for this course, I often set aside time to discuss course material with a group of students from the class.

www.manaraa.com

Sociability Scale

Below, you will see a series of statements concerning the OSBLE+ Dashboard. Read each statement carefully and select the statement that comes closest to indicate how you feel about OSBLE+. There are no correct or incorrect responses. Do not spend too much time on any one statement, but give the response that seems to describe how you feel. Please respond to all items.

Ratings: Not applicable at all (1), Rarely applicable (2), Moderately applicable (3), Largely applicable (4), Totally applicable (5)

1. The OSBLE+ environment enables me to easily contact my classmates (1)
2. I do not feel lonely in the OSBLE+ environment (2)
3. The OSBLE+ environment enables me to get a good impression of my classmates (3)
4. The OSBLE+ environment allows spontaneous informal conversations (4)
5. The OSBLE+ environment enables us to develop into a well performing class (5)
6. The OSBLE+ environment enables me to develop good work relationships with my classmates (6)
7. The OSBLE+ environment enables me to identify myself with the class (7)
8. I feel comfortable with the OSBLE+ environment (8)
9. The OSBLE+ environment allows for non-programming-related conversations (9)
10. The OSBLE+ environment enables me to make close friendships with my classmates (10)

www.manaraa.com

Self-Sociability

Describe yourself as you generally are now, not as you wish to be in the future. Describe yourself as you honestly see yourself. Indicate for each statement whether it is 1. Very Inaccurate, 2. Moderately Inaccurate, 3. Neither Accurate nor Inaccurate, 4. Moderately Accurate, or 5. Very Accurate as a description of you. Please respond to all items.

1. Feel comfortable around people
2. Often feel uncomfortable around others.
3. Act comfortably with others
4. Have little to say.
5. Am skilled in handling social situations.
6. Find it difficult to approach others.
7. Talk to a lot of different people at parties.
8. Have difficulty expressing my feelings.
9. Start conversations.
10. Only feel comfortable with friends.

System Usability Scale

For each usability session the System Usability Scale will be used as a quick judge of usability. Please rate the following questions on a scale of 1 to 5 where 1 means you "Strongly Disagree" and 5 means you "Strongly Agree" with the statement.

1. I think that I would like to use this system frequently.

www.manaraa.com

2. I found the system unnecessarily complex.
3. I thought the system was easy to use.
4. I think that I would need the support of a technical person to be able to use this system.
5. I found the various functions in this system were well integrated.
6. I thought there was too much inconsistency in this system.
7. I would imagine that most people would learn to use this system very quickly.
8. I found the system very cumbersome to use.
9. I felt very confident using the system.
10. I needed to learn a lot of things before I could get going with this system.

www.manaraa.com

K Study I-III Empirical Study Intervention Survey

OSBLE+ Experience

Activity Feed Use

Select the statement which most accurately describes your use of the OSBLE+ activity feed:

1. I used the activity feed from within Visual Studio most frequently
2. I used the activity feed directly on plus.osble.org via a web browser most frequently
3. I used the activity feed about the same frequency from within Visual Studio AND via a web browser.
4. Other (please elaborate):_____
5. I did not use the activity feed.

In the box below, please share any feedback you have regarding the OSBLE+ Dashboard? e.g. the sidebars: calendar or Files & Links, the activity feed, assignments tab, grades tab, etc... (not the Visual Studio IDE) **<OPEN ENDED>**

OSBLE+ Suggestions Design

OSBLE+ Suggestions (Help Seeking) 1

In this section, questions will be preceded by images of OSBLE+ suggestions you might have seen while using the OSBLE+ system.

See [Figure 4.11. Help-Seeking intervention prompts (middle)]

See [Figure 4.12. Runtime Errors: Get Help! help-seeking intervention]

Did you ever encounter/view a "Runtime Errors: Get Help!" suggestion? (First Image)

1. Yes

www.manaraa.com

2. No
3. I don't remember.

Did you interact with it in any manner?

1. Yes, I clicked the "Ask a Question" link.
2. Yes, I clicked the dismiss 'X'
3. Yes, I clicked both the "Ask a Question" link and the dismiss 'X'.
4. No
5. I don't remember.
6. Other (please specify)_____

Did you ever interact with the "Ask a Question!" window after following the "Ask a Question" link in the OSBLE+ suggestion? (Second Image)

1. Yes
2. No
3. I don't remember.

Please describe how you interacted with the "Ask a Question!" window. **<OPEN ENDED>**

On a scale from 0-10, based on your experience using this suggestion, how likely are you to use this suggestion in the future?

www.manaraa.com

OSBLE+ Suggestions (Help Seeking) 2

See [Figure 4.11. Help-Seeking intervention prompts (left)]

See [Figure 4.13. Build Errors: Get Help! help-seeking intervention]

Did you ever encounter/view a "Build Errors: Get Help!" suggestion? (First Image)

1. Yes
2. No
3. I don't remember.

Did you interact with it in any manner?

1. Yes, I clicked the "Ask a Question" link.
2. Yes, I clicked the dismiss 'X'
3. Yes, I clicked both the "Ask a Question" link and the dismiss 'X'.
4. No
5. I don't remember.
6. Other (please specify)_____

Did you ever interact with the "Ask a Question!" window after following the "Ask a Question" link in the OSBLE+ suggestion? (Second Image)

1. Yes
2. No
3. I don't remember.

Please describe how you interacted with the "Ask a Question!" window. **<OPEN ENDED>**

www.manaraa.com

On a scale from 0-10, based on your experience using this suggestion, how likely are you to use this suggestion in the future?

OSBLE+ Suggestions (Help Seeking) 3

See [Figure 4.11. Help-Seeking intervention prompts (right)]

See [Figure 4.14. Others Available to Help! help-seeking intervention]

Did you ever encounter/view a "Others Available to Help!" suggestion? (First Image)

1. Yes
2. No
3. I don't remember.

Did you interact with it in any manner?

1. Yes, I clicked the "Ask for Assistance" link.
2. Yes, I clicked the dismiss 'X'
3. Yes, I clicked both the "Ask for Assistance" link and the dismiss 'X'.
4. No
5. I don't remember.
6. Other (please specify)_____

Did you ever interact with the "Classmates Offering Help!" window after following the "Ask for Assistance" link in the OSBLE+ suggestion? (Second Image)

1. Yes
2. No

www.manaraa.com

3. I don't remember.

Please describe how you interacted with the "Classmates Offering Help!" window. **<OPEN ENDED>**

On a scale from 0-10, based on your experience using this suggestion, how likely are you to use this suggestion in the future?

OSBLE+ Suggestions (Help Giving) 1

See [Figure 4.15. Help-Giving intervention prompts (left)]

See [Figure 4.16. Help other Students! help-giving intervention]

Did you ever encounter/view a "Help Other Students!" suggestion? (First Image)

1. Yes
2. No
3. I don't remember.

Did you interact with it in any manner?

1. Yes, I clicked the "Help your classmates out" link.
2. Yes, I clicked the dismiss 'X'
3. Yes, I clicked both the "Help your classmates out" link and the dismiss 'X'.
4. No
5. I don't remember.
6. Other (please specify)_____

www.manaraa.com

Did you ever interact with the "Help Another Student!" window after following the "Help your classmates out!" link in the OSBLE+ suggestion? (Second Image)

1. Yes
2. No
3. I don't remember.

Please describe how you interacted with the "Help Another Student!" window. **<OPEN ENDED>**

On a scale from 0-10, based on your experience using this suggestion, how likely are you to use this suggestion in the future?

OSBLE+ Suggestions (Help Giving) 2

See [Figure 4.15. Help-Giving intervention prompts (right)]

See [Figure 4.17. Help Your Classmates! (early assignment submission) help-giving intervention]

Did you ever encounter/view a "Help Your Classmates!" suggestion? (First Image)

1. Yes
2. No
3. I don't remember.

Did you interact with it in any manner?

1. Yes, I clicked the "Let your classmates know you can help them!" link.
2. Yes, I clicked the dismiss 'X'

www.manaraa.com

3. Yes, I clicked both the "Let your classmates know you can help them!" link and the dismiss 'X'.
4. No
5. I don't remember.
6. Other (please specify)_____

Did you ever interact with the "Help out your classmates!" window after following the "Let your classmates know you can help them!" link in the OSBLE+ suggestion? (Second Image)

1. Yes
2. No
3. I don't remember.

Please describe how you interacted with the "Help out your classmates!" window. **<OPEN ENDED>**

On a scale from 0-10, based on your experience using this suggestion, how likely are you to use this suggestion in the future?

OSBLE+ Suggestions Social 1

See [Figure 4.18. Social Interaction intervention prompts (left)]

See [Figure 4.19. You Are Available! social interaction intervention]

Did you ever encounter/view a "You Are Available!" suggestion? (First Image)

1. Yes
2. No
3. I don't remember.

www.manaraa.com

Did you interact with it in any manner?

1. Yes, I clicked the "View/Change your status" link.
2. Yes, I clicked the dismiss 'X'
3. Yes, I clicked both the "View/Change your status" link and the dismiss 'X'.
4. No
5. I don't remember.
6. Other (please explain)_____

Did you ever interact with the "Availability Details" window after following the "View/Change your status" link in the OSBLE+ suggestion? (Second Image)

1. Yes
2. No
3. I don't remember.
4. I used the "Availability" link found at the top of the OSBLE+ activity feed
5. Other (please explain):_____

Please describe how you interacted with the "Availability Details" window. **<OPEN ENDED>**

On a scale from 0-10, based on your experience using this suggestion, how likely are you to use this suggestion in the future?

www.manaraa.com

See [Figure 4.18. Social Interaction intervention prompts (middle)]

See [Figure 4.20. Make a Post! (topical) social interaction intervention]

Did you ever encounter/view this specific "Make a Post!" suggestion? (First Image)

1. Yes
2. No
3. I don't remember.

Did you interact with it in any manner?

1. Yes, I clicked the "Tell others what you are doing!" link.
2. Yes, I clicked the dismiss 'X'
3. Yes, I clicked both the "Tell others what you are doing!" link and the dismiss 'X'.
4. No
5. I don't remember.
6. Other (please specify)_____

Did you ever interact with the "Let know what you are doing!" window after following the "Tell others what you are doing!" link in the OSBLE+ suggestion? (Second Image)

1. Yes
2. No
3. I don't remember.

www.manaraa.com

Please describe how you interacted with the "Let know what you are doing!" window. **<OPEN ENDED>**

On a scale from 0-10, based on your experience using this suggestion, how likely are you to use this suggestion in the future?

OSBLE+ Suggestions Social 3

See [Figure 4.18. Social Interaction intervention prompts (right)]

See [Figure 4.21. Make a Post! (assignment submission) social interaction intervention]

Did you ever encounter/view this specific "Make a Post!" suggestion? (First Image)

1. Yes
2. No
3. I don't remember.

Did you interact with it in any manner?

1. Yes, I clicked the "Make a post about it!" link.
2. Yes, I clicked the dismiss 'X'
3. Yes, I clicked both the "Make a post about it!" link and the dismiss 'X'.
4. No
5. I don't remember.
6. Other (please specify)_____

Did you ever interact with the "Help out your classmates!" window after following the "Make a post about it!" link in the OSBLE+ suggestion? (Second Image)

www.manaraa.com

1. Yes
2. No
3. I don't remember.

Please describe how you interacted with the "Let your classmates know how the assignment went!" window. **<OPEN ENDED>**

On a scale from 0-10, based on your experience using this suggestion, how likely are you to use this suggestion in the future?

IDE

Using the overview of the Visual Studio IDE with OSBLE+ suggestions for reference, please answer the following questions regarding the OSBLE+ suggestions in Visual Studio and the OSBLE+ dashboard.

www.manaraa.com

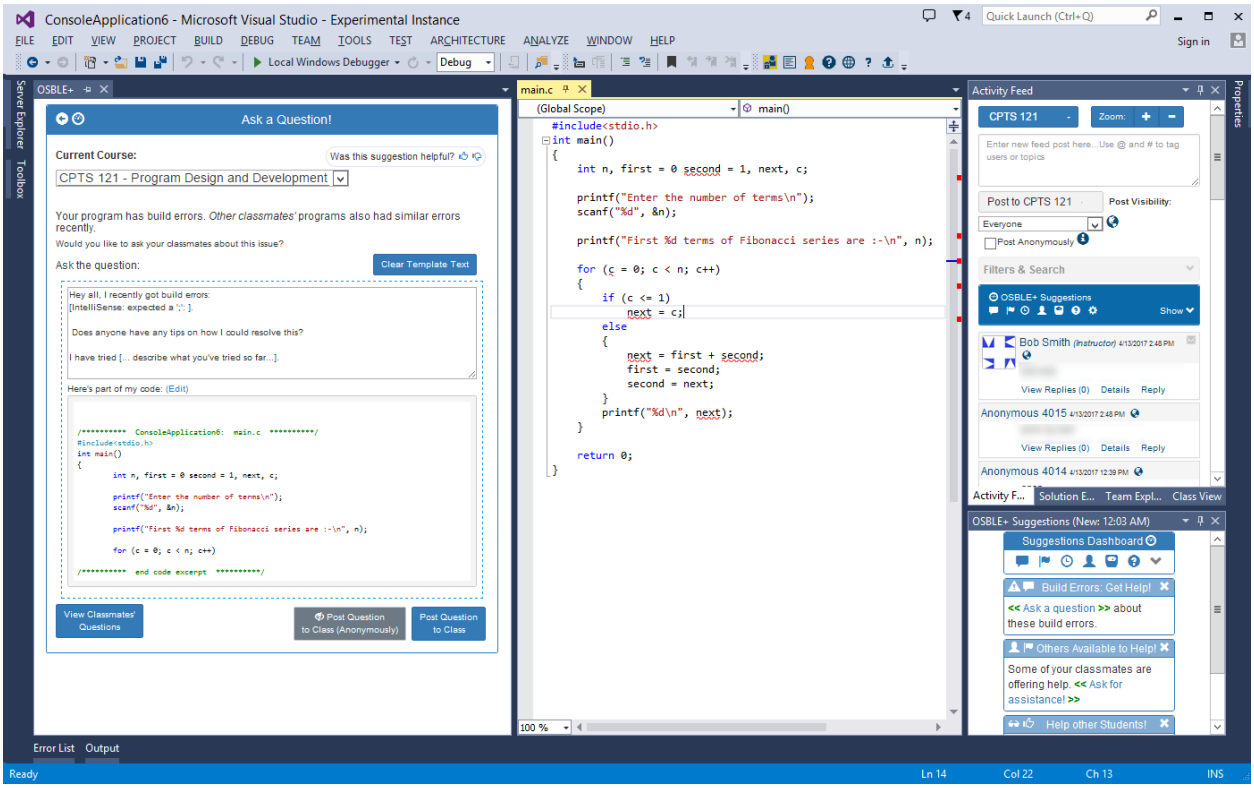

Figure 11.20. OSBLE Suggestions overview via the IDE

See [Figure 4.8. OSBLE Suggestions access via the OSBLE+ dashboard]

Select the statement which most accurately describes your use of the OSBLE+ Suggestions feature:

1. I used the OSBLE+ Suggestions feature from within Visual Studio most frequently
2. I used the OSBLE+ Suggestions feature directly on plus.osble.org via a web browser most frequently
3. I used the OSBLE+ Suggestions feature about the same frequency from within Visual Studio AND via a web browser.
4. Other (please elaborate):_____
5. I did not use the OSBLE+ Suggestions feature.

How frequently did you use the OSBLE+ Suggestions?

www.manaraa.com

1. Daily (11)
2. 4-6 times a week (12)
3. 2-3 times a week (13)
4. Once a week (14)
5. Never (15)

What MOST affected your usage of the OSBLE+ suggestions? (please describe) **<OPEN ENDED>**

What -one- thing would need to change in order to increase the likelihood of you using the OSBLE+ suggestions dashboard?) **<OPEN ENDED>**

Do you have any additional feedback regarding the OSBLE+ Suggestions feature?) **<OPEN ENDED>**

Regarding the OSBLE+ dashboard and Visual Studio plugin: Overall, how do you feel the use of the OSBLE+ dashboard and Visual Studio plugin affected your experience throughout the course?

1. Negative
2. Somewhat negative
3. Neutral
4. Somewhat positive
5. Positive

What change(s) would have made your experience positive instead of negative?) **<OPEN ENDED>**

www.manaraa.com

What change(s) would have made your experience positive?) **<OPEN ENDED>**

Regarding the OSBLE+ suggestions dashboard: Overall, how do you feel the use of the OSBLE+ suggestions affected your experience throughout the course?

1. Negative
2. Somewhat negative
3. Neutral
4. Somewhat positive
5. Positive

What change(s) would have made your experience positive instead of negative?) **<OPEN ENDED>**

What change(s) would have made your experience positive?) **<OPEN ENDED>**

If you have any further feedback, please leave your comments below before clicking 'NEXT') **<OPEN ENDED>**

www.manaraa.com

REFERENCES

Aljohani, N. R., & Davis, H. C. (2013). Learning Analytics and Formative Assessment to Provide Immediate Detailed Feedback Using a Student Centered Mobile Dashboard. *2013 Seventh International Conference on Next Generation Mobile Apps, Services and Technologies (NGMAST)*, 262–267. https://doi.org/10.1109/NGMAST.2013.54

Antin, J., & Churchill, E. F. (2011). Badges in social media: A social psychological perspective. *CHI 2011 Gamification Workshop Proceedings*, 1–4.

Astin, A. W. (1999). Student involvement: A developmental theory for higher education. *Journal of College Student Development*, *40*(5), 518–529.

Balaji, M. S., & Chakrabarti, D. (2010). Student Interactions in Online Discussion Forum: Empirical Research from 'Media Richness Theory' Perspective. *Journal of Interactive Online Learning*, *9*(1), 1–22.

Bandura, A. (1986). *Social foundations of thought and action : a social cognitive theory*. Englewood Cliffs, NJ, USA: Prentice Hall.

Bandura, A. (1990). Perceived self-efficacy in the exercise of personal agency. *Applied Sport Psychology*, *2*, 128–163.

Bandura, A. (1997). *Self-efficacy: the exercise of control*. New York: Worth Publishers.

Barker, L. J., McDowell, C., & Kalahar, K. (2009). Exploring factors that influence computer science introductory course students to persist in the major. *SIGCSE Bull.*, *41*(1), 153–157. https://doi.org/10.1145/1539024.1508923

www.manaraa.com

Barr, J., & Gunawardena, A. (2012). *Classroom salon: a tool for social collaboration*. 197. https://doi.org/10.1145/2157136.2157196

Bayliss, J. (2009). Using games in introductory courses: tips from the trenches. *Proceedings of the 40th ACM Technical Symposium on Computer Science Education*, 337–341. Chattanooga, TN, USA: ACM.

Beldarrain, Y. (2006). Distance Education Trends: Integrating new technologies to foster student interaction and collaboration. *Distance Education*, *27*(2), 139–153. https://doi.org/10.1080/01587910600789498

Bennedsen, J., & Caspersen, M. E. (2007). Failure rates in introductory programming. *ACM SIGCSE Bulletin*, *39*(2), 32–36. https://doi.org/10.1145/1272848.1272879

Biggers, M., Brauer, A., & Yilmaz, T. (2008). Student Perceptions of Computer Science: A Retention Study Comparing Graduating Seniors with Cs Leavers. *Proceedings of the 39th SIGCSE Technical Symposium on Computer Science Education*, 402–406. https://doi.org/10.1145/1352135.1352274

Black, P., & Wiliam, D. (1998). Assessment and Classroom Learning. *Assessment in Education: Principles, Policy & Practice*, *5*(1), 7–74. https://doi.org/10.1080/0969595980050102

Boyer, E. L., & Mitgang, L. D. (1996). *Building Community: A New Future for Architecture Education and Practice*. Princeton, NJ: The Carnegie Foundation for the Advancement of Teaching.

Braun, R., Brookes, W., Hadgraft, R., & Chaczko, Z. (2019). Assessment Design for Studio-Based Learning. *Proceedings of the Twenty-First Australasian Computing Education Conference*, 106–111. https://doi.org/10.1145/3286960.3286973

www.manaraa.com

Brennan, K. (2009). Scratch-Ed: an online community for scratch educators. *Proceedings of the 9th International Conference on Computer Supported Collaborative Learning*, *2*, 76–78. Rhodes, Greece: International Society of the Learning Sciences.

Brunswik, E. (1956). *Perception and the representative design of psychological experiments, 2nd ed*. Berkeley, CA, US: University of California Press.

Buffardi, K., & Edwards, S. H. (2013). Effective and Ineffective Software Testing Behaviors by Novice Programmers. *Proceedings of the Ninth Annual International ACM Conference on International Computing Education Research*, 83–90. https://doi.org/10.1145/2493394.2493406

Bureau of Labor Statistics, U.S. Department of Labor. (2017, May). Computer and Information Technology Occupations : Occupational Outlook Handbook: : U.S. Bureau of Labor Statistics. Retrieved April 11, 2019, from Bureau of Labor Statistics website: https://www.bls.gov/ooh/computer-and-information-technology/home.htm

Burgess, L. G., Riddell, P. M., Fancourt, A., & Murayama, K. (2018). The Influence of Social Contagion Within Education: A Motivational Perspective. *Mind, Brain, and Education*, *12*(4), 164–174. https://doi.org/10.1111/mbe.12178

Burguillo, J. C. (2010). Using game theory and Competition-based Learning to stimulate student motivation and performance. *Computers & Education*, *55*(2), 566–575. https://doi.org/10.1016/j.compedu.2010.02.018

Butz, C. J., Hua, S., & Maguire, R. B. (2006). A Web-based Bayesian Intelligent Tutoring System for Computer Programming. *Web Intelli. and Agent Sys.*, *4*(1), 77–97.

www.manaraa.com

Carrington, P. J., Scott, J., & Wasserman, S. (2005). *Models and Methods in Social Network Analysis*. Cambridge University Press.

Carter, A.S. (2016). *An Empirical Investigation of the Influence of Online Social and Programming Behaviors on Learning Outcomes in Early Computing Courses* (Dissertation). Washington State University.

Carter, A.S., & Hundhausen, C. D. (2011). A review of studio-based learning in computer science. *J. Comput. Sci. Coll.*, *27*(1), 105–111.

Carter, A.S., & Hundhausen, C. D. (2015). The Design of a Programming Environment to Support Greater Social Awareness and Participation in Early Computing Courses. *J. Comput. Sci. Coll.*, *31*(1), 143–153.

Carter, A.S., Hundhausen, C. D., & Adesope, O. (2015). The Normalized Programming State Model: Predicting Student Performance in Computing Courses Based on Programming Behavior. In *ICER '15. Proceedings of the Eleventh Annual International Conference on International Computing Education Research* (pp. 141–150). Retrieved from http://doi.acm.org/10.1145/2787622.2787710

Carter, J., Ala-Mutka, K., Fuller, U., Dick, M., English, J., Fone, W., … Sheard, J. (2003). How shall we assess this? *ACM SIGCSE Bulletin*, *35*, 107–123. https://doi.org/10.1145/960492.960539

Cassel, L. N., McGettrick, A., Davies, G., Topi, H., & Sloan, B. (2007). An initiative to attract students to computing. *SIGCSE Bull.*, *39*(1), 133–134.

Chickering, A. W., & Gamson, Z. F. (1987). Seven Principles for Good Practice in Undergraduate Education. *AAHE Bulletin*. Retrieved from http://eric.ed.gov/?id=ED282491

www.manaraa.com

Christe, B., & Feldhaus, C. (2013). Exploring Engineering Technology Persistence and Institutional Interventions: A Review of the Literature. *Journal of Engineering Technology*, *30*(2), 44.

Dasarathy, B., Sullivan, K., Schmidt, D. C., Fisher, D. H., & Porter, A. (2014). The Past, Present, and Future of MOOCs and Their Relevance to Software Engineering. *Proceedings of the on Future of Software Engineering*, 212–224. https://doi.org/10.1145/2593882.2593897

Dawson, S. (2008). A Study of the Relationship between Student Social Networks and Sense of Community. *Educational Technology & Society*, *11*(3), 224–238.

Del Fatto, V., Dodero, G., Gennari, R., Gruber, B., Helmer, S., & Raimato, G. (2018). Automating Assessment of Exercises as Means to Decrease MOOC Teachers' Efforts. In Ó. Mealha, M. Divitini, & M. Rehm (Eds.), *Citizen, Territory and Technologies: Smart Learning Contexts and Practices* (pp. 201–208). Springer International Publishing.

Deterding, S., Dixon, D., Khaled, R., & Nacke, L. (2011). From game design elements to gamefulness: defining "gamification." *Proceedings of the 15th International Academic MindTrek Conference: Envisioning Future Media Environments*, 9–15. https://doi.org/10.1145/2181037.2181040

Edwards, S. H., & Perez-Quinones, M. A. (2008). Web-CAT: Automatically Grading Programming Assignments. *Proceedings of the 13th Annual Conference on Innovation and Technology in Computer Science Education*, 328–328. https://doi.org/10.1145/1384271.1384371

Edwards, S. H., Snyder, J., Pérez-Quiñones, M. A., Allevato, A., Kim, D., & Tretola, B. (2009). Comparing Effective and Ineffective Behaviors of Student Programmers. *Proceedings of*

www.manaraa.com

the Fifth International Workshop on Computing Education Research Workshop, 3–14. https://doi.org/10.1145/1584322.1584325

Elliot, N. (2014, June 24). Facebook Still Dominates Teens' Social Usage [Blog]. Retrieved December 4, 2015, from Nate Elliott's Blog website: http://blogs.forrester.com/nate_elliott/14-06-24-facebook_still_dominates_teens_social_usage

Ferguson, R., & Shum, S. B. (2012). Social Learning Analytics: Five Approaches. *Proceedings of the 2Nd International Conference on Learning Analytics and Knowledge*, 23–33. https://doi.org/10.1145/2330601.2330616

Freeman, L. C. (1978). Centrality in social networks conceptual clarification. *Social Networks*, *1*(3), 215–239. https://doi.org/10.1016/0378-8733(78)90021-7

Fullan, M., & Pomfret, A. (1977). Research on Curriculum and Instruction Implementation. *Review of Educational Research*, *47*(2), 335–397. https://doi.org/10.2307/1170134

Gašević, D., Joksimović, S., Eagan, B. R., & Shaffer, D. W. (2018). SENS: Network analytics to combine social and cognitive perspectives of collaborative learning. *Computers in Human Behavior*. https://doi.org/10.1016/j.chb.2018.07.003

Gehringer, E. F. (2001). Electronic peer review and peer grading in computer science courses. In *Proceedings of the 32nd SIGCSE Technical Symposium on Computer Science Education* (pp. 139–143). New York: ACM Press.

Gehringer, E. F., Ehresman, L. M., & Skrien, D. J. (2006). Expertiza: students helping to write an ood text. In *OOPSLA '06: Companion to the 21st ACM SIGPLAN symposium on Object-*

www.manaraa.com

oriented programming systems, languages, and applications (pp. 901–906). Retrieved from http://research.csc.ncsu.edu/efg/expertiza/papers/oopslaes06.pdf

Gephi - The Open Graph Viz Platform. (n.d.). Retrieved December 29, 2018, from https://gephi.org/

Goldberg, L. R., Johnson, J. A., Eber, H. W., Hogan, R., Ashton, M. C., Cloninger, C. R., & Gough, H. G. (2006). The international personality item pool and the future of public-domain personality measures. *Journal of Research in Personality*, *40*(1), 84–96. https://doi.org/10.1016/j.jrp.2005.08.007

Goldman, M., Little, G., & Miller, R. C. (2011). Collabode: Collaborative coding in the browser. In *Proc. 4th Int. Workshop on Cooperative and Human Aspects of Software Engineering* (pp. 65–68). New York: ACM.

Gould, J. D., & Lewis, C. (1985). Designing for Usability: Key Principles and What Designers Think. *Commun. ACM*, *28*(3), 300–311. https://doi.org/10.1145/3166.3170

Graham, M. J., Federick, J., Byers-Winston, A., Hunber, A. B., & Handelsman, J. (2013). Increasing persistence of college students in STEM. *Science*, *341*(27 Sept.), 1455–1456.

Hartmann, B., MacDougall, D., Brandt, J., & Klemmer, S. R. (2010). What would other programmers do: suggesting solutions to error messages. In *Proc. 28th Conference on Human Factors in Computing Systems* (pp. 1019–1028). New York: ACM.

HELP Lab. (2012). Online Studio-Based Learning Environment. Retrieved January 14, 2012, from OSBLE website: https://osble.org/

Huang, J., Piech, C., Nguyen, A., & Guibas, L. (2013). Syntactic and Functional Variability of a Million Code Submissions in a Machine Learning MOOC. *1st Workshop on Massive Open*

www.manaraa.com

Online Courses. Presented at the 16th Annual Conference on Artificial Intelligence in Education.

Hundhausen, C. D., Agarwal, P., & Trevisan, M. (2011). Online vs. face-to-face pedagogical code reviews: An empirical comparison. In *Proceedings 2011 SIGCSE Symposium on Computer Science Education* (pp. 117–122). New York: ACM Press.

Hundhausen, C. D., Agrawal, A., & Agarwal, P. (2013). Talking about code: Integrating pedagogical code reviews into early computing courses. *ACM Transactions on Computing Education*, *13*(3), 14:1–14:28. https://doi.org/10.1145/2499947.2499951

Hundhausen, C. D., Agrawal, A., Fairbrother, D., & Trevisan, M. (2009). Integrating pedagogical code reviews into a CS 1 course: an empirical study. In *Proc. 40th ACM technical symposium on computer science education* (pp. 291–295). New York: ACM.

Hundhausen, C. D., Agrawal, A., & Ryan, K. (2010). The design of an online environment to support pedagogical code reviews. In *Proceedings of the 41st ACM technical symposium on Computer science education* (pp. 182–186). New York: ACM.

Hundhausen, C. D., & Brown, J. L. (2007). What You See Is What You Code: A "Live" Algorithm Development and Visualization Environment for Novice Learners. *Journal of Visual Languages and Computing*, *18*(1), 22–47.

Hundhausen, C. D., Brown, J. L., Farley, S., & Skarpas, D. (2006). A methodology for analyzing the temporal evolution of novice programs based on semantic components. In *Proceedings of the 2006 ACM International Computing Education Research Workshop* (pp. 45–56). New York: ACM Press.

www.manaraa.com

Hundhausen, C. D., & Carter, A. S. (2014). Facebook me about your code: An empirical study of the use of activity streams in early computing courses. *Journal of Computing Sciences in Colleges*, *30*(1), 151–160.

Hundhausen, C. D., Douglas, S. A., & Stasko, J. T. (2002). A meta-study of algorithm visualization effectiveness. *Journal of Visual Languages and Computing*, *13*(3), 259–290.

Hundhausen, C. D., Olivares, D. M., & Carter, A. S. (2017). IDE-Based Learning Analytics for Computing Education: A Process Model, Critical Review, and Research Agenda. *Trans. Comput. Educ.*

Hyyrynen, V., Hämäläinen, H., Ikonen, J., & Porras, J. (2010). MyPeerReview: An Online Peer-reviewing System for Programming Courses. *Proceedings of the 10th Koli Calling International Conference on Computing Education Research*, 94–99. https://doi.org/10.1145/1930464.1930481

Issa, G., Hussain, S. M., & Al-Bahadili, H. (2014). Competition-Based Learning: A Model for the Integration of Competitions with Project-Based Learning Using Open Source LMS. *Int. J. Inf. Commun. Technol. Educ.*, *10*(1), 1–13. https://doi.org/10.4018/ijicte.2014010101

Jadud, M. C. (2006). Methods and Tools for Exploring Novice Compilation Behaviour. In *ICER '06. Proceedings of the Second International Workshop on Computing Education Research* (pp. 73–84). Retrieved from http://doi.acm.org/10.1145/1151588.1151600

Jakobsson, M. (2011). Game Studies - The Achievement Machine: Understanding Xbox 360 Achievements in Gaming Practices. Retrieved September 28, 2012, from http://gamestudies.org/1101/articles/jakobsson

www.manaraa.com

Jenkins, J., Brannock, E., Cooper, T., Dekhane, S., Hall, M., & Nguyen, M. (2012). Perspectives on active learning and collaboration: JavaWIDE in the classroom. *Proceedings of the 43rd ACM Technical Symposium on Computer Science Education*, 185–190. Raleigh, NC, USA: ACM.

Johnson, J. (2010). *Designing with the Mind in Mind: Simple Guide to Understanding User Interface Design Rules*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.

Jung, E. H. (2018). Age Differences in Using Facebook: Younger Adults vs. Older Adults. *Proceedings of the 30th Australian Conference on Computer-Human Interaction*, 590–592. https://doi.org/10.1145/3292147.3292227

Kelleher, C., & Pausch, R. (2005). Lowering the Barriers to Programming: A Taxonomy of Programming Environments and Languages for Novice Programmers. *ACM Comput. Surv.*, *37*(2), 83–137. https://doi.org/10.1145/1089733.1089734

Kolb, A. Y., & Kolb, D. A. (2005). Learning Styles and Learning Spaces: Enhancing Experiential Learning in Higher Education. *Academy of Management Learning & Education*, *4*(2), 193–212.

Kolb, D. A. (1984). *Experiential learning: experience as the source of learning and development*. Retrieved from http://www.learningfromexperience.com/images/uploads/process-of-experiential-learning.pdf!(date of download: 31.05.2006)

Kölling, M., Quig, B., Patterson, A., & Rosenberg, J. (2003). The BlueJ system and its pedagogy. *Journal of Computer Science Education*, *13*(4), 249–268.

Kossinets, G., & Watts, D. J. (2006). Empirical Analysis of an Evolving Social Network. *Science*, *311*(5757), 88–90. https://doi.org/10.1126/science.1116869

www.manaraa.com

Kreijns, K., Kirschner, P. A., Jochems, W., & van Buuren, H. (2004). Determining Sociability, Social Space, and Social Presence in (A)synchronous Collaborative Groups. *CyberPsychology & Behavior*, *7*(2), 155–172. https://doi.org/10.1089/109493104323024429

Lambropoulos, N., Faulkner, X., & Culwin, F. (2012). Supporting social awareness in collaborative e-learning. *British Journal of Educational Technology*, *43*(2), 295–306. https://doi.org/10.1111/j.1467-8535.2011.01184.x

Lazowska, E. (2016, March 31). Where The Jobs Are – 2016 Edition » CCC Blog. Retrieved March 17, 2019, from https://www.cccblog.org/2016/03/31/where-the-jobs-are-2016-edition/

Lewis, C. M., Yasuhara, K., & Anderson, R. E. (2011). Deciding to Major in Computer Science: A Grounded Theory of Students' Self-assessment of Ability. *Proceedings of the Seventh International Workshop on Computing Education Research*, 3–10. https://doi.org/10.1145/2016911.2016915

Lipsey, M. W., Puzio, K., Yun, C., Hebert, M. A., Steinka-Fry, K., Cole, M. W., … Busick, M. D. (2012). *Translating the Statistical Representation of the Effects of Education Interventions into More Readily Interpretable Forms* (No. NCSER 2013-3000). Retrieved from U.S. Department of Education. Institute of Education Sciences, National Center for Education Statistics website: https://ies.ed.gov/ncser/pubs/20133000/

Ma, W., Adesope, O. O., Nesbit, J. C., & Liu, Q. (2014). Intelligent tutoring systems and learning outcomes: A meta-analytic survey. *Journal of Educational Psychology*, *106*(2007), 901–918.

www.manaraa.com

Macqueen, J. (1967). Some methods for classification and analysis of multivariate observations. *In 5-Th Berkeley Symposium on Mathematical Statistics and Probability*, 281–297.

Malone, T. W. (1981). Toward a theory of intrinsically motivating instruction. *Cognitive Science*, *5*(4), 333–369. http://dx.doi.org/10.1016/S0364-0213(81)80017-1

Maloney, J., Resnick, M., Rusk, N., Silverman, B., & Eastmond, E. (2010). The Scratch programming language and environment. *ACM Transactions on Computing Education*, *10*(4), 1–15.

Mann, S., & Frew, Z. (2006). Similarity and Originality in Code: Plagiarism and Normal Variation in Student Assignments. *Proceedings of the 8th Australasian Conference on Computing Education - Volume 52*, 143–150. Retrieved from http://dl.acm.org/citation.cfm?id=1151869.1151888

Martin, J., Edwards, S. H., & Shaffer, C. A. (2015). The Effects of Procrastination Interventions on Programming Project Success. *Proceedings of the Eleventh Annual International Conference on International Computing Education Research*, 3–11. https://doi.org/10.1145/2787622.2787730

McDowell, C., Werner, L., Bullock, H. E., & Fernald, J. (2006). Pair Programming Improves Student Retention, Confidence, and Program Quality. *Commun. ACM*, *49*(8), 90–95. https://doi.org/10.1145/1145287.1145293

Miller, K., Zyto, S., Karger, D., & Mazur, E. (2014). Improving Online Class Forums by Seeding Discussions and Managing Section Size. *Proceedings of the First ACM Conference on Learning @ Scale Conference*, 173–174. https://doi.org/10.1145/2556325.2567866

www.manaraa.com

Mills, S. C., & Ragan, T. J. (2000). A tool for analyzing implementation fidelity of an integrated learning system. *Educational Technology Research and Development*, *48*(4), 21–41. https://doi.org/10.1007/BF02300498

Mujumdar, D., Hallenbach, M., Liu, B., & Hartmann, B. (2011). Crowdsourcing suggestions to programming problems for dynamic web development languages. *CHI '11 Extended Abstracts on Human Factors in Computing Systems*, 1525–1530. https://doi.org/10.1145/1979742.1979802

National Center for Education Statistics, U.S. Department of Education. (2011). BPS: 2009 Beginning Postsecondary Students. Retrieved November 19, 2011, from http://nces.ed.gov/surveys/bps/

Nepal, S., Paris, C., & Bista, S. (2015). Gamification on the Social Web. In S. Nepal, C. Paris, & D. Georgakopoulos (Eds.), *Social Media for Government Services* (pp. 197–220). https://doi.org/10.1007/978-3-319-27237-5_10

Nesbit, J. C., Adesope, O. O., Liu, Q., & Ma, W. (2014). How Effective are Intelligent Tutoring Systems in Computer Science Education? *2014 IEEE 14th International Conference on Advanced Learning Technologies (ICALT)*, 99–103. https://doi.org/10.1109/ICALT.2014.38

Ni, L. (2009). What makes CS teachers change?: factors influencing CS teachers' adoption of curriculum innovations. *ACM SIGCSE Bulletin*, *41*(1), 544–548. https://doi.org/10.1145/1508865.1509051

www.manaraa.com

Ni, L., McKlin, T., & Guzdial, M. (2010, March 10). *How do computing faculty adopt curriculum innovations?: the story from instructors*. 544–548. https://doi.org/10.1145/1734263.1734444

Nielsen, J. (1993). Iterative User-Interface Design. *Computer*, *26*(11), 32–41. https://doi.org/10.1109/2.241424

Norman, D. A. (2013). *The Design of Everyday Things: Revised and Expanded Edition*. New York: Basic Books.

Ntourmas, A., Avouris, N., Daskalaki, S., & Dimitriadis, Y. (2018). Teaching Assistants' Interventions in Online Courses: A Comparative Study of Two Massive Open Online Courses. *Proceedings of the 22Nd Pan-Hellenic Conference on Informatics*, 288–293. https://doi.org/10.1145/3291533.3291563

O'Grady, M. J. (2012). Practical Problem-Based Learning in Computing Education. *Trans. Comput. Educ.*, *12*(3), 10:1–10:16. https://doi.org/10.1145/2275597.2275599

Okoshi, T., Ramos, J., Nozaki, H., Nakazawa, J., Dey, A. K., & Tokuda, H. (2015). Reducing Users' Perceived Mental Effort Due to Interruptive Notifications in Multi-device Mobile Environments. *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, 475–486. https://doi.org/10.1145/2750858.2807517

Olivares, D. (2015). Exploring Learning Analytics for Computing Education. *Proceedings of the Eleventh Annual International Conference on International Computing Education Research*, 271–272. https://doi.org/10.1145/2787622.2787746

Olivares, D., Adesope, O., Hundhausen, C., Ferreira, R., & Gašević, D. (in press). Using social network analysis to measure the effect of learning analytics in computing education.

www.manaraa.com

Proceedings of the 19th IEEE International Conference on Advanced Learning Technologies (ICALT). IEEE.

Olivares, D. M., & Hundhausen, C. D. (2016). OSBLE+: A next-generation learning management and analytics environment for computing education. In *Proceedings of the 47th ACM Technical Symposium on Computer Science Education* (p. 5). Retrieved from http://dx.doi.org/10.1145/2839509.2850509

Olivares, D. M., & Hundhausen, C. D. (2017). Supporting learning analytics in computing education. *Proceedings 7th International Learning Analytics & Knowledge Conference*. Presented at the Learning Analytics & Knowledge conference, Vancouver, BC, Canada.

Oman, P. W., & Cook, C. R. (1990). A Taxonomy for Programming Style. *Proceedings of the 1990 ACM Annual Conference on Cooperation*, 244–250. https://doi.org/10.1145/100348.100385

Pieterse, V., & Liebenberg, J. (2017, November 16). *Automatic vs manual assessment of programming tasks*. 193–194. https://doi.org/10.1145/3141880.3141912

Pintrich, D., Smith, D., Garcia, T., & McKeachie, W. (1991). *A manual for the use of the motivated strategies for learning questionnaire* (Technical Report No. NCRIPTAL-91-B-004). Retrieved from National Center for Research to Improve Postsecondary Teaching and Learning website: http://eric.ed.gov/ERICDocs/data/ericdocs2sql/content_storage_01/0000019b/80/23/3c/44.pdf

Politz, J. G., Patterson, D., Krishnamurthi, S., & Fisler, K. (2014). CaptainTeach: Multi-stage, In-flow Peer Review for Programming Assignments. *Proceedings of the 2014 Conference on*

www.manaraa.com

Innovation & Technology in Computer Science Education, 267–272. https://doi.org/10.1145/2591708.2591738

Porter, L., Bouvier, D., Cutts, Q., Grissom, S., Lee, C., McCartney, R., … Simon, B. (2016). A Multi-institutional Study of Peer Instruction in Introductory Computing. *Proceedings of the 47th ACM Technical Symposium on Computing Science Education*, 358–363. https://doi.org/10.1145/2839509.2844642

Porter, L., Lee, C. B., Simon, B., & Zingaro, D. (2011). Peer instruction: do students really learn from peer discussion in computing? In *Proceedings of the seventh international workshop on Computing education research* (pp. 45–52). New York: ACM.

Prause, C. R., & Jarke, M. (2015). Gamification for Enforcing Coding Conventions. *Proceedings of the 2015 10th Joint Meeting on Foundations of Software Engineering*, 649–660. https://doi.org/10.1145/2786805.2786806

Prior, J., Laudari, S., & Leaney, J. (2019). What is the Effect of a Software Studio Experience on a Student's Employability? *Proceedings of the Twenty-First Australasian Computing Education Conference*, 28–36. https://doi.org/10.1145/3286960.3286964

Radermacher, A. D., & Walia, G. S. (2011). Investigating the effective implementation of pair programming: An empirical investigation. *Proceedings of the 42Nd ACM Technical Symposium on Computer Science Education*, 655–660. https://doi.org/10.1145/1953163.1953346

Ramalingam, V., & Weidenbeck, S. (1998). Development and validation of scores on a computer programming self-efficacy scale and group analyses of novice programmer self-efficacy. *J. Educational Computing Research*, *19*(4), 367–381.

www.manaraa.com

Reily, K., Finnerty, P. L., & Terveen, L. (2009). Two peers are better than one: aggregating peer reviews for computing assignments is surprisingly accurate. *Proceedings of the ACM 2009 International Conference on Supporting Group Work*, 115–124. Retrieved from http://doi.acm.org/10.1145/1531674.1531692

Resnick, M., Maloney, J., Monroy-Hernández, A., Rusk, N., Eastmond, E., Brennan, K., … Kafai, Y. (2009). Scratch: programming for all. *Commun. ACM*, *52*(11), 60–67.

Rodger, S. H., Bashford, M., Dyck, L., Hayes, J., Liang, L., Nelson, D., & Qin, H. (2010). Enhancing K-12 education with Alice programming adventures. *Proceedings of the Fifteenth Annual Conference on Innovation and Technology in Computer Science Education*, 234–238. Bilkent, Ankara, Turkey: ACM.

Rößling, G., Joy, M., Moreno, A., Radenski, A., Malmi, L., Kerren, A., … Iturbide, J. Á. V. (2008). Enhancing Learning Management Systems to Better Support Computer Science Education. *SIGCSE Bull.*, *40*(4), 142–166. https://doi.org/10.1145/1473195.1473239

Rosson, M. B., Carroll, J. M., & Sinha, H. (2011). Orientation of Undergraduates Toward Careers in the Computer and Information Sciences: Gender, Self-Efficacy and Social Support. *Trans. Comput. Educ.*, *11*(3), 1–23.

Rotter, J. B. (1966). Generalized expectancies for internal versus external control of reinforcement. *Psychological Monographs: General and Applied*, *80*(1), 1–28. https://doi.org/10.1037/h0092976

Rovai, A. P. (2002). Development of an instrument to measure classroom community. *Internet and Higher Education*, *5*, 197–211.

www.manaraa.com

Ryan, A. M., Pintrich, P. R., & Midgley, C. (2001). Avoiding Seeking Help in the Classroom: Who and Why? *Educational Psychology Review*, *13*(2), 93–114. https://doi.org/10.1023/A:1009013420053

Salinger, S., Oezbek, C., Beecher, K., & Schenk, J. (2010). Saros: an eclipse plug-in for distributed party programming. *Proceedings of the 2010 ICSE Workshop on Cooperative and Human Aspects of Software Engineering*, 48–55. Retrieved from http://doi.acm.org/10.1145/1833310.1833319

Schlenker, B. R., Schlenker, P. A., & Schlenker, K. A. (2013). Antecedents of academic engagement and the implications for college grades. *Learning and Individual Differences*, *27*, 75–81. https://doi.org/10.1016/j.lindif.2013.06.014

Schmidt, H. G., Molen, H. T. V. D., Winkel, W. W. R. T., & Wijnen, W. H. F. W. (2009). Constructivist, Problem-Based Learning Does Work: A Meta-Analysis of Curricular Comparisons Involving a Single Medical School. *Educational Psychologist*, *44*(4), 227–249. https://doi.org/10.1080/00461520903213592

Schmidt, J. P., Geith, C., Håklev, S., & Thierstein, J. (2009). Peer-To-Peer Recognition of Learning in Open Education. *The International Review of Research in Open and Distributed Learning*, *10*(5). Retrieved from http://www.irrodl.org/index.php/irrodl/article/view/641

Schön, D. (1983). *The reflective practitioner: How professionals think in action*. New York: Basic Books.

Scott, J. (2017). *Social Network Analysis*. SAGE.

www.manaraa.com

Selim, H. M. (2007). Critical Success Factors for E-Learning Acceptance: Confirmatory Factor Models. *Computers & Education*, *49*(2), 396–413. https://doi.org/10.1016/j.compedu.2005.09.004

Shadish, W. R., Cook, T. D., & Campbell, D. T. (2002). *Experimental and Quasi-Experimental Designs for Generalized Causal Inference*. Boston: Houghton Mifflin Company.

SmartBear Software. (2012). CodeCollaborator. Retrieved January 13, 2012, from http://smartbear.com/products/development-tools/code-review/

Stephenson, C., Miller, A. D., Alvarado, C., Barker, L., Barr, V., Camp, T., … Zweben, S. (2018). *Retention in Computer Science Undergraduate Programs in the U.S.: Data Challenges and Promising Interventions.* New York, NY: ACM.

Teusner, R., Hille, T., & Staubitz, T. (2018). Effects of Automated Interventions in Programming Assignments: Evidence from a Field Experiment. *Proceedings of the Fifth Annual ACM Conference on Learning at Scale*, 60:1–60:10. https://doi.org/10.1145/3231644.3231650

Tinto, V. (1997). Classrooms as Communities: Exploring the Educational Character of Student Persistence. *The Journal of Higher Education*, *68*(6), 599–623. https://doi.org/10.2307/2959965

Trochim, W. M. K. (2006). Quasi-Experimental Design. Retrieved April 9, 2016, from Quasi-Experimental Design website: http://www.socialresearchmethods.net/kb/quasiexp.php

U.S. Department of Education, National Center for Education Statistics, Integrated Postsecondary Education Data System (IPEDS), Completions Survey. (2017). Retrieved March 17, 2019, from https://nces.ed.gov/ipeds/use-the-data

www.manaraa.com

Verbert, K., Duval, E., Klerkx, J., Govaerts, S., & Santos, J. L. (2013). Learning analytics dashboard applications. *American Behavioral Scientist*, *57*(10), 1500–1509. https://doi.org/10.1177/0002764213479363

Verbert, K., Govaerts, S., Duval, E., Santos, J. L., Assche, F., Parra, G., & Klerkx, J. (2014). Learning Dashboards: An Overview and Future Research Opportunities. *Personal Ubiquitous Comput.*, *18*(6), 1499–1514. https://doi.org/10.1007/s00779-013-0751-2

Verbert, K., Govaerts, S., Duval, E., Santos, J. L., Van Assche, F., Parra, G., & Klerkx, J. (2013). Learning dashboards: an overview and future research opportunities. *Personal and Ubiquitous Computing*. https://doi.org/10.1007/s00779-013-0751-2

Vogt, C. M. (2008). Faculty as a Critical Juncture in Student Retention and Performance in Engineering Programs. *Journal of Engineering Education*, *97*(1). https://doi.org/10.1002/j.2168-9830.2008.tb00951.x

Vygotsky, L. S. (1978). *Mind in Society*. Cambridge, MA: Harvard University Press.

Wang, Y., & Li, X. (2007). Social Network Analysis of Interaction in Online Learning Communities. *Seventh IEEE International Conference on Advanced Learning Technologies (ICALT 2007)*, 699–700. https://doi.org/10.1109/ICALT.2007.228

Wasserman, S., & Faust, K. (1994). *Social Network Analysis: Methods and Applications*. Cambridge University Press.

Watson, C., & Li, F. W. B. (2014). Failure Rates in Introductory Programming Revisited. *Proceedings of the 2014 Conference on Innovation & Technology in Computer Science Education*, 39–44. https://doi.org/10.1145/2591708.2591749

www.manaraa.com

Watson, C., Li, F. W. B., & Godwin, J. L. (2013). Predicting Performance in an Introductory Programming Course by Logging and Analyzing Student Programming Behavior. *Proceedings of the 2013 IEEE 13th International Conference on Advanced Learning Technologies*, 319–323. IEEE Computer Society.

Watson, C., Li, F. W. B., & Godwin, J. L. (2014). No Tests Required: Comparing Traditional and Dynamic Predictors of Programming Success. In *SIGCSE '14. Proceedings of the 45th ACM Technical Symposium on Computer Science Education* (pp. 469–474). Retrieved from http://doi.acm.org/10.1145/2538862.2538930

Weiner, B. (1979). A theory of motivation for some classroom experiences. *Journal of Educational Psychology*, *71*(1), 3–25. https://doi.org/10.1037/0022-0663.71.1.3

Wilson, M., & Scalise, K. (2006). Assessment to improve learning in higher education: The BEAR Assessment System. *Higher Education*, *52*(4), 635–663. https://doi.org/10.1007/s10734-004-7263-y

Wood, D., Bruner, J. S., & Ross, G. (1976). The role of tutoring in problem solving. *Journal of Child Psychology and Psychiatry*, *17*(2), 89–100. https://doi.org/10.1111/j.1469-7610.1976.tb00381.x

Yu, T., & Jo, I.-H. (2014). Educational Technology Approach Toward Learning Analytics: Relationship Between Student Online Behavior and Learning Performance in Higher Education. *Proceedings of the Fourth International Conference on Learning Analytics And Knowledge*, 269–270. https://doi.org/10.1145/2567574.2567594

www.manaraa.com

Reproduced with permission of copyright owner. Further reproduction prohibited without permission.