TEACHING SOFTWARE ENGINEERING FOR THE MODERN ENTERPRISE

THESIS

Presented in Partial Fulfillment of the Requirements for the Degree Master of Science in the Graduate School of The Ohio State University

By

Michael J. Herold

Graduate Program in Computer Science and Engineering

The Ohio State University

2013

Master's Examination Committee:

Rajiv Ramnath, Advisor

Jayashree Ramanathan
Copyright by
Michael J. Herold
2013
Abstract

Software engineering education traditionally relies on a standard lecture-and-readings format to teach students about the discipline. This paradigm is comfortable for instructors, as it is the accepted standard in most disciplines in higher education. However, software engineering contains many subjects that rely on tacit knowledge. By definition, this knowledge is difficult to transfer through traditional communication, such as reading and lecture, instead necessitating practice and learning-by-doing. Furthermore, traditional software engineering courses do not teach students business concepts, instead couching the program in core computer science and the study of abstraction. Without understanding business context, students are less able to see the impact that their work makes throughout their careers and require more on-the-job training in their initial positions once out of school. This research addresses the problems of using traditional methods for teaching software engineering and excising business education from the technical discipline. Two contributions are made.

First, this thesis describes a novel curriculum for teaching enterprise software engineering. By grounding the curriculum in a constructivist philosophy, the curriculum transforms the instructor into a facilitator for learning, rather than a lecturer. The course is restructured as an inverted classroom, pushing the concept of lecture into the student's study time and focusing the in-class time on active learning games, discussion, and case studies. This allows the course to contain more material, enabling it to begin with two
units focused on business context. This refocusing encourages students to think beyond the software, considering how their work affects the position of the business and the people that will use it.

Second, an a portfolio of curriculum evaluation measures its efficacy and the strength of the individual teaching methods used within. This evaluation contains three separate studies. The first study is a preliminary evaluation of the course that verifies its effectiveness. Additionally, the study elicits research questions for further examination. The second study is an in-depth look at the experiences both students and instructors have in the inverted classroom. By evaluating the individual components of the course, including the active learning games, discussion, and prerecorded lectures, this inquiry analyzes student reactions, the effects on instructor preparation, and acceptance of the format. The third study measures the effectiveness of the curriculum's major active learning game, a Lego-based workshop on agile development. Through using a questionnaire that measures student engagement and tracks their internalization of the course concepts, this study shows that the active learning game is as effective as the prerecorded lectures for teaching agile development concepts. This is a significant result because the concepts the workshop teaches are known to be tacit knowledge that is difficult to transfer in traditional fashion. The thesis then concludes with an analysis of future work to improve the curriculum and better measure its efficacy.
Dedication

To my fiancée, whose love and support show me the way.
Acknowledgments

First, I would to thank my advisor, Dr. Rajiv Ramnath for his support and encouragement throughout my graduate career. My road was a bumpy one and he helped me to make it through the worst. I learned more about the context in which software operates, and the way that it affects those who use it, than I ever thought I would thanks to his support and guidance over the years. Rajiv, I hope that your enthusiasm lasts for many years to come and that you help many more students find their way.

I also would like to thank my co-advisor, Dr. Jay Ramanathan. Despite sometimes feeling like we were speaking a different language, I always enjoyed hearing her viewpoint on things and struggling to see the world through her eyes. Best of luck trying to solve the grand problems of the world, as you have always insisted we try to do. Your guidance will help your students reach higher than they think possible.

I also want to thank the students and CETI members with whom I have worked over the past couple of years. I know I will forget some, as there are so many, so if I do forget your name here, never doubt that I appreciated all of your help and input. For co-authoring with me and showing me that I am much more of a small-town guy than I thought (I really did enjoy South Dakota, even if you did not!), I want to thank Joe Bolinger and Tom Lynch. For taking the reins on the SWSS project and following it through, I want to thank SaranyaDevi Ganesan. Chaitanya Shivade, Satyajeet Raje, Zhe
Xu, Hyun Jeong Yoo, Heesung Lee, and Kayhan Moharerri were always there to bounce ideas off of and vent frustrations to.

Lastly, I would like to thank my family and friends. My parents have always been supportive of me in everything that I do. My friends help remind me that I am, in fact, human and cannot do everything, but are always there to talk when I need them. And my fiancée, Kate; I know I could not have made it through graduate school without your love, support, and most importantly, patience. You help me be a better person, pointing out the good that I do with my work even when I don’t see it. Thank you so much for everything that you do for me; I’m so excited to begin our life together.
Vita

2005 ..................................................... Pickerington High School North

2009 ..................................................... B.A. Computer Science and Mathematics,
Capital University

2009 to present ..................................... Graduate Teaching/Research Associate,
Department of Computer Science and
Engineering, The Ohio State University

Publications


**Fields of Study**

**Major Field: Computer Science and Engineering**
Table of Contents

Abstract .................................................................................................................................................. ii
Dedication .......................................................................................................................................... iv
Acknowledgments ................................................................................................................................. v
Vita ....................................................................................................................................................... vii
List of Tables ....................................................................................................................................... xi
List of Figures ....................................................................................................................................... xii
Chapter 1: Introduction ......................................................................................................................... 1
 Contributions ........................................................................................................................................ 4
 Document Organization ....................................................................................................................... 5
Chapter 2: Background and Related Work ............................................................................................ 6
 The Inverted Classroom ......................................................................................................................... 6
 Distance Learning ................................................................................................................................. 11
 Case-based Learning ............................................................................................................................ 15
 Active Learning .................................................................................................................................... 19
Chapter 3: Enterprise Software Engineering ......................................................................................... 22
 Prerecorded Lectures and Quizzes ....................................................................................................... 22
List of Tables

Table 1: An abbreviated list of course topics with mappings from two cases............... 42
Table 2: Descriptive statistics for the student surveys.............................................. 68
Table 3: The means and p-values for the ANOVA between quiz scores. ...................... 75
Table 4: Descriptive statistics analyzing student acceptance of the agile workshop........ 77
List of Figures

Figure 1: An example of the old slide format versus the new. ............................................. 24
Figure 2: The three paradigms of podcast lectures. ................................................................. 26
Figure 3: Two examples of the Lego animals built in the active learning workshop. ...... 33
Figure 4: An outline of the case study creation method. ......................................................... 45
Figure 5: Student responses measuring enjoyment in the agile workshop. ....................... 78
Figure 6: Percent of students picking the correct answer on the agile questionnaire. ...... 79
Figure 7: Students’ exam question grades. ............................................................................ 80
Chapter 1: Introduction

Traditional software engineering curricula focus mainly on small-scale system development following structured processes, as taught in the canonical software engineering textbooks (Pressman, 2009; Sommerville, 2010). This narrow concentration leads students to naïveté when considering software processes; they either follow one prescribed process without considering its implications or eschew the process entirely (Ghezzi, 2005; van Vilet, 2006). Both of these process implementations are insufficient in real practice, leading either to large amounts of rework or inferior end products. By not understanding or not implementing a software process, the work that software engineers do may not produce a return-on-investment for the business. This business return-on-investment is ultimately the requisite outcome of any enterprise software engineering project and should thus be the focus of such processes (Ramnath, 2006). Currently, this is not made salient in many software engineering curricula.

The focus on either bleeding-edge, fashionable processes or tried-and-true, traditional processes both illuminate the source of the problem within software engineering education: there is little focus on core, foundational principles. Teaching specific processes can be useful, as it gives students reference processes to rely on, but the real value of this teaching method comes when the process is grounded in core principles, with an understanding of why a process works as it does, instead of just how the process looks when in action.
By refocusing their teaching on principles, educators actually allow themselves more flexibility and time-savings when preparing to teach a course. If lessons are structured around principles, then the instructor can weave in material from a variety of processes, which are free to change each time the course is taught. This ability to drop in, at any time, replacement processes that demonstrate the principles, increases student exposure to a variety of processes. Also, by consistently changing the processes used in the course, the instructor keeps the course up-to-date, fresh, and exciting not only for the students, but for himself. This freshness can help keep both instructor and students engaged in the course, thus making the teaching and learning experience less onerous and improving learning outcomes.

In addition to being inflexible, traditional programs do not adequately prepare students for their initial jobs in several key areas, including software and process engineering (Hawthorne, 2005). By bringing the focus of software engineering courses to a set of core, foundational principles, educators enable students to be more flexible. This flexibility allows new software engineers to make an immediate impact in their initial jobs, which in turn can improve industry opinion of the student’s educational program.

There is a need for a new model of teaching software engineering. Research has shown the need for contextualized education in software engineering (Bareiša et al., 2007; Fendler & Winschiers-Theophilus, 2010). Educators better prepare their students for their careers by contextualizing software engineering education within the business context. Understanding the effects their decisions make to the whole business entity is crucial to the initial success of fresh graduates. In particular, the typical computer science
graduates do not understand the value chain (Porter, 1998), or flow of resources into, through and out of the business. Without this understanding, the contributions they make to their companies are unclear and it is difficult to see the impact that they make.

In order to address this need for contextualized education, software engineering educators must embrace social aspects of software engineering with the philosophy of constructivism (Vygotsky, 1978). Software engineering is, at its heart, a social discipline concerned with the workings of people amongst software systems. As such, software engineering education should embrace the social aspect of the discipline, relying on interactions between the students and instructors to shape the way that instruction occurs.

Some educators have experimented with constructivist methods for teaching software engineering, using concept scaffolds to guide student-driven learning (Ludi, 2005). However, scaffolding is inadequate without the support of peers and instructors and without the focus on delivering value to the stakeholders of a computer system. Software engineering education, as a field, must tap its industry resources and integrate the concepts of business value and software processes to produce student understanding of their place in the wider world of software consumers.

To do this, they must innovate on the educational methods that are used in software engineering courses. Without a sea change in the educational paradigm from the lecture-based, narrowly-focused curricula of the past to new, interactive, engaging methods of producing long-term, integrated understanding, students will not be best served by learning the software engineering field from academia. The reliance on old
educational techniques and patterns will limit the impact that fresh graduates can immediately make without extensive training after graduation.

**Contributions**

This research examines the current status of software engineering education and addresses the lack of focus on core principles within the discipline and the grounding of software engineering within the larger context of the business. This thesis focuses on one main contribution, with a small set of minor contributions encapsulated within.

The overarching contribution is a software engineering curricula that is grounded within the context of the parent enterprise and focused on a set of software engineering principles that guide the teaching of software processes. By grounding the curriculum within the business context, students learn to make decisions that maximize the business value of developed software and increase the return-on-investment of information technology. Focusing on software engineering principles removes the tight coupling between specific processes and student learning, thus broadening students’ understanding and enabling them to be more introspective when identifying the software process to use in a given circumstance. This enables students to make immediate impact in their first jobs, increasing their value to their employers and their subsequent careers.

Teaching enterprise software engineering with traditional instruction techniques is difficult because these techniques are too classroom time-intensive to allow all of the curriculum content to be taught in one course. To counter this, the curriculum uses non-traditional techniques that allow for more information density and greater hands-on practice to help the students learn the material in a shortened amount of in-class time.
These techniques include the use of an inverted classroom format and active learning games for specific parts of the curriculum. The minor contributions of these techniques form a solid basis for teaching the software engineering curriculum and allowing students to internalize concepts for use later in their careers.

**Document Organization**

Chapter 2 begins with a brief overview of educational methods that are employed by the software engineering curriculum, then defines some basic terminology used within this thesis. Next, Chapter 3 describes the software engineering curriculum as designed from this research and discusses the rationale for the methods used within the curriculum. Following that, Chapter 4 presents some evaluation of the curriculum from the joint perspectives of the students and the instructors. Finally, Chapter 5 summarizes the contributions of this research and outlines some possible future work.
Chapter 2: Background and Related Work

The complexity of teaching software engineering is ever increasing. Employers look for specific skills when hiring fresh graduates, meaning that a principles-based software engineering course must ground itself in a subset of the latest technologies being used in industry. As such, instructors must increase the amount of material that is taught within a course. However, courses are time-boxed into academic terms (e.g. quarters or semesters) that are a fixed length. The increased level of material coupled with the fixed term length necessitates some innovative teaching methods to adequately cover the course modules in such a way that students actually retain the information. To combat this increased level of material, our novel curriculum relies on the inverted classroom. The following section discusses the literature around the inverted classroom and supports the decision to use it.

The Inverted Classroom

Traditional classrooms follow a lecture format that is instructor-centered, meaning that contact hours (i.e. hours that students are in contact with the instructor, whether for lecture, office hours, laboratory sessions or examinations) primarily consist of the instructor communicating information to the students. This structure maximizes the amount of time that instructors have to speak to their students, but limits the time available for lecture alternatives, like active learning activities, discussion, and in-class work.
When designing a curriculum, instructors need to take different learning styles into account. Often, this is done via the VARK (Visual, Auditory, Read/write, Kinesthetic) model, which describes a model of learning styles separated by modality (Fleming & Mills, 1992). Students learn differently in each modality, with most students learning best within one or two modalities. By limiting the style of teaching to lectures, instructors appeal mostly to auditory learners, which leaves students who learn best in different modalities at a disadvantage compared to their auditory-learning peers.

There are many techniques that can help instructors teach students of all learning styles, and much research has been done to outline some basic principles for achieving this. Often these principles expand upon the VARK model and incorporate pieces of other learning models as well (Felder & Silverman, 1988). The key to each set of teaching principles is variety in teaching modality; by not relying solely on the lecture format, instructors can vary the way that they convey information to students, thus appealing to a variety of learning styles.

However, variety in instruction delivery takes more time than a pure lecture format. As such, educators continue to innovate in their delivery methods to allow them to teach everything needed for a subject in a multi-modal fashion. Starting in a Principles of Economics course, one such innovation, called the inverted classroom, relies on lecture in a non-traditional format: recorded lectures watched outside contact hours (Lage, Platt, & Treglia, 2000). By recording lectures, usually in a multi-modal Microsoft PowerPoint presentation with voiceovers, educators allow students to consume lectures
during their study hours, re-watch lectures in which the student struggled to learn to
concepts, and open up contact hours to non-traditional teaching formats.

The inverted classroom has been adapted for use in many different environments,
for different reasons, including: teaching lifelong learning skills (Bland, 2006), allowing
self-pacing of course content (Boutell & Clifton, 2011), restructuring an entire program
curriculum (Gannod, Burge, & Helmick, 2008), teaching specific courses in which
students routinely struggle (Papadopoulos & Rico, 2010), and opening up traditional
lecture courses to more active learning (Toto & Nguyen, 2009).

When the requirements for accreditation by the Accreditation Board for
Engineering and Technology, Inc., (ABET) added a component for enabling lifelong
learning in students (Engineering Accreditation Commission, 2013, p. 3), an Electrical
Engineering program started using the inverted classroom to establish lifelong learning
practices in its students (Bland, 2006). Bland felt that the inverted classroom format
offers characteristics that are more in-line with student-directed learning, which aids in
teaching learning skills. The implementation of the inverted classroom showed several
positive results including faster progression through course content, an increase in study
and learning skills, and high approval and adoption by other instructors, at the expense,
however, of a higher level of frustration in students. The faster progression through
content and high levels of frustration led the author to the conclusion that the method is
not suitable for use in all subjects, as complex subjects require significantly better course
materials for positive results.
In introductory programming courses, there is a wide range of ability, ranging from novices who are learning to program for the first time, to intermediate students who have learned to program through self-teaching or previous instruction. This range leads to many students struggling to keep up with the pace of the course, due to the limited time available for each concept. To combat this, Boutell & Clifton (2011) implemented an inverted classroom format to allow students to review material when they felt the need and keep up with the pace of the course. This allows the use of the 3x2 format (meaning three “lecture” sessions and two “laboratory” sessions per week) more time for active learning, such as programming exercises, experimentation with the new content, and guided application of the content to a real-world problem. A secondary benefit of the inverted classroom format comes from the ability to disseminate the teaching material to an external audience, which garners attention to the program without significant monetary expenditure.

Educators at Miami University evaluated the inverted classroom as a vehicle to completely reshape the software engineering curriculum by inverting modules within each course when deemed beneficial (Gannod, Burge & Helmick, 2008). Their pilot study inverted a section of a special topics course on service-oriented architectures and web services and laid the groundwork for an inverted programming fundamentals course and data structures course. The instructors used the Apple iTunes platform to release video podcasts of each lecture for consumption by the students at any time. This accompanied an increase in the number of learning activities to fifteen, and a corresponding increase in instructor-mentored implementation of course content. Survey
results, as reported, showed high levels of student satisfaction with the technique, with suggested improvements in the delivery mechanism.

Since the inverted classroom allows the use of more in-class contact hours for practice problems and guided activities, Papadopoulos & Rico (2010) used the format to teach a Statics course. Statics courses are predominantly composed of applied mathematics and problem solving; these two skill sets usually require large amounts of practice to learn, understand, and internalize, making the Statics course an ideal candidate for the inverted classroom. Papadopoulos, the instructor for the course, used a semi-inverted format, with pre- and post-lecture learning modules for each day of class. This semi-inverted classroom allowed the instructor to remain comfortable by lecturing, but enabled the lecture to be shorter and more focused, thus leaving time for in-class problem solving. Reactions to this course format were positive from both the students and the instructor, with the students expressing the sense that they learned more than they would have with a traditional course format. The instructor noted a large time investment for setting up the learning modules for the course, which was a notable increase over the traditional classroom. However, the students performed better in the inverted classroom, which lead the instructor to prefer the format over the conventional method.

The different uses of the inverted classroom are widely shown through these experience reports, but leave the question of its effectiveness as a pedagogical technique without evaluation. Toto & Nguyen (2009) attempted to address the efficacy of the inverted classroom in their implementation for an industrial engineering course. By using a combination of course surveys and learning style inventories, they correlated student
satisfaction and study habits with learning styles. The results of this analysis show a multifaceted relationship between learning style dimensions with satisfaction with the format. This means that the inverted classroom, as a technique, is not a panacea for all types of students and must be implemented carefully to realize its benefits within any course structure.

The inverted classroom has many benefits, as discussed in this section. However, the technique relies heavily on the use of pre-recorded lectures, or podcasts, which are used outside the inverted classroom in many ways. The next section surveys the literature around distance learning, which focuses heavily on the efficacy of podcasts as educational tools.

**Distance Learning**

The inverted classroom creates a hybrid classroom experience, with a blend of features from both in-person and distance learning. As such, a small set of literature on distance learning is important to consider for this research. Distance learning originated with correspondence courses, where instructors created lessons and sent them to students via mail order (Moore & Kearsley, 2005). As technology increased, these correspondence courses became more widespread through first radio, then television, and eventually the Internet.

Currently, the most common type of distance learning course is the Internet course, with instructors reaching geographically distributed students through learning management systems. Through these collaborative computing systems, instructors distribute their lectures as *podcasts*, which are either audio-only or audiovisual
recordings, to the students who then watch them on their own time. The instructor then
gives the students assignments that the students complete, just as in a normal classroom
setting.

Many researchers have been studying this Internet-based educational paradigm in
recent years. Heilesen (2010) tested the efficacy of podcasts as educational tools in terms
of their effect on student academic performance. He found results to be inconclusive,
meaning that podcasts did not show significant gains or losses when compared to
traditional lecture methods. However, Heilesen also found that both students and
instructors find increased utility from podcasts when compared to traditional lectures, as
the podcasts serve as a tool not only for initial learning, as a lecture does, but also as a
guide for later studying. He conjectures that this may help improve the academic
environment, but that at the time the paper was written, there were insufficient
longitudinal studies to show whether this is truly the case.

Though recorded lectures are primarily used for distance learning, they can be
incorporated in the classroom in ways other than the inverted classroom. Kurtz, Fenwick,
& Elsworth (2007) experimented with recorded lectures as a supplement to their software
engineering curriculum. While the students in the test section did not show any
measurable learning gain on exam scores, they did outperform students in the traditional
section on the course projects and thus received on average higher grades than their
fellows. The authors suggest that this is because the students were forced to do more
work outside of the classroom than in a traditional format, thus helping them to
internalize concepts better. However, the prerecorded lectures were not popular with the
students, as they felt the increased burden was “extra” work. The implication is that student attitudes are central to the success of recorded lectures.

Lonn & Teasley (2009) tried to discover the implications for using podcasts as an educational tool and the perceptions of both students and instructors as to their efficacy. They found that the efficacy of the podcasts was highly dependent on the manner in which instructors used the podcasts. If the instructors merely used the podcasts as a supplementary tool, like distributing PowerPoint slides for review, then the podcasts displayed properties similar to those other review materials. However, if instructors used the podcasts to “capture fundamental topics for review while devoting face-to-face time for more discussion, student-led interaction, and other innovative activities,” (Lonn & Teasley, 2009, p. 92) then the podcasts affected a change in the educational paradigm and allowed instructors to create a variety of learning opportunities for the students.

Distance learning is not a phenomenon present solely in technical disciplines. Podcasts and video streaming have also been studied in the nursing discipline. A study conducted by McKinney & Page (2009) attempted to use podcasts as remedial tools to help nursing students learn pathophysiological processes, which are a common stumbling point for students. The students found the podcasts to be useful as a review tool, but indicated a preference for face-to-face contact with the instructor. This result suggests that podcasts cannot solely take the place of instructors without a loss in student comfort.

By using podcasts to re-teach mini-lectures, short lectures that cover small topics that typically confound students, Smith & Fidge (2008) hoped to mitigate the problems of high dropout and failure rates in an introductory computer science course. The students
showed an appreciation for the effort put into creating the mini-lectures and felt an increased level of confidence about the course. However, this confidence did not translate into quantitative results; there was no measurable improvement in student grades and the dropout rate remained statistically unchanged. The authors conclude that using podcasts specifically to improve student grades is unlikely to yield results, but that podcasts can improve student opinions of the course and perhaps raise the level of engagement in the course.

Walls et al. (2010) noted a large increase in the interest in using podcasts as an educational resource starting with a body of work in 2006. This interest was put forth by researchers and instructors, but did not show an echo in student interests. Thus, Walls et al. set out to measure student interest and gauge whether podcasts would be a welcome addition in the classroom. Their results show that in their sample, “only two-thirds of students owned and used digital media players,” (Walls et al., 2010, p. 375) and that similar studies showed only a slowly increasing trend in familiarity with podcasting technology. The authors indicate that assuming students will be familiar with podcasting technologies is dangerous and that students may not be as ready for the transition as innovative faculty members.

Overall, the literature indicates that using podcasts, which is an integral part of the inverted classroom, is valuable in the right circumstances. The overwhelming evidence shows that podcasts, when used as supplemental tools, are accepted by students and show positive results, but that relying on them entirely for instruction produces inferior results. In the context of the inverted classroom, these results are not troublesome because of the
symbiotic relationship between pre-recorded podcasts and alternative methods like active learning and class discussion. Thus, an evaluation of those alternative methods is necessary to determine the validity of the instructional method. Such an evaluation follows, through a survey of the case-based learning literature.

**Case-based Learning**

Story telling is a core part of the human experience, whereby knowledge is transmitted between storyteller and listener and the listener infers lessons from the story. This tradition hails back as far as Aesop, a slave in ancient Greece, who wrote a collection of fables (Aesop, Temple & Temple, 1998). A fable is a short, succinct story that is intended to impart a moral lesson. Starting from this point, where stories are constructed to teach small lessons, educators devised the concept of case studies.

A case study is a recounting of an experience that shows a particular educational concept in a real, or fictionalized, context that were heavily used in law, business and medicine before being adapted to science education. First described as an educational tool for science by Clyde Herreid in 1994, case studies were heralded as a primary method for increasing the quality of science education in the United States (Herreid, 1994). By taking the context of a situation and distilling it down to an analysis and mapping to a set of principles, Herreid took experiences and used them to teach higher order thinking to science students.

Through a long series of articles stemming from his research, Herreid defined what cases look like and should say (Herreid, 1997), what material makes a good case and how to identify such situations (Herreid, 1998), a process for writing cases (Herreid,
and defined the dilemma between using real experiences and carefully crafted fiction for the basis of educational case studies (Herreid, 2002). These papers form the basis of educational case studies in science and are a synthesis of practices in the older case-based educational fields of law, business and medicine.

Researchers in those older domains have created guidelines for creating cases for their disciplines. In medicine, Dolmans et al. (1997) defined seven principles for creating good cases. They state that cases must: draw on prior knowledge in the students; cause the students to elaborate on newly acquired knowledge to practice it; have relevant context for the knowledge at hand; allow students to integrate knowledge into a cohesive whole; provide the necessary tools for self-directed learning; pique student interest in the subject matter; and have clearly defined objectives in the mind of the instructor (i.e. learning objectives). These principles allow cases to be concise and useful to both the students and the faculty.

Raju & Sankar (1999) used case-based learning to teach an engineering management course. The authors developed a case study and then used it in a course and with professionals in the field to determine its utility among participants. The goal of using the case study was to bring real-world issues into the classroom and expose the previously insular student culture to these problems that don’t manifest in an academic environment. The results were generally favorable, with students finding the cases to be useful, clear, and challenging, thus motivating their work in the course. The students also appreciated the cross-disciplinary nature of the case, as it was jointed created by engineering and business faculty. The takeaway from this study is that cross-disciplinary
cases based on real-world projects can bring value by exposing students to content that they would otherwise avoid.

Marks, Freeman & Leitner (2001) converted a general-education computer science course to a case-based format in order to teach the concepts without delving into programming topics. The instructors’ goal was to reach non-major students and teach them the basic concepts underlying computer science “in a popular-science format.” The students rate the course very highly and report positively on their learning experiences and engagement in the course. They also score well and exit the course with high marks. This report supports the hypothesis that case-based learning can be very engaging for students, even when dealing with concepts outside their direct interests.

Students in technology disciplines generally have high opinions of case studies (Rosson, Carroll & Rodi, 2004). In order to provide concrete, hands-on projects, Rosson et al. used case studies. The instructors used case studies as the basis for the hands-on projects in a scenario-based usability course, and developed a prototype of an online tool to distribute the case studies to the students. They also formed activities from several case studies for use in the classroom. Students rated the activities and projects well, with 61% stating a “very good” or “excellent” opinion of the activities. These results show that case-based learning is enjoyable for the students when executed well.

Although most case-based learning curricula use multiple cases to illustrate various points throughout the course, some instructors have attempted to use one overarching case to teach the majority of a course (Hilburn et al., 2006). This resulted in a large case study that sprawled through many topics. However, the researchers noted
that it, “addressed issues ranging over a disparate set of problem domains, software engineering practices, and scenario elements; this resulted in students learning about software engineering practices in bits and pieces [...] and hence [there was] no accumulation of scenario experience that allowed progress toward more substantial and complex problems” (Hilburn et al., 2006, p. M1F-5). This result supports the claim that case studies should be small, focused views into real-world problems that address only a few main concepts at once.

When teaching a case-based course, an instructor typically must have a library of case studies from which to pull content. Jiang, Ganoe & Carroll (2010) propose four requirements for creating a balanced case study library. The first requirement, *authentic learning*, addresses the issue first brought forth by Herreid (Herreid, 2002) about the veracity of cases. Authentic cases better prepare students to do instead of just to know. The second requirement, *social interaction*, speaks to the need for students to collaborate with their peers to learn and reinforce their understanding. The case study library must provide motivation for the students to work together. The third requirement, *resource accumulation and updates*, states that, in order to avoid a tragedy of the commons, a case study library needs to reach critical mass of use. This implies that case study libraries should be open for use by more than one instructor, in order to help the resource grow. Lastly, a case study library should develop towards *communities of practice*. Because case studies are domain-specific and require students to use domain-specific knowledge, case study libraries need to focus on one particular community of practice without
branching into another. These requirements for case study libraries will help guide the creation of our own case study library.

These studies support the idea that case-based learning is an effective method for conveying complex, domain-specific topics. They also put forth guidelines to creating cases and putting together case study libraries. While case studies are effective, they are not maximally engaging, as there is still a level of abstraction between the students and the content. To overcome this minor limitation, our novel curriculum uses active learning games and activities to engage students; a short survey of the active learning literature follows.

**Active Learning**

The concept of active learning first entered the educational vernacular in the 1980s, when a series of United States government reports urged higher education faculty to actively engage students in their learning (Study Group, 1984). These reports were followed by further arguments by university faculty to their peers (Cross, 1987). By appealing to more learning styles – in particular tactile learning – these influential reports argued that the country could turn around its flagging science expertise and retain the title of “world leader.”

Soon, educators answered. Bonwell & Eison (1991) created a workshop intended to teach educators how to bring active learning into the classroom. This workshop continued and its creators continued to update it for nearly ten years. The result of this workshop and other researchers was a sizeable increase in active learning in primary,
secondary and higher education classrooms (Bonwell, 2000). Today, active learning is an integral part of science education throughout the country.

Active learning has been shown to be effective in engaging students in a Computer Science 2 (CS2) course (Briggs, 2005). The author showed significant increases in student grades in the course by redesigning it to consist primarily of heavily guided, in-class, team programming assignments that are targeted at a single concept in the curriculum. By relying on the knowledge students acquired in their Computer Science 1 (CS1) course, the instructor empowered students to elaborate on their knowledge and expand their skills in a supportive, guiding environment. The results of this study show that student grades improved significantly, with a unilateral shift in the distribution of grades. Many fewer students failed the course and students expressed satisfaction with the challenge inherent in the format. These results show that active learning can significantly improve student acceptance and engagement in a course, even when the course material is difficult.

When creating active learning activities, it is helpful to modularize the content of the course to enable the instructor to switch to an active learning exercise for particular portions of the class. Quade (2006) modularized a software engineering course into nine modules that were focused on cognitive and social course goals. The author notes that creating active learning modules is time-consuming up front, but both traditional students and project sponsors from industry find the result to be accessible and engaging.

Sowell et al. (2010) experimented with transforming a traditional software development course into an active learning format by combining prerecorded lectures
with in-class games and activities. This format closely mirrors the inverted classroom format, as discussed previously. The in-class activities were primarily studio programming assignments where students would work in teams and program, debug, and document laboratory assignments. Students showed improvements in course grades and engagement and rated their satisfaction with the active learning format higher than that in the traditional classroom. The instructor noted some improvements to the format that could be done, including rotation seating assignments within the teams in each lab to ensure that students get experience doing each activity. This report shows the benefits of the active learning paradigm through collaboration with peers and heavy interaction with the instructor.

This short review of the active learning literature shows the effectiveness of the approach. Active learning, case-based learning and the inverted classroom form the three pillars of our novel software engineering curriculum. Between the three methods, the curriculum should appeal to a variety of students and the instructors will be able to cover a large amount of material while leaving time to bring their expertise into their teaching through discussion and anecdotes. Next, Chapter 3 contains a thorough description of the software engineering curriculum and describes how each of these educational paradigms fits into the curriculum.
Chapter 3: Enterprise Software Engineering

In order to teach the ever expanding amount of software engineering content expected by industry, the curriculum needed to be reimagined to a format conducive to compressed information. However, compression of content can have the side-effect of degrading the fidelity of the subject matter, thus impeding learning. The need to effectively teach, instead of merely communicate, the material necessitated some innovative teaching methods. The method that we chose, for reasons covered in Chapter 2, was the inverted classroom, a technique that moved traditional lecture out of the classroom and into the students’ study time. This allowed for the material to be covered by the students in their own time, while opening the class for more interactive learning.

This chapter discusses the major components of the enterprise software engineering curriculum, the reasoning behind their implementation, and structure of the course. The chapter describes a set of five components: first, the prerecorded lectures and accompanying quizzes; second, the emphasis on in-class discussion; third, the active learning games and activities used within the course; fourth, the course content and modules; fifth, course management issues.

Prerecorded Lectures and Quizzes

The main feature of the inverted classroom is the lectures as given out of the classroom. The literature shows several ways in which the lectures can be delivered, with the most popular being video podcasts. A podcast refers to the recording and episodic
release of content to an audience that subscribes to the stream. Viewers download the podcast episodes once and can watch them as many times as they wish. This format lent itself to both instruction and review, as students can pause, rewind, and rewatch sections that they did not understand on their first viewing. Because of these benefits, we selected this format for the curriculum.

Setting up the pre-recorded lectures for the first time was a time-consuming process. Since the software engineering course was not a brand new course, there were pre-prepared lectures that were used for previous sections. These lectures all contained material that was still going to be taught with the course redesign, so they acted as a seed for the new instructional materials. The process to create the pre-recorded lectures consisted of five steps: first, we sorted through the old lecture presentations and recombined the content into more coherent sections; second, we recreated the lecture slides in a format that was more conducive to pre-recorded lectures; third, we recorded audio for each slide; fourth, we created a video from the lecture slides and overlaid the audio over the proper slides in the video; lastly, we uploaded the videos to a popular video sharing website and distributed links to them to the students.

The first step in transforming the previous lectures into a form suitable for pre-recording was to tighten up the content and reorganize it to flow more logically within each lecture. The goal here was to create videos that could be consumed at any time without strict dependencies between them. That is, the second video did not rely on the first video to describe background knowledge for the section; it was self-contained. The reasoning behind this decision was to improve the utility of the videos as a study aid for
By eliminating strict dependencies between videos, we felt that students
could easily rewatch particular videos that they struggled with the first time through. If
each video required the student to rewatch other videos just to recapture the context of
the lecture, we felt the students would be much less likely to use the videos as study aids.

After restructuring the lectures into self-contained units, we remade the lecture
slides in a more visually appealing, video-ready format. This conversion involved paring
down the textual content to a minimum and bolstering the result with diagrams, images,
and animations where useful. Figure 1 shows an example of this process. The original
slide, shown on the left, outlined the process of business-IT alignment as talking points,
with a lot of information on one slide. The new slide, shown on the right, illustrated the
same process with a diagram that was then discussed in the audio recording. The
reasoning behind converting slides to this format was that the video should provide a
more engaging format to look at while the lecturer discusses the topic in the background.
Students who are auditory learners could then listen to the lecture, while visual learners could study the diagram and clarify it via the audio.

The third step to create the pre-recorded lectures was to record the voiceover audio for the slides. In order to create a coherent voiceover, we wrote a script for the speaker to read while recording. This process involved several iterations of outlining. The first outline enumerated the main points of the lecture to give an overall view of the lecture flow. Next, we refined this outline to cover each point on the slides. From this refined outline, we wrote out the full sentences that needed to be read by the speaker. The speaker then rehearsed the speeches and recorded the voiceover using an open source recording suite. We separated the audio into short files focused around individual slides. The reason for chunking the recordings was to allow students to access just a small part of the lecture easily, through opening one file, which made it easier for students to study just the content they needed to. There was a secondary benefit to recording shorter sections: logistically, it was difficult to keep the area surrounding the speaker quiet for long periods of time. The background noise made it easier to record short sections, rather than record long sections that were later split.

The fourth step in the production of the prerecorded lectures was to record the video and overlay the audio. When creating video podcasts, there are three main paradigms for what to show on video: an animation or series of images with an audio overlay; the animation or images with the narrator overlaid on a smaller picture-in-

---

1 An unplanned benefit came from writing full sentences for the speaker to read for each slide. We attached the speaking notes to the lecture slides, which we then distributed to the students. The notes were used as a supplement for the students to study, if they preferred. This allowed the students to print out and highlight important phrases that were not included in the slide visual.
Figure 2: The three paradigms of podcast lectures.

(a) Showing only the slides for visuals, with voiceover audio. (b) Showing a picture-in-picture view of the lecturer as he or she is speaking. (c) Cutting between views of the slides and the lecturer, as appropriate.

picture; and a series of cuts between a narrator and animation or images. These paradigms are shown in Figure 2. For simplicity and ease of reference, we chose to use still images with a voiceover. The images that made up the videos were the slides and were recorded using the timing functionality within the slide presentation software. This allowed the video to be shot in one continuous take, with smooth fade-through-black transitions between the slides to create a consistent, clean look-and-feel. We then overlaid the audio on a single track in the recording software and rendered the resulting composite in a web-optimized format.

The last step in video production was to arrange for easy distribution of the videos. At first, we hosted the prerecorded lectures for download through the department's computing resources. These packages contained the videos, the slide decks, the transcripts of the audio, and the audio files; students found it cumbersome to handle all of these files, so we searched for a replacement technique. When doing this search, we considered that the students in the course were primarily members of the millennial generation, also known as Generation Y. Students from this generation show a high
reliance and use of technology, with social networks being a primary use of technology (Cabral, 2011). Millennials also time- and place-shift much of their video watching through use of their MP3 players (Nielsen, 2009). By considering these two attributes of our students, we decided to use the popular, social, video distribution website YouTube for hosting. By using YouTube, we enabled students to watch the videos on any YouTube-capable device. This helped us to meet our goal of allowing the students to watch the lectures in their own time, in the manner that they wished.

The videos were then distributed to the students a week before the date assigned for discussion. The students were asked to watch the video on their own time before the discussion session, take notes as they saw fit, and ask any questions on the course newsgroup. This course design was intended to allow the students ample time to learn the material from the pre-recorded lectures and clear up any confusion they had about the lectures.

On the day of each discussion session, the students were given a short ten-minute quiz on the topics that the video covered. The intent of this quiz was to ensure the students had watched each lecture. The quizzes were structured as open-ended essay questions with many possible solutions. By asking questions that did not have a cut-and-dry, rote answer, we felt that we provided incentive for students to really try to learn and understand the material in a way that memorization doesn't create.

The quizzes were graded on a three-point scale, with students receiving credit based on the completeness of the provided answer. If the student demonstrated thorough knowledge of the subject material, with examples of how the concepts applied to the
question, he received three points. To receive two points, the student had to demonstrate that he had watched the video by attempting to provide an example of the content, but his understanding was either incomplete or slightly askew. We awarded one point to students who failed to internalize the concepts, but showed some thought behind their answers.

This three-point scale was intended to simplify the grading process and give easy-to-understand feedback to the students on their performance. We made the quizzes worth ten percent of the course grade, which is similar to the attendance requirement of many courses in the department. By making the quizzes worth a cumulative ten percent, we felt that students would be motivated to do well on the quizzes by watching and studying the lectures, but would only be penalized slightly if the instructional method did not mesh well with their learning style.

The quizzes were put into place to ensure that students watched the prerecorded lectures, because the in-class sessions were focused on activities and discussion that required the students to be familiar with the material. The discussion component, as discussed next, formed the basis for students to practice the skills, terminology, and methods that they learned through the prerecorded lectures.

**In-Class Discussion**

In the traditional classroom, with a lecturer standing in front of a room of students and speaking with their authority, students are inherently passive participants in the learning routine. In recent years, engineering education has started to move toward a constructivist learning theory, where students are active participants in their learning
In-class discussion forms a key component to meet two ideals of a constructivist pedagogy: construction and collaboration.

Construction refers to the heart of constructivism, that is, new knowledge is constructed on the foundation of learner's prior knowledge, whether it comes from classroom teaching or through the experience of living. Collaboration is a tenet of constructivism because learning is an inherently social experience, as “thinking is situated in a particular context of intentions, social partners, and tools” (Greeno et al., 1996, p. 19). Because of this couching within context, in some cases elaboration – or describing an educational topic – is not the key for learning, but rather collaborative elaboration, consisting of immediate feedback between students, is the medium by which students achieve more complete learning outcomes (Van Meter & Stevens, 2000, p. 123). Peer discussion forms the heart of these principles.

Following each prerecorded lecture and quiz, the instructors use the remaining class time as a discussion session. The objective in the discussion session is to get the students to talk about the material from the lecture, argue their answers to the quiz, and check their understanding. If the students do not immediately start with their own sets of questions, the instructors prompt discussion by bringing up the lecture slides and focusing discussion around what they feel are the key elements of each module. Each instructor views different portions of each module as more important based on his experience in industry.

The instructors also insert anecdotes from their own industry experience that show the importance of the course module. Often it is difficult for students to
immediately see the impact how the course material will affect their careers, so the stories help show the importance of understanding the software engineering process. Some instructors also bring in their employees during the course module most focused around their careers. For instance, when the testing module is under review, one instructor brings in his testing lead to discuss what he does, why he does it, and how testing is important to the overall product success.

These anecdotes and visiting workers bring a third component of successful constructivist pedagogies: authentic tasks. By bringing in guests who do the kind of work the lecture talks about, the instructors motivate students to apply the course concepts to the term projects and help them to bring the abstract concepts taught in the course to a real, useful level of understanding. This ability to move from abstract knowledge to implementable, concrete actions is an important goal of this curriculum, as it will help students make immediate impact in their careers.

While it is certainly true that devoting entire class periods to discussion increases the opportunity for collaborative elaboration, it is not a guarantee that the discussion is beneficial to every student in the course. The emphasis on discussion requires the instructor to know who his students are, what their backgrounds and relevant experience in the course modules are, and the span of that background experience. If, for instance, there is one non-traditional student or graduate student in the course who monopolizes the discussion by asking very technical questions, while the rest of the class consists of mostly inexperienced undergraduates, then the undergraduates will likely not find the discussion beneficial, because they do not understand all of the context required by the
topic. The instructor must, therefore, be cognizant of the background necessary to understand discussion material and steer the conversation in such a way that the maximal number of students are engaged and understand the debate.

Because of the balance between covering topics in too much detail to understand for the average student versus too little challenge for the average student to find interesting, there is a notion of the quality of discussion that must be considered. *Discussion quality* is an important metric for the success of this curriculum, as low-quality discussion will not elicit collaborative elaboration, which is the theoretical underpinning of holding in-class discussion. High-quality discussion, on the other hand, is important for the students to solidify their understanding and maximize their learning outcomes for the course. We must, therefore, evaluate the quality of these discussions to test whether the emphasis on student dialogue has the intended effect.

While discussion makes up a significant portion of the in-class hours for the course, there is another component that is important for keeping students engaged and excited about the course. That piece is the active learning games and activities that take place throughout the course to demonstrate particular concepts in an in-depth, hands-on fashion.

*Active Learning Games*

In order to teach very applied concepts to the students in a way that is fast enough, yet still instills the concepts to a degree in which they can recall specifics even months later, we resorted to games and activities. By engaging students through a fun, seemingly non-academic activity, we remove some stressors of the classroom and bring
team-based incentives to motivate them to recall the lecture information and use it appropriately.

Agile development is currently a very popular topic in the software industry. Since the Agile Manifesto was published in 2001 (Beck et al., 2001), software engineers started developing new processes to get work done in a sense-and-respond fashion, rather than relying totally on upfront planning. Agile principles are worthwhile to teach, but without actually practicing them, students find them difficult to understand. This situation is the perfect candidate for an active learning game.

The first active learning we developed is a team-based, agile development workshop that uses Lego bricks as the medium of instruction. This workshop is an hour long simulation of product development in an agile fashion. Before the day of the workshop, the students watch an inverted lecture that teaches them the principles of agile development in comparison to the more structured processes like the Rational Unified Process or the waterfall development model. This pre-lecture is intended to give the students some idea of the principles they should use in the workshop.

When the students arrive at the classroom for the workshop, they are split into teams of approximately six people, with a graduate student acting as the customer for their team. Another graduate student or the course instructor acts as the facilitator for the workshop, calling out instructions, keeping time for the iterations, and giving short, three-minute mini-lectures on a topic in between the iterations. All of the teams receive the same user stories for each iteration, and they all attempt to build the same construct: a
Lego animal, as shown in Figure 3, that matches the design requirements set forth by the user stories.

The development process within the workshop is divided into three twenty minute iterations. The teams are given five minutes to plan before each iteration, discussing the story cards, assigning a difficulty to each, and committing to a set of story cards with the customer; the students are prohibited from building anything during this stage, as it is purely planning-oriented. They then have ten minutes to construct the animal per the story cards they committed to, followed by a five minute acceptance test and retrospective where the whole class has a discussion about what went right during the iteration and what went wrong. The discussion for each iteration focuses on a different agile principle, to maximize learning gains about a core set of principles.

One core principle of agile development is that the customer is a team member; they work with the development team, have as much or more say than anyone else in the
direction the product takes, and is always involved with the product, whether it is the first
day of design work or the day before launch. This principle is stressed in the inverted
lecture on agile development, so the students should realize that it is central to the
workshop.

However, we designed the workshop to deliberately hide the fact that the
customer is a team member by making the customers appear disinterested in the product
and what the team is doing. The customers are all instructed to act distant toward their
teams, wandering off to talk with other customers, playing on their cell phones, and only
engaging with their teams when asked a direct question. Because of this apparent attitude,
the student team members almost unilaterally forget about their customer, only tracking
their customer down to ask a question if they truly do not understand the meaning of a
story card.

We also instruct the customers to reject in the acceptance test at least one story
card that the students did not ask them about. They can make up a reason, reject a card
arbitrarily, or honestly reject a card that they felt the students didn’t meet, and they
should make clear to the students that they could have met the acceptance test had they
only asked about the story card. Following this failed acceptance test, the workshop
facilitator asks all of the teams whether they had a story card fail to be accepted. Because
the customer was instructed to reject at least one, the students unanimously answer in the
affirmative and the facilitator gives a mini-lecture about the customer being part of the
team and involving them every step of the way. The students subsequently involve their
customers more in each iteration and have fewer problems with the acceptance test.
Velocity is an important concept within agile development, as it is the only way agile teams have of gauging the amount of work they get done in each iteration. By measuring the effort – gauged by the difficulty they gave to each story card – they put into the story cards to get them accepted, the students can then better plan their next iteration by only planning to do enough story cards to cover the level of difficulty that they completed in the first iteration.

The customer, however, is not interested in how hard the team works, but in how much value they provide the business in each iteration. To demonstrate this balance in the workshop, we assigned each story card a business value, which is printed on the card. After the first iteration, the workshop facilitator tells the students what the number means and explains to them the importance of business value when making project management decisions. Thus, the students must evaluate their planning decisions and maximize the amount of business value that they can accomplish for the velocity from the last iteration. The decisions that they make after understanding that business value is important are different than those they make, with the students deciding to do fewer, higher business value, more difficult cards, rather than complete all the easy tasks.

To further demonstrate the value of working with their customer and measuring velocity, we make the students graph both their velocity and their delivered business value on the chalkboard so they can compare their performance with the other teams, as well as with their own performance in previous iterations. We generally see a large increase in business value in the second iteration, due to the focus on delivering the most business value in the iteration. We also see a slight increase in velocity from the first
iteration to the second, due to the students getting fewer (usually zero) story cards rejected.

In the second iteration, the customers are instructed to work closely with their students and decide which member of the group is the biggest asset to the team. At the end of the iteration, this team member is swapped with the strongest member of another team. We intend this process to simulate “brain drain” and the subsequent issues that are caused when a team member is an “information silo.” Because the strongest team member is often the one working on the most difficult task or the one leading smaller sub-teams, they are regularly the only person that knows everything that is happening on the team. The subsequent loss of this team member hampers the productivity of the group and slows their progress during the third iteration.

The principle that we want to instill through this brain drain is a core foundation of eXtreme Programming (XP): pair programming. By having two developers work together, developers practicing XP can mitigate the problems caused by brain drain because each information silo is effectively two people instead of just one. This allows the team to suffer minimal loss of knowledge if a team member leaves the company or is no longer able to work. An interesting side effect of the students being removed from their original teams is that they move from being the most productive team member to generally the least productive on the new team, because they haven’t bought into the animal the new team is creating and want to work on and finish their old team’s animal. This has a secondary benefit of showing that hiring the best talent does not necessarily show the biggest performance gain for a team.
In the third iteration, the teams should have their process refined to the point that they can easily finish all of the remaining tasks, despite losing their strongest member. We find that the students all include the customer, are involved with the planning of the last iteration, and understand and use the concepts of velocity and business value when planning out the last iteration. After the final iteration, we take pictures of the groups with their Lego animals and issue surveys to check that the students learned the concepts that we wanted them to learn from the workshop. The results of these surveys will be discussed in Chapter 4.

The students enjoy the agile workshop and we feel it has been a success. As such, we want to explore different activities that provide the same tactile learning experience in a less time-intensive fashion. Yet, not every topic is appropriate for this experience; it has to be one that naturally lends itself to an activity and is complex enough to warrant the extra time to ensure students really grasp the concept and its importance to software engineering. The topic that lends itself immediately to this format is the subject of our next activity, the legacy system call center game.

One topic that is extremely important for new software engineers to understand, but is difficult to impress without hands-on experience is legacy systems and the problems inherent to interoperating them. Inside large enterprises, the information technology department has many legacy systems that are left over from years or even decades ago, yet are still in use because it is too costly to update or replace them. These legacy systems provide core business functions but are often written in outdated languages and frameworks that younger employees never learn. However, the data that
these systems output are often needed in newer systems or new systems need to use the old system somewhere in their data processing pipeline. As such, the new systems need to learn to “speak the same language” as the older systems, adapting to the structure and format of the old systems’ inputs and outputs.

The translation from old data formats to newer designs is a non-trivial process, and when the presence of dozens of these legacy systems is made apparent, the complexity of the problem quickly becomes a *wicked problem* (Batie, 2008). Wicked problems are those that are ill-defined or intractable using normal scientific approaches due to an overwhelming level of complexity. If we cannot approach a problem conventionally, then adequately explaining the problem is unlikely. In order to convey this issue, active learning can play a key part.

The activity we devised to teach about legacy interoperability simulates the call center of an international corporation. It consists of an interactive demonstration about what happens when people with different skill sets, who speak different languages, can be guided to work together to some degree, through working together. The activity is designed to run within the time limit of a normal lecture and involves many students in the course in different roles within the activity.

The instructor chooses one student to act as the call dispatcher. This student sits at the front of the room, facing the other students and receives “calls” that he then distributes to groups of students in the class. The instructor divides the other students into smaller groups who are “specialists” in a particular area, e.g. Java systems or web applications. The groups of specialists work out simple problems (usually tasks like
addition or multiplication; the task type is irrelevant so long as the responsibilities are divided among the groups) and then communicate the answers either back to the dispatcher or to another group.

The groups must face and work around challenges that make interacting with the other groups more difficult. For example, one group might be based in India and the group can only communicate to other groups through an appropriate language, e.g. Tamil or Hindi, that someone in their group knows well. If they are told to work with another group on a problem, then the groups must collaborate with the dispatcher and devise a way to communicate through the language barrier. This simulates the problems inherent in large legacy systems, where discrete components format their data in different manners and are unable to be extended to use other formats.

The flow of the activity revolves around tasks that the instructor issues to the dispatcher. The dispatcher knows what each group is capable of – for example, the “Java group” could be in charge of addition and the “web applications” group responsible for multiplication – and must figure out how to use the groups to complete the task given by the instructor. By sending jobs out and giving orders as to how the result should be routed afterward, the dispatcher can work with a person from each group (known as the communicator), one at a time to discuss what needs to happen next. The communicators from each group cannot talk with each other unless specified by the dispatcher and must always meet the restrictions placed upon them when talking to another communicator. When talking with the dispatcher, the communicators are free to use whatever language and means they deem necessary.
The result of this complex task is that it becomes an enormous amount of work to accomplish a simple task; for instance, calculating the value of the right-hand side of the Pythagorean theorem \((a^2 + b^2 = c^2)\) can take most of a class period with the right constraints placed on the student groups. Coupled with mini-lectures and help from the instructor, the activity takes a whole class period to impress upon the students the importance of the complexity of enterprise legacy systems.

The instructor, at the end of the class, reveals what the students were actually doing. The students often leave this call center activity bewildered by how difficult the simple task was given the constraints and understand how the activity relates to their future careers as software engineers. This reaction is the desirable conclusion to the activity and indicates that it is accomplishing the job of teaching about legacy enterprise systems.

However, there are some lessons that cannot be taught via game or activity, whether due to a level of complexity that is untenable for a limited-time game, or conceptual incompatibility with the active medium. To address these topics, we rely on case studies. In order to use case studies effectively, we must continually update our case library with new material. This process is discussed next.

**Creating Cases**

Case studies form the basis of many educational programs from law, medicine, and business. They have only recently entered technical disciplines, like software engineering. Because of this, there is no large case library from which to choose cases to
teach specific topics. The lack of a case library requires educators to create case studies specifically for their courses.

However, case study creation methods are not concrete, instead existing in a realm similar to writing guidelines, at a very abstract level (Dolmans, 1997; Herreid 1998; Herreid, 2000; Herreid, 2002; Jiang, 2009). Because of the concrete nature of cases, where the context in which the case operates is important and often contains the bulk of the instructional value (Herreid, 1994), cases can appear to be merely stories when not formatted in a useful manner.

In order to extract the pertinent information from a situation, format it in a way that is conducive to instruction, and bring a level of consistency to the created cases, an instructor must have a concise, repeatable method to use when creating cases and a body of material from which to draw the case foundations. The structure of our research group is easily able to provide case foundations, as there are upwards of a dozen different projects being worked on by members of the research group at any given time. As such, we needed a method to harness this information and construct cases.

As a precursor to the method creation, we compiled a list of the course modules and enumerated each key topic within the modules. This list was then used to check for coverage of each topic by cases. In order to ensure that we account for most topics via case study, we used a visual aid in the form of a table. This tool enables an instructor to rapidly filter through a list of cases to determine an optimal case set for the course.

As shown in abbreviated form in Table 1, a table breaks down course modules into themes, which are then subdivided into individual topics, theories, methods, and
techniques. Depending on the number of cases chosen for the course in a given semester, this chart can be customized to show the topic coverage of the given set of cases. If the case set does not cover enough of the course topics, the instructor can modify his choices to better reflect the course topics.

By using a quick, visual check to ensure the case set an instructor chose will work for the class, we encourage instructors to be thoughtful about the composition of their case sets. However, this does not solve the problem of needing a case library to choose from when creating a case set. Creating a case library is a time-intensive process which is

<table>
<thead>
<tr>
<th>Module</th>
<th>Business Context</th>
</tr>
</thead>
<tbody>
<tr>
<td>Theme</td>
<td>Environment</td>
</tr>
<tr>
<td><strong>Key Topics, Theories, Methods, and Techniques</strong></td>
<td>Porter's 5 Forces</td>
</tr>
<tr>
<td></td>
<td>Sense/Respond</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 1: An abbreviated list of course topics with mappings from two cases

Three course modules are shown in abbreviated fashion, with colors mapping two cases onto the module contents. Red cells correspond to the ECO case, blue cells to the SID case, and purple cells are covered by both. White cells are not covered by either of the cases.

<table>
<thead>
<tr>
<th>Module</th>
<th>Software Engineering Process</th>
</tr>
</thead>
<tbody>
<tr>
<td>Theme</td>
<td>Structured Process</td>
</tr>
<tr>
<td><strong>Key Topics, Theories, Methods, and Techniques</strong></td>
<td>Constraints</td>
</tr>
<tr>
<td></td>
<td>Requirement Specifications</td>
</tr>
<tr>
<td></td>
<td>Roles</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Module</th>
<th>Software Engineering Practices</th>
</tr>
</thead>
<tbody>
<tr>
<td>Theme</td>
<td>Requirements</td>
</tr>
<tr>
<td><strong>Key Topics, Theories, Methods, and Techniques</strong></td>
<td>Elicitation</td>
</tr>
<tr>
<td></td>
<td>Functional &amp; Non-Functional</td>
</tr>
<tr>
<td></td>
<td>Stakeholder Identification</td>
</tr>
<tr>
<td></td>
<td>Use Cases</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Legend</th>
<th>ECO Case</th>
<th>SID Case</th>
<th>Both Cases</th>
</tr>
</thead>
</table>
aided by knowing what topics need to be covered. A secondary benefit of the visual organization table is that it acts as a thinking and planning aid. Enumerating the themes and topics of the course can help an instructor elicit knowledge about cases that specifically cover each topic. This can help motivate the first step in creating cases: gathering the material.

In order to write a good case study, one first requires solid, thoughtful base material to build from. Generally, the base material should be from a project that the case writer experienced, or the case writer should work closely with a team member from the project who was intimately involved with each phase of the project. Without this deep background knowledge, it is difficult to distill the concrete lessons that form the heart of a case study.

We create our case studies from the body of work done by the students in our research group. Because of the size of the group and breadth of projects, we are able to cherry-pick cases that focus on almost the entirety of the course topics. By leveraging these student-led projects, we gain a twofold educational benefit. First, the research students who write or co-write the cases are forced to be more introspective about their work, which helps them learn better project management skills and see software engineering as an interdisciplinary venture (Bolinger, 2010). Second, the students in the software engineering course see the complications that arise from real-world projects, even when those projects are only slightly removed from their own situation as classroom students.
The process by which we write the case studies consists of eight discrete steps that can be separated into three groups by the type of content they discuss. These steps and groups can be seen in Figure 4 on page 45.

The first group, “Creating the Context,” is made up of three steps and is intended to familiarize the students with the case project and the environment in which it was envisioned. This helps the students to understand the case better, as they have more information as to the constraints placed on the project design. The theory behind why contextualization aids understanding is underpinned by the need for authentic tasks and related cases within constructivism (Hadjerrouit, 2005).

The first step in contextualization is to introduce the project and its sponsor or business. This is followed by the author concisely stating the problem that the project solves. By understanding the project, the project sponsor, and the problem statement, the students can immediately start considering how they would approach the project and what the main problem points in the process will be. After discussing the problem statement, the case characterizes the context and environment in which it exists. Without the context and environment, the project is effectively unconstrained. The constraints placed on the case are the real source of the learning experience; since all real-world problems are constrained in some manner, the case author must show how he was able to overcome the constraints and accomplish his goal.
The second group of tasks, “Engineering the Software Process,” consists of only two steps. This section focuses on the people involved in the project and how they work together to create the software engineering process. The interpersonal and interdisciplinary work inherent in software engineering is often lost on students, resulting in an overwhelming adjustment when entering the workplace. By focusing our cases

<table>
<thead>
<tr>
<th>Creating the Context</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1.</strong> Introduce the project and its sponsor or business.</td>
</tr>
<tr>
<td><strong>Step 2.</strong> State the problem.</td>
</tr>
<tr>
<td><strong>Step 3.</strong> Characterize the context and environment of the project.</td>
</tr>
<tr>
<td>a. Choose one or two techniques from the Business Context module in Table 1 for the characterization.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Engineering the Software Process</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 4.</strong> Introduce the people involved in the project.</td>
</tr>
<tr>
<td><strong>Step 5.</strong> Characterize the project from an organizational standpoint and as a software engineering process.</td>
</tr>
<tr>
<td>a. Choose two or three patterns from the Organizational Pattern Reference (Coplien et al., 2005) for the organizational description.</td>
</tr>
<tr>
<td>b. For each organizational pattern, show how the pattern influenced the process by choosing one technique from the Software Engineering Process module in Table 1.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Analysis and Design by Patterns</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 6.</strong> Restate the problem from Step 2.</td>
</tr>
<tr>
<td><strong>Step 7.</strong> Characterize the analysis process from the project.</td>
</tr>
<tr>
<td>a. Choose two or three techniques from the Software Engineering Process module in Table 1.</td>
</tr>
<tr>
<td>b. For each analysis technique, show how it was relevant to each pattern from Step 5a.</td>
</tr>
<tr>
<td>c. For each analysis technique, show the result of its process and how it affected the system’s high-level design.</td>
</tr>
<tr>
<td><strong>Step 8.</strong> Characterize the low-level design within the context of the project analysis.</td>
</tr>
<tr>
<td>a. Choose one or two techniques from the Software Engineering Practices module in Table 1.</td>
</tr>
<tr>
<td>b. For each low-level design technique, show how a specific implementation can be derived by referring to a design pattern from the Software Design Pattern Reference (Gamma et al., 1995).</td>
</tr>
<tr>
<td>c. For each implementation strategy from Step 8b, explain how it satisfies the high-level design derived during Step 7c.</td>
</tr>
</tbody>
</table>

Figure 4: An outline of the case study creation method.
around the human aspects of the discipline, we help students realize that they will not be working alone and need to develop their interpersonal skills.

When writing about the effect people had on the project, the author first introduces the people involved in the case. Without knowing who the people involved are – whether the practitioners were from technical, business, or other disciplines – and how they interacted, the readers of the case cannot truly understand why the work proceeds as it does in the case. Next, the case author characterizes the organizational behaviors in the case through the use of two or three patterns selected from an organizational patterns resource (Coplien et al., 2005). The emphasis on established patterns helps the case author ensure that the phenomena he observed during the project are reproducible and important in cases other than this one particular project. The reason for discussing two or three patterns is twofold: every case we have observed can pinpoint at least two or three organizational patterns that manifested during the project; and by focusing on a small set of patterns, we can go in depth into each without creating long, unwieldy cases. By limiting the amount of content in each case, we make them easier to read and comprehend, which increases student satisfaction with the cases.

The third group, “Analysis and Design by Patterns,” has three steps. This group shows some of the analysis and design in the project and how the people, organizational patterns, and context all shape the outcome of the project. Project outcomes are dependent on the context and the people involved, so showing this explicit connection helps to reinforce this concept for the students. Without specific case studies, students do not realize that, regardless of how competent one individual is, the software engineering
process can fail if the context is not conducive to the project and project team doesn’t
work together. The discussion of the analysis and design within these domains helps
illustrate this point.

This section begins with the author restating the problem statement from the first
group, with additional understanding overlaid on the original problem. As projects
progress, the team members come to understand the problem better, see where the
problem areas are, and adapt their work to address issues that arise during the project.
The case author shows some of this understanding to demonstrate this principle.
Following this revised problem statement, the case author characterizes the analysis
process he went through during the project through the selection of two or three software
ingineering process techniques. He then discusses how these techniques were affected by
the people and organizational patterns inherent in the project and how these changes
influenced the high-level design resulting from the analysis process. After this, the case
author selects one or two design patterns from a design pattern reference (Gamma et al.,
1995) to illustrate some of the project’s low-level design. This low-level design is the
result of the contextual constraints placed on the design process and the case author uses
these constraints to show how the high-level design was instantiated. The case concludes
with this example of how design is a process of iterative refinement, without clear-cut,
obvious implementations of design patterns.

From these three groups of steps, the case author forms a complete case in a
standardized, easy-to-follow format. Because all of the cases within the course are
constructed through this process, the students learn to read the cases throughout the
course and distill the knowledge contained within them. The case studies provide the final piece of the constructivist curriculum by allowing students to see how work is done in real contexts and guiding them through the use of the course principles in a standardized manner.

With the pedagogical methods outlined above, it is useful to understand the content contained within the course and how the modules were constructed and arranged. As such, we discuss the course content and learning modules next.

**Course Topic Areas**

We divide the topics we cover in the curriculum into learning modules that are each covered by an inverted lecture. Compared to a traditional software engineering course, the learning modules are geared toward enterprise systems, which naturally leads to covering more business material. The course begins with two units that are entirely business-oriented, in order to ground the rest of the course in a mindset that considers how the products the students are designing for their term projects will actually deliver value to the business they are “working for.” Following this introduction through the business material, we transition to a curriculum that closely follows the prototypical software engineering course (Lethbridge et al., 2006).

The first learning module, an introduction to enterprise software engineering, lays the groundwork for what software engineering is, but specifically what it means to be an *enterprise* software engineer. The learning objectives for the module focus on what enterprise are, what they look like and are really supposed to do, and the unique challenges they encounter. We ground this content in the context of business-IT
alignment and the challenges inherent within. By couching the content in this way, we show the students that every decision made when creating an information system should back the overall business model of the parent entity.

Leading from this idea of business-IT alignment, the second module covers the business context. This unit contains several different models that help the students think about how products are designed and the business analysis aspects of enterprise software engineering. The learning objectives for the module are to learn about industry segments and determining business strategy; how an enterprise determines its niche and product offerings, how to translate a business strategy into an actual implementation, how information technology and computing help businesses, and how technology departments must work to maximally help the business. The content is entirely from business and economics literature, touching on concepts like competitive advantage, the Balanced Scorecard (Kaplan, 2000), the value chain (Porter, 1996), and a model of the five competitive forces (Porter, 1979). The students then use the concepts learned in the first two modules to start their term projects by performing a business analysis for their chosen companies.

Following the two module introduction to software-related business concepts, the third module covers software engineering processes. This module is the start of the core software engineering content. Its learning objectives are understanding what a software process is and their benefits; awareness of the two types of software processes, namely structured and agile; and familiarity with designing and customizing software processes. The key content in this module is that there are many prescribed software processes that
experts vouch for, but that a software engineer doesn’t have to follow any of them fully and is free to cherry pick the best facets of different processes, creating his own customized software process. However, the software engineering should consider the needs of the business and the parameters of the project to tailor the process to the project; he should not customize a process just because he does not like some part of it.

The next learning module shows examples of customized software processes in the form of case studies. The instructor freely chooses premade case studies from our case library and gives the cases to the students. These cases are a mixture of technical talks from large software companies like Microsoft and IBM, and full case studies that were created by our graduate students using the process described on page 40. The learning objectives of this module are merely a reinforcement of the concepts covered in the first three modules, as well as a preview of the material covered in the rest of the course.

Software processes are the umbrella under which the rest of the class follows. The next module starts covering the steps of the software development lifecycle (SDLC) with the first step: requirements. The learning objectives of this module are understanding the concepts of requirements and requirements identification, and familiarity with requirements work products and how they capture requirements. This is the point at which most traditional software engineering courses begin. We also introduce the course textbook (IBM, 1997) at this point, as it begins at the requirements stage as well. The students integrate requirements work products into their term projects at this point.
The sixth course component is somewhat of an interlude. The students are working consistently on their term projects at this point, but the tools that they have at their disposal are limited. To correct for this, we introduce a unit that focuses solely on the Unified Modeling Language (UML), a pictorial language for sketching, blueprinting, and designing systems. UML is a standard language for capturing work products in a consistent, widely understandable manner. The learning objective of this module are understanding of UML and how to create the various types of diagrams contained within it.

After learning about UML and how to use it, the students are ready to apply it in the most UML-intensive course module: analysis. The learning objectives of this module are familiarity with the concept of analysis and how it fits into the software development lifecycle, as well as the techniques used and work products associated with analysis. For this course, we show the students that you can apply analysis techniques at all three levels of abstraction for the system: the domain, problem (or goal), and solution levels. Domain analysis is the highest level and concerns how the enterprise as a whole works. When dealing with what actually needs to be addressed by a project, problem analysis is the level that is used. Finally, when analyzing exactly how a problem is solved, a software engineer performs a solution analysis. These three levels are incorporated into the term project and the students go back and revise their previous business analyses with the newly introduced techniques.

The eighth course module, software architecture, is where students start designing how the system will meet its non-functional requirements. The learning objectives for
this unit are: understanding software architecture and its relationship to the other stages of the software development lifecycle, how to design and analyze software architectures, and the industry standards for architecture frameworks. We introduce the concept of software frameworks through examples like Google Android, Java Enterprise Edition, and the .NET Framework. Students create the underlying architecture for their term projects including deployment strategies and design patterns. This module is the one in which most students struggle, as architectures are difficult to design without prior experience in the area.

At this point, the students’ term projects average about twenty pages of text and diagrams, which is a large amount of information for any incoming team members to understand. As such, we introduce project and risk management in the ninth learning module, to help the students begin to understand how teams larger than a few people have to work to get their work done. The learning objectives of this module are understanding the multiple aspects of project management and the accompanying techniques, familiarity with techniques for software estimation, and experience with the concepts and techniques of risk management. We use a book about the foundations of risk management (DeMarco & Lister, 2003) as the basis for this chapter, as enterprise systems are rife with risk that can be the undoing of technology-reliant companies when unaccounted for. The students develop project schedules, team management plans and risk management work products for their term projects for this module.

When students enter the course, they often have the misunderstanding that software engineering is merely the practice of creating and designing software well.
These students are typically confused about the course until they enter the tenth unit, which focuses on software design. The learning objectives for this module are an understanding of object-oriented design, the rationale behind it, and competence with responsibility-driven design. The main feature of this unit is the introduction of the design technique known as Class-Responsibility-Collaboration (CRC) cards (Beck & Cunningham, 1989), which focuses on noun and verb extraction from the problem statement and assignment into object classes and responsibilities. At this point, the students start designing the systems they have been analyzing for weeks.

One aspect of software engineering that students typically struggle with is the subject of the eleventh course module: testing. Software testing is a complex field and inadequate understanding of it leads to the bugs that consumers see in the systems they use every day. To combat this lack of knowledge, we designed this unit to illustrate testing in all of its forms. The learning objectives of this module are: be familiar with software testing and its goals, be exposed to the different stages of testing, be exposed to test methods and testing tools in both agile and structure processes; be exposed to the role of tests in the software development lifecycle, and gain competence with creating testing work products. The students develop a testing plan that covers the three domains of unit, integration, and system testing.

The final learning module for the course covers the final two stages of the software development lifecycle, deployment and maintenance, but does so under the broader term IT service management. Part of the Information Technology Infrastructure Library (ITIL), service management refers to the practices of managing systems and
people from deployment to end-of-life, including change management, configuration management, and release management (Steinberg et al., 2011). The learning objectives for this unit are: familiarity with IT service management and the ITIL framework, exposure to the stages of the service management lifecycle, and exposure to the key processes of service management. This unit serves to introduce the students to the level of complexity that is involved in large-scale enterprise systems that solve even more convoluted problems. The students do not have to produce additional work products for this unit, but they do need to incorporate the concepts into a new revision of their previous work.

After all of the course modules, the students have a complete project workbook that is intended to be part of their job portfolio when applying for their first career positions out of school. The projects average seventy pages in length when professionally formatted; this professional formatting is a requirement of the project, so the students can include it in their portfolios. Overall, the students do well on the projects, demonstrating that, despite the overwhelming amount of material, they are able to assimilate the skills of the software engineering trade and use them to analyze a business, discover its points of technological improvement, and create a thorough plan to solve those problems using a tailored software process.

Because of the amount of material covered and the structure of the novel curriculum, course management varies from the traditional classroom and requires its own considerations. The next section discusses the most salient management issues and the support structure we use to aid the instructors in successfully teaching the course.
Course Delivery and Management Elements

The concept of the inverted classroom creates unusual management issues for the instructors and course coordinator. First, we discuss the need for a particular kind of instructor that is difficult to find in traditional academia. Second, we note an unforeseen difficulty arising from the inverted structure of the curriculum and methods to contain and mitigate the issue. Lastly, we outline the traditional curricular materials and arguments as to why they remain a part of the course syllabus.

Being an enterprise software engineering course, the course is more heavily focused on industry than others in the department. Due to this, the requirements for the instructor to be an effective teacher differ widely from the requirements for traditional courses. We find that the optimal instructor for this course is someone who has deep experience in several roles in industry, prior teaching experience, and a willingness to adapt their classroom technique based on the reactions of students.

A wealth of experience is necessary because it allows the instructor to share anecdotes of how work occurs in a real enterprise. Lack of industry experience hampers an instructor and makes it more difficult to connect with students, since the course concepts are directly related to how software engineering works within the context of the enterprise. An instructor may know all of the course content very well, but without anecdotes about how it applies, this knowledge lacks the impact it needs to make an impression on the students.

Prior teaching experience is helpful as in all educational context, but in some ways experience with teaching in a professional context, whether through directing
training programs or mentoring employees, can be more advantageous than experience in academia. The reason for this is that experience in the academic instruction often comes through teaching through the traditional method of lecturing. This learned behavior is all too easy to fall back into, which undermines the inverted classroom pedagogy and limits the use of in-class time for class discussion. Professional teaching, on the other hand, often uses “softer” techniques like anecdotes, discussion, and activities that are more in line with the structure of this curriculum.

The novel curriculum we present in this thesis is very student-focused, which necessitates a certain degree of flexibility when handling the in-class interactions between the instructor and students. With the focus of in-class time being discussion- and activity-based, the students often set a flow to the class that the instructor did not plan. This is perfectly acceptable, as letting the students drive the discussion helps ensure that they are getting the maximal amount out of each class interaction, but it increases the burden on the instructor in that he cannot completely prepare each day’s in-class content and can merely anticipate the questions that the students will raise.

This interaction paradigm is difficult to adjust to for inflexible, lecture-driven instructors, as they are inevitably asked questions that catch them off-guard. For those that like to be seen as complete experts in their domain, this can be an uncomfortable situation and may require quick mental footwork or an unnatural behavior of them: replying with, “I don’t know.” Being comfortable with this reply is helpful when teaching this course, as long as one is willing to come back to the next class or the discussion board with a thorough answer.
We find that, for this curriculum, these qualities are the primary indicators of whether an instructor will be successful course instructors. However, they are not the sole determinant of an instructor’s success, as the course has an atypical structure when compared to a traditional course.

Because the inverted lectures are shown outside the classroom via prerecorded video, the lecturer is not necessarily the course instructor. We note that students felt unsure about who is in charge of the course: the instructor, or the disembodied voice on the video lectures. Some students feel uncomfortable with the arrangement, because they don’t want to ask questions of the instructors about the prerecorded lectures. These students feel that the instructors have not watched the videos – and in some cases they are right. Instructors have a tendency to use the prerecorded lectures as a crutch, relying on the students to ask questions and leading the discussion based on those questions, rather than planning ahead of time.

Combating the issue of classroom control can be done in a variety of manners. First, if the video lectures are not narrated by the instructor, but are narrated by another faculty member (e.g. the course coordinator), it is helpful to have the instructor and video narrator both in the first class of the term. When explained to the students, the roll of the video narrator becomes more clear, as the narrator is really a facilitator to starting the discussions the students will have with the instructor. In every other way, the instructor should be in control of the class so as to provide a united front to the students where they only ever interact with the instructor after the first day. This method helps eliminate some
of the confusion the students experience, as they will no longer wonder who “the voice” is and why he or she is narrating the videos instead of their instructor.

Second, if the instructors will have a long-term relationship with the course (i.e. teach the course over a long period of time or be involved in updating the course content), it is useful to have the instructors narrate some of the videos, or all of them for his or her section. This is a time investment in the same manner as every class preparation is, so if the program has high instructor turnover, as is often the case when using adjunct faculty, the instructors may not see the benefit of recording their own inverted lectures. However, when recording their own videos, the instructors become intimately familiar with the material and, in the long run, save class preparation time or free up time to include fresh content for the students, in the form of supplementary materials.

These supplementary materials can help instructors fill in the gaps left by the inverted lectures. However, in order to provide adequate resources to the students for them to successfully create their term projects, we found that we still require traditional textbooks. Students find translating the course content into work products for the term project to be difficult without adequate demonstration. As such, we provide one core textbook for the course, along with an assortment of supplementary books.

As a software process, the IBM Rational Unified Process (RUP) is an excellent example that allows teams to be very structured and thorough in their documentation, but still be flexible in how they present the information. These traits lead us to pick the standard RUP book as a course textbook (IBM, 1997). While it is a book aimed at a
single software process, we felt that the level of detail, yes ease of reading the book made it an ideal candidate as an example of how to think about a custom software process.

We give the students a list of supplementary materials at the end of each lecture. These are typically online articles, academic papers, or very focused books on each topic. As software engineering is a people-focused discipline, it is rapidly evolving and core books are always being replaced by newer versions. However, we find that some subfields have books that cover the topic in a broad manner, without focusing on specifics. For example, *Waltzing with Bears* (DeMarco & Lister, 2003) provides an in-depth consideration of risk management, from a viewpoint that focuses on broad ideas, like addressing risk at the earliest possible moment and planning ahead for dealing with eventual problems. This perspective allows us to use it as a resource for helping students address the risks to their projects without narrowly prescribing a process that might later be obsolete.

This type of high-level book helps guide students down a path without bogging them down in the details of a process; it points them in a direction while still allowing the process to be uniquely theirs. Following our constructivist philosophy, we allow students to make mistakes because we believe that is fundamentally how we learn.

The next chapter of the thesis lays out the evaluation methods we use to test the efficacy of the curriculum and describes the results we found through three different types of evaluation. We use both quantitative and qualitative methods for both formative and summative assessment in an preliminary evaluation of the curriculum, an in-depth look into the effects of the inverted format, and the effects of the agile workshop.
Chapter 4: Curriculum Evaluation

This chapter presents the evaluation methodology and result of four different evaluations of the novel curriculum. The first evaluation is an initial assessment of the curriculum to determine the level of acceptance of the curriculum with both students and instructors. The second study is an in-depth look into the experience that students and instructors have within the inverted classroom format. The final assessment determines the efficacy of the Lego workshop described on page 32.

Preliminary Evaluation

Any new curriculum requires evaluation to determine whether the new curriculum performs better than, worse than, or the same as its predecessor. Evaluation becomes even more important when changing the modality of teaching; for example, moving a conventional course to an online version, or in the case of this curriculum, a traditional lecture-based class to an inverted classroom. However, when making a drastic shift in format, there are bound to be implementation bugs that cause issues, but do not necessarily reflect the efficacy of the course.

To account for the problems with initial implementation, we ran the course for a year, resulting in three sections of the course before instituting any formal evaluation. This allowed the instructors to fix early issues without them biasing the results of the assessment. The only criterion for success with the method during the first year was whether the instructors qualitatively felt that the curriculum was working within the
bounds of using a new method. Essentially, this meant that any small problems were fixed on-the-fly and noted for the next iteration.

In the second year of the course, we implemented both formative and summative assessments. First put forth by Scriven (1967) and later refined to its current form by Bloom et al. (1971), formative assessment refers to any type of assessment that intends to improve student learning by providing feedback on the teaching method to the educator; in essence, it is about improving the tools of teaching and creating new ones. In contrast, summative assessment (Bloom et al., 1971) includes any evaluations given ex post facto in order to estimate student attainment or curriculum efficacy for purposes of grading students, instructors, or curricula.

We use two types of formative assessment of the students for this preliminary evaluation. The first is through weekly quizzes on which we give timely feedback in the next class. We give detailed feedback about what was correct and incorrect to every student to help show them what they had difficulty with on the quiz. We also advise the students who do poorly on the quizzes about study habits and the expectations for the course. In addition, this detailed feedback usually drives the discussion for the first half of the class period; we find that this helps all of the students, as even the ones who did well on the quiz correct or reaffirm their understanding of the content.

The second formative assessment of the students is the term project, which we have students submit in pieces at the end of each learning module. We mark up and provide comments on each individual section that the students submit, but do not give them an actual grade on the submissions. We hand back the projects in the next class
period and the students are then free to make revisions as they see fit based on the
comments they receive. These revisions are then submitted with the next section, so the
project workbooks are in a state of continual improvement throughout the semester. The
grades for the project are assigned on the quality of the final submission, which is the
entire workbook formatted to professional quality. We find that the students are generally
grateful for the comments, as they genuinely want to do well in the course, and that they
incorporate the suggested changes nearly unilaterally on each revision.

These two formative assessments also evaluate the course and help the instructor
to adjust his planned instruction for the class following each quiz and project submission
by addressing some common problems he sees on each. In addition to these two
instruments, we find through quiz grades that there are two sections in which the students
generally lack understanding: acceptance tests for non-functional requirements and the
concepts in risk management. We use this formative assessment to discover the need to
revise these two learning modules for the next time the class is taught.

For summative assessment of the students, we have two instruments for
evaluation. The first is the project grades at the end of the course. We use these grades to
measure the efficacy of the curriculum as a whole by comparing the results to the
previous offering of the course in a traditional format. We find that the inverted
classroom compares favorably with the old curriculum, in that there is no significant
change (neither a gain nor a loss) in the average grade received in the course. This
provided the initial verification that the method works at least well enough to continue
developing.
The second summative assessment for the students is the cumulative grades the students received between their quiz and final examination scores. The quiz scores show a lower grade on the first quiz (average score of 5.4/10) than the other quizzes, with the next lowest average being 7.1/10 on the third quiz. Through feedback, we attribute this to the students being unused to the inverted format and typically choose this quiz score as the single quiz to drop from their final grade. Otherwise, the quiz grades are comparable to the quizzes from the traditional format, as are the final examination scores. As such, this instrument is also supportive as an initial verification that the curriculum is viable.

Our summative assessment of the course uses four sets of instruments. First, we issue a single, open-ended question on the first day of class and the day of the final examination. This question, “describe how you would identify, design and develop a large-scale software application for an enterprise,” provides a baseline for the process the student will follow at the start of the class, and a measurement of the shift in their thinking at the end of the class. In the results of the questionnaire, 40% of the students show increased awareness and consideration of both business context and requirements identification. From a concept application standpoint, 75% of the students who, at the start of the class, only give vague ideas of what they would implement in their process, later name specific techniques they would use in the process at the end of the class. Also, 95% of the responses after the class show awareness of non-functional requirements and 25% talk about risk management when discussing project management. With the exception of risk management, these results are promising, and further verify the efficacy of the inverted curriculum.
The second summative assessment of the course is a set of interviews with a small number of students to gather student reactions to the course and the novel elements used within the new curriculum. The questions for these interviews are shown in Appendix A. The students uniformly enjoy the discussion-based format of the class. When asked how they studied for the course in comparison to their other classes, they indicate that they adapt traditional methods to the inverted format by printing the slides and taking notes along with the video, as if they were really sitting in the classroom. All of the interviews indicate that the agile development Lego game was a success, with students stating they had fun and really felt they learned what agile development means. The students rate the case studies highly as well, with all students rating each case study between a 3 and a 5 on a Likert scale of 5. However, we note some pushback from the students on the topic of the project. The students felt that the project was tedious and didn’t enjoy it because they weren’t “doing anything real,” and were “just making things up on paper.”

As a weak third instrument, we use the university-mandated Student Evaluations of Instruction (SEIs) system for summative assessment. Student participation in these online evaluations is notably low, with only fifteen percent of the students responding. The responses on the SEIs are inconclusive, as approximately half of the students respond positively to the course format and the other half respond negatively, in roughly equal strength of opinion. However, every student assesses the workload in the course as higher than a typical, comparably difficult course and higher than they expected. This is a mild cause for concern and is something that we should consider later.
The final summative instrument that we use for this preliminary evaluation is an interview with the two instructors of the course during this term. The questions for these interviews are shown in Appendix B. The instructors indicate that preparing for teaching in the inverted classroom is more time-intensive than in a traditional lecture format, as they feel they need to know the specific content in the prerecorded lectures more closely than they do when lecturing. Despite this increased time burden, they both rate the method as effective and state that it actually improved their ability to teach. This improvement is largely due to the extra time they have to interact with students and bring their own experience into the classroom, without feeling like they are spinning off on a narrative tangent.

Overall, both formative and summative assessments for this preliminary study are overwhelmingly positive and encourage the further development of the course. Also, the responses of the students inspire some of the design and structure of the second study, which we discuss next.

_In-Depth Study of Student and Instructor Experiences_

The preliminary curriculum assessment raised several important research questions that we want to further study. The primary source of these questions is interviews with the students, as the interviews were semi-structured and allowed the students a broad area in which to respond and talk about their thoughts on the course. Also, since the interviews were conducted by a graduate student who was not involved with the grading of the course, the students felt comfortable commenting on what they liked and did not like about the course. The research questions that interest us include:
1. Does the inverted classroom place a greater time burden on study habits?
2. Does the inverted format increase the quantity of discussion in the classroom?
3. Does the inverted format increase the quality of discussion in the classroom?
4. Are weekly quizzes an effective means of ensuring students watch the lectures?
5. Does the inverted format lessen the preparation time for instructors?
6. Does the inverted classroom help lessen the gap in instruction quality across multiple instructors of different skill level?

The first four questions come directly from interacting with the students in an informal setting, while the last two concern the effect that the inverted format has on instructors. For this research, we want to collect targeted metrics in order to empirically answer our research questions and verify the results through the use of student and instructor interviews.

Throughout the course of this assessment study, we worked with three separate instructors who taught simultaneous sections of the software engineering course. The first participant was the course coordinator and instigator of this research. He has taught in the university setting for nearly a decade and has taught using an inverted classroom prior to the study. The second instructor was an adjunct lecturer from an industry partner who has taught in both a university and a professional setting. The third instructors was also an adjunct lecturer from an industry partner. He has not taught in a university prior to this study, but has taught in (and helped design) the Distinguished Engineer program in his
company. Thus, we had a distribution of dissimilar instructors and can use this to validate whether the inverted classroom helps “even the playing field” between the three.

Each instructor uses the prerecorded lectures for each learning module, as described on page 48. The lectures are distributed the same way to the students in each class, with the students choosing the format they wish to use for the prerecorded lectures. In order to ensure that the students actually watch, listen to, or read the lectures before coming to class, we use weekly quizzes on discussion days that account for fifteen percent of the course grade.

*Data Collection*

For instruments, we distribute questionnaires on the third, sixth, and ninth weeks of the term (out of a ten-week quarter) to all of the students present in each class. The questions for these surveys are shown in Appendix C. The intent of these interim questionnaires is to collect continuous measurements that show how the students change their attitudes throughout the course and gauge their study habits; specifically, they target the first research question on page 66 and overall acceptance measure for the course.

We also conduct interviews with a total of nine students, three from each lecture, and each of the two new instructors. The questions for the students are shown in Appendix D and the questions for the instructors in Appendix E. Both sets of interviews are conducted by a graduate student to help make the interviewees comfortable, and are semi-structured through the questions, but allowing open-ended answers.

The last instrument that we use is the set of anonymized grades from each of the three sections as a summative evaluation of the performance of the students in each class.
We used the grades as an indicator of instructor quality, as the students in a section with a high quality instructor should perform better than students in a class with a low quality instructor. We used this notion to answer the sixth research question on page 66.

**Analysis**

After the collection of this data, we analyzed it using appropriate techniques. For the student surveys, we computed descriptive statistics, as shown in Table 2. The response rate for the morning section is 83%, the first evening

<table>
<thead>
<tr>
<th>#</th>
<th>Response</th>
<th>Frequency</th>
<th>Percent</th>
<th>Valid %</th>
<th>Cumulative %</th>
</tr>
</thead>
<tbody>
<tr>
<td>Q1</td>
<td>0 hours</td>
<td>2</td>
<td>1.3%</td>
<td>1.3%</td>
<td>1.3%</td>
</tr>
<tr>
<td></td>
<td>1-6 hours</td>
<td>128</td>
<td>83.1%</td>
<td>83.1%</td>
<td>84.4%</td>
</tr>
<tr>
<td></td>
<td>6-12 hours</td>
<td>23</td>
<td>14.9%</td>
<td>14.9%</td>
<td>99.4%</td>
</tr>
<tr>
<td></td>
<td>12+ hours</td>
<td>1</td>
<td>0.6%</td>
<td>0.6%</td>
<td>100.0%</td>
</tr>
<tr>
<td>Q2</td>
<td>lecture</td>
<td>144</td>
<td>93.5%</td>
<td>94.7%</td>
<td>N/A</td>
</tr>
<tr>
<td></td>
<td>reading</td>
<td>59</td>
<td>38.3%</td>
<td>38.8%</td>
<td>N/A</td>
</tr>
<tr>
<td></td>
<td>project</td>
<td>93</td>
<td>60.4%</td>
<td>61.2%</td>
<td>N/A</td>
</tr>
<tr>
<td>Q3</td>
<td>PowerPoint notes</td>
<td>98</td>
<td>63.6%</td>
<td>65.3%</td>
<td>N/A</td>
</tr>
<tr>
<td></td>
<td>audio</td>
<td>13</td>
<td>8.4%</td>
<td>8.7%</td>
<td>N/A</td>
</tr>
<tr>
<td></td>
<td>video</td>
<td>104</td>
<td>67.5%</td>
<td>69.3%</td>
<td>N/A</td>
</tr>
<tr>
<td>Q4</td>
<td>very easy</td>
<td>4</td>
<td>2.6%</td>
<td>2.6%</td>
<td>2.6%</td>
</tr>
<tr>
<td></td>
<td>easy</td>
<td>41</td>
<td>26.6%</td>
<td>26.6%</td>
<td>29.2%</td>
</tr>
<tr>
<td></td>
<td>neither easy nor difficult</td>
<td>92</td>
<td>59.7%</td>
<td>59.7%</td>
<td>89.0%</td>
</tr>
<tr>
<td></td>
<td>difficult</td>
<td>17</td>
<td>11.0%</td>
<td>11.0%</td>
<td>100.0%</td>
</tr>
<tr>
<td></td>
<td>very difficult</td>
<td>0</td>
<td>0.0%</td>
<td>0.0%</td>
<td>100.0%</td>
</tr>
<tr>
<td>Q5</td>
<td>disliked it a lot</td>
<td>1</td>
<td>0.6%</td>
<td>0.7%</td>
<td>0.7%</td>
</tr>
<tr>
<td></td>
<td>disliked it</td>
<td>5</td>
<td>3.2%</td>
<td>3.3%</td>
<td>3.9%</td>
</tr>
<tr>
<td></td>
<td>neither liked nor disliked it</td>
<td>68</td>
<td>44.8%</td>
<td>44.4%</td>
<td>48.4%</td>
</tr>
<tr>
<td></td>
<td>liked it</td>
<td>73</td>
<td>47.4%</td>
<td>47.7%</td>
<td>96.1%</td>
</tr>
<tr>
<td></td>
<td>liked it a lot</td>
<td>6</td>
<td>3.9%</td>
<td>3.9%</td>
<td>100.0%</td>
</tr>
</tbody>
</table>

See Appendix C for the specific wording for each question.
section is 95%, and the second evening section is 91%, giving an overall response rate of 90%. From the aggregate descriptive statistics, we then report general findings as discussed later in this chapter.

When considering the results of the interview questions, we group the responses by topic for both student interviews and instructor interviews. We divide the student interviews into five topics: student satisfaction with the inverted format (six questions), evaluation of the discussion in the class (six questions), how effective the quizzes are as motivators to watch the lectures (four questions), their impressions on the active learning games (six questions), and the impact of project presentations (two questions).

We split the instructor interview questions into five topics as well: knowledge elicitation questions to gain information on the background of the instructors (two questions), how they prepare for teaching this course (three questions), how effective they feel the motivational methods to watch the lectures are (two questions), and how the inverted format affected their teaching (six questions).

We then analyze the anonymized student quiz grades using a one-way ANOVA, with each quiz as a variable of analysis, excluding missing grades from the analysis. The student grades provide us with an indirect measure of instructor quality. Because the instructors all use the same primary method of instruction – the prerecorded videos – we feel that this is an adequate instrument to test whether the inverted format makes the quality of instruction more homogeneous across instructors of different skill level. For this analysis, our null hypothesis is that there will not be a statistically significant different between the mean grades on each quiz; that is, $H_0: \mu_0 = \mu_1 = \mu_2$. Our
alternative hypothesis is that there is a significant difference between the mean grades on each quiz; that is, $H_A : \exists i, j : \mu_i \neq \mu_j$.

Results

Individually, the instruments were not strong enough to make conclusions. However, when we integrated the analyses of the three instruments, we gained a holistic view of the experience in the inverted classroom in both a qualitative and quantitative manner. The quantitative data from the surveys and student grades gave us a notion of the results we saw and the qualitative interviews helped to confirm those suspicions.

The first research question we addressed was the question of whether the inverted classroom increases the study burden on students. The results of the student interviews showed conflicting results when it came to time spent on the course. Two students said they spent around twice as much time on this course than on their other courses. Four students quoted approximately the same amount of time for this course compared to their other courses, and two students told us that they spent less time. This seems to indicate that the amount of time students spend on the course is dependent on their personality.

We did not ask about their study habits for other courses specifically, as it was outside the scope of this research, so the qualitative results may suffer from a hidden bias. However, the quantitative results were conclusive that the students spend an appropriate amount of time on the course.

By looking at the survey results presented as descriptive statistics in Table 2, Question 1, we saw that 84.4% of the students in the course spent six hours or less preparing for the class. The course is a three credit hour, senior-level course. Following
the popular heuristic of studying for three hours per each credit hour (Van Blerkom, 2011), the students should spend approximately nine hours studying for the course. Given that only one student said he spent more than twelve hours studying, the evidence shows that the inverted format fell within acceptable bounds for the median student.

We next addressed the question of whether the new curriculum increased the quantity of discussion. In order to verify our suspicions, we asked students during the follow-up interviews whether they felt there was more discussion in the course than a comparable one. Eight of the nine students said that there was noticeably more discussion. The single student who said there was not more discussion has a different account than the other two students from his section and, when we clarified, his response was that there was a series of long pauses while the instructor waited for students to ask questions, but that the discussion did finally happen. Thus, the evidence suggested that there was an increase in discussion with the inverted classroom.

We also address the question of whether the quality of discussion was higher in the inverted classroom through the interviews with the students. Four of the nine interviewed students said that the quality of discussion was noticeably better than in other comparable classes; however, none of them agreed that the quality was substantially better. One of those four students said that she felt the quality was slightly better, even when compared to a course that has an in-depth discussion component. The remaining five students all agreed that there was not enough evidence to say that the quality of the discussion was better. Their reasoning was that there was a wide disparity in the relative skill levels of the students in the class, so some students asked questions that are far
above the skill of their younger classmates, leaving the latter students without a real understanding of the discussion. When flipping that arrangement, some lesser skilled students asked questions that lead to an “easier” discussion that the more skilled students said felt like a waste of time. Because of this split opinion, we did not have enough evidence to definitively state whether the quality of discussion was higher in the inverted classroom.

Our third research question asked whether using quizzes was an effective motivator for ensuring students watched the prerecorded lectures. This is a core concern of the inverted classroom; if students do not watch the prerecorded lectures, then they are effectively receiving no lecture and learning entirely based on the discussions. It is impractical for the instructor to attempt to teach entirely through discussion in a technical course, so the recordings are vital to the class. We addressed this question through interviews with both the students and instructors. Eight of the nine students said that the quizzes were effective in making sure they watched the videos. In addition, they felt that the quizzes helped to guide their study, as the open-ended questions forced them to go through the lectures more carefully than they otherwise would. The one dissenting student said that skimming through the lecture notes was enough to do “well enough” on the quiz to get partial credit, which satisfied his personality.

The instructors both feel that the quizzes ensure students watch the lectures. However, they both agree that the quizzes have an unintended consequence: the students get very anxious about the quizzes, but for different reasons. One instructor feels that the students do not handle the time limit for the quiz well and rush through, which degrades
their performance. The other feels that there are simply too many quizzes and the students feel inordinately pressured by the presence of a quiz every week of the term. Both instructors agree that there is a need for a better motivational mechanism for this purpose.

The results say that the students are more likely to watch the lectures because of the quizzes. Depending on the student, some students may only skim the lecture beforehand, as they feel they can do “good enough” through partial credit. But overwhelmingly, we find that the students want to do well in the course and thus watch the lectures. However, we must use caution when using quizzes as an enforcement mechanism. The evidence shows that, from an instructor’s point of view, the anxiety induced by the quizzes may not be worth the extra eyes on the video. Thus, it is a balancing act for instructors who wish to implement an inverted classroom format.

The next result we address is the question of whether the inverted classroom increases the preparation time for an instructor teaching the course. We rely on self-reporting from the instructors during their interviews to understand this phenomenon. We find that the amount of time an instructor spends preparing is entirely dependent on his personality and experience.

One instructor is experience in teaching in academia, having been a lecturer on occasion throughout his career. Thus, he is accustomed to the traditional classroom and want to have content planned for a maximal amount of class time. To do this, he watches each prerecorded lecture two or three times to find what he feels are the weaknesses in each learning module. Then, he considers how he can address the gap in the lecture and
looks for online materials to supplement the lecture. This is very time-consuming for him and leads to him spending two to three times more time preparing for class than he does with a comparable traditional class. However, he notes that his preparation for a traditional class may be less thorough, since with the inverted classroom, he can act as a foil for the prerecorded lecture.

The other instructor leads the Distinguished Engineer training program at his company, in which he teaches a curriculum that is very similar to the content of this course. As such, he feels that he does not require very much time to prepare for the course, spending approximately half the amount of time that he would spend if he had to lecture in class. His primary use of preparation time is spent in finding guest lecturers from among his employees that come speak about the narrow topics covered in the learning modules. He also searches through non-classified work products that he can bring in from his industry work to share with the students, so they can experience what work products look like in a real-world enterprise.

Overall, we feel that we can say that, on a day-to-day basis, the amount of time needed to prepare for class in the inverted classroom is less than a traditional classroom. However, there are different concerns for the inverted classroom that can lead certain personality types to spend more time than average when preparing for the course. Instructors must be aware of the circumstances that cause this if implementing an inverted classroom format.

Our final research question from this study asks whether the inverted classroom helps to increase the homogeneity of instruction between different instructors when
compared to a traditional format. To address this, we use the students’ anonymized quiz grades across the three sections of the course that are taught by three different instructors. We apply a one-way ANOVA to the data, with each quiz as a variable of analysis. The results of this analysis are shown in Table 3.

The results of this analysis are inconclusive. On six of the eight quizzes, we are able to reject the null hypothesis that there is no statistically significant difference between the distribution of scores. However, on the other two, we are unable to reject the null hypothesis, which means that it is possible that the samples were chosen from the same distribution. Since the number of quizzes in which we can say that there is likely a difference in distribution is greater than the number of quizzes that are similar, these results indicate that the instruction is dissimilar, which answers the research question in the negative.

However, the instrument we use for this analysis is flawed. We implicitly assume that the prerecorded lecture is the main determinant of quiz score by using it as an instrument. But, when we consider the results from the instructor interviews and how

| # | $\mu_1$ | $\mu_2$ | $\mu_3$ | p-value | $p < 0.05?$
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2.47</td>
<td>2.37</td>
<td>2.67</td>
<td>0.353</td>
<td>No</td>
</tr>
<tr>
<td>2</td>
<td>2.44</td>
<td>2.63</td>
<td>2.12</td>
<td>0.0307</td>
<td>Yes</td>
</tr>
<tr>
<td>3</td>
<td>2.36</td>
<td>1.71</td>
<td>2.82</td>
<td>5.31e-07</td>
<td>Yes</td>
</tr>
<tr>
<td>4</td>
<td>1.83</td>
<td>2.08</td>
<td>1.42</td>
<td>0.0301</td>
<td>Yes</td>
</tr>
<tr>
<td>5</td>
<td>2.33</td>
<td>1.84</td>
<td>2.27</td>
<td>0.0504</td>
<td>No</td>
</tr>
<tr>
<td>6</td>
<td>2.42</td>
<td>2.03</td>
<td>2.58</td>
<td>0.0237</td>
<td>Yes</td>
</tr>
<tr>
<td>7</td>
<td>2.58</td>
<td>2.03</td>
<td>2.70</td>
<td>0.000672</td>
<td>Yes</td>
</tr>
<tr>
<td>8</td>
<td>1.83</td>
<td>2.26</td>
<td>2.76</td>
<td>8.45e-05</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Table 3: The means and p-values for the ANOVA between quiz scores.

Quizzes were graded on a scale of 0-3.
much their methods of preparation are different, we see that the prerecorded lecture is merely a small part of the content that the students receive. This invalidates our assumption and shows that we need a different instrument to test this hypothesis.

This study shows that the effects that we want to happen in the inverted classroom largely occur, which is a good indication of the efficacy of the curriculum. Further study is required to understand how the inverted classroom “levels the playing field” between instructors, due to a faulty instrument. Despite this, we feel that the curriculum is shown to be a valid instructional method that students enjoy, instructors find stimulating, and that the instructional method accomplishes the goals that it intends. Next, we discuss the study of the agile Lego workshop we use as the foundation of the active learning component of the curriculum.

**Efficacy of Active Learning**

The agile Lego workshop is a key component of our curriculum. As such, it requires its own evaluation separate from the rest of the course. For this study, we collect quantitative data from the students via a questionnaire that we issue at the end of every Lego workshop. We use the questionnaires from two sections of the course taught simultaneously by one instructors. The sample size for the study is 78 students, split almost equally between the two sections, with one section containing 37 students and the other holding the other 41.

We construct the questionnaires by selecting 8 questions out of a question bank. Five of the eight questions are always the same and attempt to measure two things: first, the level of engagement the students have during the workshop; second, we check three
core vocabulary terms covered only in the prerecorded lecture to see if they watched the video before the workshop. We choose the other three questions randomly from a bank of five questions that ask about concepts covered only by the workshop. The reason for this question format is to check whether the students get more questions correct from the workshop than from the lecture. Each question only has one correct answer, but the other answers are all written to intuitively “sound” right to make guessing the answer more difficult. The four questionnaires that we use are shown in Appendix F.

Since we issue the questionnaires immediately after the workshop, they are a weak indicator of long-term concept retention, but an excellent measure of short-term recall. To complement these properties, we use the students’ grades on the final examination questions as an indicator of long-term retention. We compare their scores on

Table 4: Descriptive statistics analyzing student acceptance of the agile workshop.

<table>
<thead>
<tr>
<th>#</th>
<th>Response</th>
<th>Frequency</th>
<th>Percent</th>
<th>Valid %</th>
<th>Cumulative %</th>
</tr>
</thead>
<tbody>
<tr>
<td>Q1</td>
<td>5 – agree</td>
<td>47</td>
<td>60.3%</td>
<td>60.3%</td>
<td>60.3%</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>25</td>
<td>32.1%</td>
<td>32.1%</td>
<td>92.3%</td>
</tr>
<tr>
<td></td>
<td>3 – neutral</td>
<td>3</td>
<td>3.8%</td>
<td>3.8%</td>
<td>96.2%</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>3</td>
<td>3.8%</td>
<td>3.8%</td>
<td>100.0%</td>
</tr>
<tr>
<td></td>
<td>1 – disagree</td>
<td>0</td>
<td>0.0%</td>
<td>0.0%</td>
<td>100.0%</td>
</tr>
<tr>
<td>Q2</td>
<td>5 – agree</td>
<td>62</td>
<td>79.5%</td>
<td>79.5%</td>
<td>79.5%</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>11</td>
<td>14.1%</td>
<td>14.1%</td>
<td>93.6%</td>
</tr>
<tr>
<td></td>
<td>3 – neutral</td>
<td>3</td>
<td>3.8%</td>
<td>3.8%</td>
<td>97.4%</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>1</td>
<td>1.3%</td>
<td>1.3%</td>
<td>98.7%</td>
</tr>
<tr>
<td></td>
<td>1 – disagree</td>
<td>1</td>
<td>1.3%</td>
<td>1.3%</td>
<td>100.0%</td>
</tr>
<tr>
<td>Q8</td>
<td>1 – disagree</td>
<td>30</td>
<td>38.5%</td>
<td>38.5%</td>
<td>38.5%</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>29</td>
<td>37.2%</td>
<td>37.2%</td>
<td>75.6%</td>
</tr>
<tr>
<td></td>
<td>3 – neutral</td>
<td>11</td>
<td>14.1%</td>
<td>14.1%</td>
<td>89.7%</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>5</td>
<td>6.4%</td>
<td>6.4%</td>
<td>96.2%</td>
</tr>
<tr>
<td></td>
<td>5 – disagree</td>
<td>3</td>
<td>3.8%</td>
<td>3.8%</td>
<td>100.0%</td>
</tr>
</tbody>
</table>
the questions dealing with agile development principles to the ones that have no agile component. Since the final examination is several weeks after the workshop, the students must have either internalized the concepts or studied them in particular for their grades to be higher on the agile development questions than the rest of the examination.

We analyze the questionnaires by first calculating descriptive statistics, as shown in Table 4. These statistics show that an overwhelming majority of students found the Legos, as a medium for instruction, to be effective, as 96.2% of respondents rate the questions between “agree” and “neutral” on a five-point Likert scale. Similarly, 97.4% of respondents rate the statement, “Legos made this subject more interesting,” between “agree” and “neutral” on the same five-point Likert scale. Finally, when we ask whether a lecture would have been more effective in teaching the concepts, 89.7% of the students

Figure 5: Student responses measuring enjoyment in the agile workshop.
rate the statement between “disagree” and “neutral,” which mirrors the results from Question 2. We flip the response to Question 8 in order to catch any students who rush through and just mark the same answer. The indication is that few, if any, students answered the questionnaire thoughtlessly, as the results closely match. The responses are shown in graphical form in Figure 5, for comparison.

In the questionnaire, we asked a mix of questions; five of the questions check vocabulary that is only covered in the agile workshop, while two of the questions are about terms that are only taught in the prerecorded lecture. Through this dichotomy, we assess the efficacy of the agile workshop versus the inverted lectures for teaching about
agile development. The results of analyzing the student answers to these questions are shown in Figure 6.

We note that there are two questions that students noticeably struggle with: W-3 and L-2. Since there is one question from the lecture-only set and one from the workshop-only that receive such poor results, we can only hypothesize that the instruction in both the agile workshop and the lecture failed for those terms; we cannot say that one technique fails more than the other. However, the mean percentage on the two lecture-only questions is notably lower than the mean score on the workshop-only set. This confirms our suspicions that much the knowledge required for agile development is tacit knowledge. That is, agile development is difficult to teach to another

Figure 7: Students’ exam question grades.

The final two columns bars show the average of the aggregate agile questions – questions 1 and 7 – and the average of the aggregate non-agile questions. The error bars show one standard deviation of uncertainty.
person through strictly written or oral communication and requires practice to teach it. This result verifies that the workshop is a necessary component to the curriculum, if the students are to learn about agile development.

The final instrument for the agile workshop is an analysis of student grades on the final examination. As stated previously, the questionnaires are given to the students immediately following the workshop, so the knowledge they gain is fresh in their minds when we ask them the follow-up questions. The true goal of education, however, is the long-term transfer of knowledge that students can recall weeks, months, or even years later. As such, we use the final examination scores as an indicator of this long-term recall. The results of this analysis are shown in Figure 7.

On the final examination, we issue two questions that are directly related to agile development principles – shown in Figure 7 as Questions 1 and 7. For this analysis, we calculate the means and standard deviations of each question and then compare the aggregate scores of the agile-related questions and the non-agile questions. As seen in the figure, there is no significant difference between the aggregate agile and non-agile scores. From this result, we conclude that the instrument shows that there is not a notable difference in efficacy between the inverted classroom and the active learning workshop. The result is a weak one, however, and the question requires further study using stronger indicators.

**Summary**

This chapter showed the results of three studies conducted on components of the novel curriculum, in an effort to evaluate its efficacy. We first carried out a preliminary
evaluation of the curriculum, to ensure that it performed well enough to continue
developing it. Next, we studied the inverted classroom to a greater depth to discover the
ways in which it affected students and instructors. Finally, we evaluated the agile
workshop component of the curriculum. However, this evaluation is not the end of the
research that must be done to discover the strengths and weaknesses of the curriculum.
We conclude this thesis with discussion about what the next steps in developing this
program should be.
Chapter 5: Conclusions and Future Work

As computing systems become increasingly ubiquitous in the information age, software engineers have an ever-expanding opportunity to affect the way that people live their lives. By designing systems that help workers get their work done, deliver immediate value to their users, and meet the needs of their customers, software engineers will be instrumental in defining the future. However, without proper instruction, freshly graduated software engineers will struggle to make valuable contributions early in their careers, as current engineering curricula do not teach the social aspects of the field, nor the focus on delivering stakeholder value early and continuously. This research addresses the need for a new educational paradigm through the introduction of a novel curriculum and the accompanying evaluation to measure its efficacy.

The designed curriculum uses the inverted classroom format to make the classroom a more social, collaborative learning environment. By using active learning games to teach concepts that rely heavily on tacit knowledge, this curriculum manages to adequately teach skills that are difficult to transfer via traditional methods like lecture and assigning readings. It also increases the amount of discussion about each learning module, which allows students to construct their understanding of the field from the learnings and misconceptions of their classmates, as well as check their understanding of the material before leaving the course. Lastly, the curriculum uses case studies to provide the students with understanding of how software processes work and are customized in
the context of real problems. This enables students to see how processes evolve when implemented in real work, which is often different than the way they were planned at the start of a project.

The ability to create case studies that highlight the importance of particular concepts is particularly important to this curriculum. As such, this research presents a novel, structured method for creating cases from the student-led project work that is common to many graduate research programs. This allows software engineering programs to utilize a previously untapped resource for capturing the workings of these student-led projects and channeling them into an outlet that delivers continual value for the program. This method has been used to create a small library of cases that are repeatedly used in the classroom to show the real-world consequences of decisions.

No curriculum should be implemented as a regular part of an educational program without evaluation to show that it is effective at achieving its educational goals. This research presents three separate evaluations of the developed curriculum that show its effectiveness and verify that it enables students to internalize the concepts. The first evaluation is a preliminary analysis of the curriculum and the initial student reactions to the course. The second takes an in-depth look into the experiences of students and instructors in the curriculum and shows some pitfalls that can occur when using the curriculum. Lastly, the third study measures the effectiveness of the agile workshop as an educational component when compared to the inverted classroom and shows that it is just as effective from an educational standpoint.
However, this research is not without its limitations. First, as a measure of the effects of the inverted classroom on the equality of perceived instruction between different teachers, taking the quiz grades as the variables of analysis was ineffective. This analysis assumed that the prerecorded lecture was the determinant of student success in the course. In reality, each instructor brings a wealth of supplementary material into the class discussion sessions that affects the learning of students within their class. This causes the instruction quality to vary widely between classes, as students might learn better from one supplement than they do from another. Perhaps because of the presence of the supplementary material, the homogeneity of instruction is actually lessened by the inverted classroom. This phenomenon requires further study as part of the future work.

Additionally, the evaluation of the quality of discussion relies entirely on qualitative evidence through the interviews with students and instructors. To accurately measure the quality, some quantitative measure is required. Discourse analysis, a set of techniques for analyzing conversation and oral and written communication, may provide the tools necessary to conduct this quantitative analysis. By recording several discussion sessions and then breaking down the conversations held using the tools of discourse analysis, a researcher could obtain a measure of the quality of discussion using a metrics-based method. These results could then be applied to discussions in different classes – either within computer science or across disciplines – to compare the quality of discussion between the classes to see if this curriculum inspires better dialogue between its students. This is another study that can be conducted as future research.
From a course development standpoint, the quality of the prerecorded lectures could be measurably improved by incorporating animation instead of relying on static slides for the visuals. By using principles of visual communication and film direction, the visual could potentially provide as much, if not more, information to the students as the audio lecture. This improvement would help students who are primarily visual learners to learn more from the lectures, which would then allow them to participate at a higher level in the class discussions and help their fellow classmates. An animated lecture could potentially be used to engage an outside audience in the program, either through professional development courses or as popular media to a mass audience. The increased capabilities for the lectures to communicate their content would be an excellent development of the course curriculum and is a significant source of future work.

Finally, further developing the concept of non-traditional media as educational tools is a worthwhile avenue for future work. The engagement and enjoyment that students exhibit during the agile Lego workshop shows that bringing media that are traditionally viewed as playthings into the classroom can actually spur students to learn more, rather than distract them from the learning. As such, the creation of different Lego workshops is a possible avenue of future work. For the first such improvement, researchers could develop a game to teach the process and principles of test-driven development, a common practice in agile workflows. By using the Legos to first create a test for the product, then actually build the product that meets those tests, students could learn about the “red, green refactoring” workflow of creating tests, refactoring the code
base, and then creating software to pass the implemented tests, in an fun, interactive, tactile manner that would appeal to students in the same manner as the agile workshop.

In conclusion, software engineering is a fast-moving target for education. By focusing on principles and collaborative learning, the curriculum described in this thesis is constructed to adapt to this ever-changing situation. Instructors guide students through the course based on their own experiences and the techniques and concerns they witness in industry. By using industry-based adjunct faculty, the students in this curriculum receive up-to-the-date knowledge about what enterprise systems are used by companies while taking the class. This better prepares them for the immediate job market and jump-starts their careers. Using this curriculum and expanding upon its principles of engaging students through collaborative learning, active learning games, and case studies tailor-made for the course.
References


Appendix A: Interview Questions for the First Student Evaluation

1. You were presented with two case studies of real-world projects and how the development process worked in each. These two case studies were the EcoFlow project and the Microsoft agile development seminar. How would you rate, on a scale of 1-5 with 1 being very ineffective and 5 being very effective, the effectiveness of using case studies to show the use of course principles?

2. What, in particular, did you learn from the case studies?

3. Do you feel that the use of case studies to ground some topics helped or hindered your ability to learn?

4. You were given an opportunity to participate in an active learning activity in the form of the Lego Agile Development boot camp. How would you rate, on a scale of 1-5 with 1 being ineffective and 5 being very effective, the effectiveness of the agile Lego workshop in teaching agile development principles?

5. What, in particular, did you learn from the agile Lego workshop?

6. How do you feel that using this method to teach agile principles helped or hindered your ability to learn?

7. The course material was presented in an “inverted classroom”. This presentation method was intended to improve in-class discussion and your ability to analyze and evaluation the course material. How would you rate, on a scale of 1-5 with 1 being
ineffective and 5 being very effective, the effectiveness of the inverted classroom for teaching the material in this course?

8. Had you ever had a class taught with this method before?

9. How do you feel the inverted classroom affected your ability to learn the course material?

10. How often did you watch the lectures before coming to the class on each topic?

11. What was the methodology you used to incorporate the content from the inverted lectures with the in-class discussion?
   a. What problems did you have in combining the content from the two means of presentation?
Appendix B: Interview Questions for the First Instructor Evaluation

1. How would you rate, on a scale of 1-5 with 1 being very ineffective and 5 being very effective, your success in teaching the material to students?

2. What are your feelings on the use of this approach? Do you feel that it improved your ability to teach? Do you feel it hindered your ability to teach?

3. Where, if at all, did you excel in effectively teaching while using this approach?

4. Where, if at all, did you struggle in effectively teaching while using this approach?

5. What skills do you think are necessary in order to successfully use this approach?

6. How are these skills different than standard skills you feel are required to be an effective teacher?
Appendix C: Student Surveys

On questions 1, 3, and 4, please circle the answer that best matches your experience. On Question 2, please provide a brief explanation.

1. How many hours did you spend preparing for CSE757 this week?
   - (a) 0 hours
   - (b) 1-6 hours
   - (c) 6-12 hours
   - (d) 12+ hours

2. How did you spend that time? (Example: 1 hour studying lecture, 2 hours reading, 1 hour project)

3. What method(s) did you use to access the lecture for this week? (May circle more than one)
   - (a) Read the Powerpoint notes
   - (b) Listened to the Powerpoint audio
   - (c) Watched the video

4. How difficult was this unit for you?
   - (a) Very easy
   - (b) Easy
   - (c) Neither easy nor difficult
   - (d) Difficult
   - (e) Very difficult

5. How much did you like this unit?
   - (a) Disliked it a lot
   - (b) Disliked it
   - (c) Neither liked nor disliked it
   - (d) Liked it
   - (e) Liked it a lot
Appendix D: Interview Questions for the Second Student Evaluation

1. You were presented with two case studies of real-world projects and how the development process worked in each. These two case studies were the EcoFlow project and the Microsoft agile development seminar. How would you rate, on a scale of 1-5 with 1 being very ineffective and 5 being very effective, the effectiveness of using case studies to show the use of course principles?

2. What, in particular, did you learn from the case studies?

3. do you feel that the use of case studies to ground some topics helped or hindered your ability to learn?

4. You were given an opportunity to participate in an active learning activity in the form of the Lego Agile Development boot camp. How would you rate, on a scale of 1-5 with 1 being ineffective and 5 being very effective, the effectiveness of the agile Lego workshop in teaching agile development principles?

5. What, in particular, did you learn from the agile Lego workshop?

6. How do you feel that using this method to teach agile principles helped or hindered your ability to learn?

7. The course material was presented in an “inverted classroom”. This presentation method was intended to improve in-class discussion and your ability to analyze and evaluation the course material. How would you rate, on a scale of 1-5 with 1 being
ineffective and 5 being very effective, the effectiveness of the inverted classroom for
teaching the material in this course?

8. Had you ever had a class taught with this method before?

9. How do you feel the inverted classroom affected your ability to learn the course
   material?

10. How often did you watch the lectures before coming to the class on each topic?

11. What was the methodology you used to incorporate the content from the inverted
    lectures with the in-class discussion?

   a. What problems did you have in combining the content from the two means of
      presentation?
Appendix E: Interview Questions for the Second Instructor Evaluation

1. How would you rate, on a scale of 1-5 with 1 being very ineffective and 5 being very effective, your success in teaching the material to students?

2. What are your feelings on the use of this approach? Do you feel that it improved your ability to teach? Do you feel it hindered your ability to teach?

3. Where, if at all, did you excel in effectively teaching while using this approach?

4. Where, if at all, did you struggle in effectively teaching while using this approach?

5. What skills do you think are necessary in order to successfully use this approach?

6. How are these skills different than standard skills you feel are required to be an effective teacher?
Appendix F: Questionnaires for Agile Workshop

This appendix contains the questionnaires used to evaluate the agile workshop. There are four questionnaires with overlapping questions to dissuade students from copying answers from their teammates. Each questionnaire is presented on a single sheet, as follows.
This is a brief questionnaire on the Agile Development class you just completed. The purpose of this questionnaire is to help us improve our teaching methods. This will not count toward your grade. All answers are anonymous.

1. Using Legos to teach Agile programming concepts is an effective method for teaching these concepts compared to a lecture presentation. (Circle the number of your answer.) Disagree | Agree
---|---
1 | 2 | 3 | 4 | 5

2. Legos made learning this subject more interesting. (Circle the number of your answer.) Disagree | Agree
---|---
1 | 2 | 3 | 4 | 5

3. What is a sprint? (Mark the box corresponding to your answer.)

   [ ] A race between developers for the most lines of code developed in an iteration.  
   [ ] A short development period of a week or two.  
   [ ] Critical items that need fixing.  
   [ ] Is the sub-unit of an iteration.

4. Knowledge about Agile development is difficult to transfer through speech or written communications so using exercises like this one helps in learning the concepts. This type of knowledge is called what? (Mark the box corresponding to your answer.)

   [ ] Abstract knowledge  
   [ ] Obscure knowledge  
   [ ] Obtuse knowledge  
   [ ] Explicit knowledge  
   [ ] Tacit knowledge

5. In software development what is most important to the customer? (Mark the box corresponding to your answer.)

   [ ] How hard a story is to implement.  
   [ ] The Business Value a story returns.  
   [ ] The number of lines of code completed per developer.

6. Agile development methodology is what type of approach to development. (Mark the box corresponding to your answer.)

   [ ] Sense and Respond  
   [ ] Rollover and regroup  
   [ ] Continuous revision model  
   [ ] Management by crisis  
   [ ] Management by plan

7. Project velocity refers to … (Mark the box corresponding to your answer.)

   [ ] How fast the project is getting done and is calculated by summing the number of stories completed divided by the number of sprints.  
   [ ] The change in the number of stories a team get finished over time  
   [ ] Is the sum of the estimates of the stories completed in the sprint  
   [ ] Is the change in the sum of the business value completed sprint to sprint.

8. A lecture would have been more effective in teaching the Agile concepts. (Circle the number of your answer.) Disagree | Agree
---|---
1 | 2 | 3 | 4 | 5

9. May we use your anonymous responses to this questionnaire for research purposes? (Mark the box corresponding to your answer.)

   [ ] Yes you may use my responses.  
   [ ] No I do not wish my answers to be used.

For questions regarding this research contact the Office of Responsible Research 800-678-6251 or the Principal Investigator: Rajiv Ramnath 614-330-7617 or by email at ramnath@cse.ohio-state.edu

A
This is a brief questionnaire on the Agile Development class you just completed. The purpose of this questionnaire is to help us improve our teaching methods. This will not count toward your grade. All answers are anonymous.

1. Using Legos to teach Agile programming concepts is an effective method for teaching these concepts compared to a lecture presentation. (Circle the number of your answer.)

   Disagree 1 2 3 4 5
   Agree

2. Legos made learning this subject more interesting. (Circle the number of your answer.)

   Disagree 1 2 3 4 5
   Agree

3. When we estimate a story as a 2 it means? (Mark the box corresponding to your answer.)

   [ ] It takes twice as long to do as a 1.
   [ ] It is twice as significant as a 1.
   [ ] It provides moderate business value.

4. Knowledge about Agile development is difficult to transfer through speech or written communications so using exercises like this one helps in learning the concepts. This type of knowledge is called what? (Mark the box corresponding to your answer.)

   [ ] Abstract knowledge
   [ ] Obscure knowledge
   [ ] Obtuse knowledge
   [ ] Explicit knowledge
   [ ] Tacit knowledge

5. Co-location provides what benefit? (Mark the box corresponding to your answer.)

   [ ] Improved communications.
   [ ] Customer buy in.
   [ ] Team satisfaction.
   [ ] Shorter commute time.
   [ ] Easier monitoring by the supervisor.

6. Agile development methodology is what type of approach to development. (Mark the box corresponding to your answer.)

   [ ] Sense and Respond
   [ ] Rollover and regroup
   [ ] Continuous revision model
   [ ] Management by crisis
   [ ] Management by plan

7. Project velocity refers to … (Mark the box corresponding to your answer.)

   [ ] How fast the project is getting done and is calculated by summing the number of stories completed divided by the number of sprints.
   [ ] The change in the number of stories a team get finished over time
   [ ] Is the sum of the estimates of the stories completed in the sprint
   [ ] Is the change in the sum of the business value completed sprint to sprint.

8. A lecture would have been more effective in teaching the Agile concepts. (Circle the number of your answer.)

   Disagree 1 2 3 4 5
   Agree

9. May we use your anonymous responses to this questionnaire for research purposes? (Mark the box corresponding to your answer.)

   [ ] Yes you may use my responses.
   [ ] No I do not wish my answers to be used.

For questions regarding this research contact the Office of Responsible Research 800-678-6251 or the Principal Investigator: Rajiv Ramnath 614-330-7617 or by email at ramnath@cse.ohio-state.edu

B

103
This is a brief questionnaire on the Agile Development class you just completed. The purpose of this questionnaire is to help us improve our teaching methods. This will not count toward your grade. All answers are anonymous.

1. Using Legos to teach Agile programming concepts is an effective method for teaching these concepts compared to a lecture presentation. (Circle the number of your answer.)
   Disagree: 1, 2, 3, 4, Agree: 5

2. Legos made learning this subject more interesting. (Circle the number of your answer.)
   Disagree: 1, 2, 3, 4, Agree: 5

3. Knowledge about Agile development is difficult to transfer through speech or written communications so using exercises like this one helps in learning the concepts. This type of knowledge is called what? (Mark the box corresponding to your answer.)
   Abstract knowledge: [ ], Obscure knowledge: [ ], Obtuse knowledge: [ ], Explicit knowledge: [ ], Tacit knowledge: [ ]

4. In software development what is most important to the customer? (Mark the box corresponding to your answer.)
   How hard a story is to implement: [ ], The Business Value a story returns: [ ], The number of lines of code completed per developer: [ ]

5. Co-location provides what benefit? (Mark the box corresponding to your answer.)
   Improved communications: [ ], Customer buy in: [ ], Team satisfaction: [ ], Shorter commute time: [ ], Easier monitoring by the supervisor: [ ]

6. Agile development methodology is what type of approach to development? (Mark the box corresponding to your answer.)
   Sense and Respond: [ ], Roll over and regroup: [ ], Continuous revision model: [ ], Management by crisis: [ ], Management by plan: [ ]

7. Project velocity refers to … (Mark the box corresponding to your answer.)
   How fast the project is getting done and is calculated by summing the number of stories completed divided by the number of sprints: [ ], The change in the number of stories a team get finished over time: [ ], Is the sum of the estimates of the stories completed in the sprint: [ ], Is the change in the sum of the business value completed sprint to sprint: [ ]

8. A lecture would have been more effective in teaching the Agile concepts. (Circle the number of your answer.)
   Disagree: 1, 2, 3, 4, Agree: 5

9. May we use your anonymous responses to this questionnaire for research purposes? [Yes you may use my responses. No I do not wish my answers to be used.]

For questions regarding this research contact the Office of Responsible Research 800-678-6251 or the Principal Investigator: Rajiv Ramnath 614-330-7617 or by email at ramnath@cse.ohio-state.edu

C
This is a brief questionnaire on the Agile Development class you just completed. The purpose of this questionnaire is to help us improve our teaching methods. This will not count toward your grade. All answers are anonymous.

1. Using Legos to teach Agile programming concepts is an effective method for teaching these concepts compared to a lecture presentation.
   (Circle the number of your answer.)

   Disagree  2  3  4  5

2. Legos made learning this subject more interesting.
   (Circle the number of your answer.)

   Disagree  2  3  4  5

3. What is a sprint?
   (Mark the box corresponding to your answer.)

   [ ] A race between developers for the most lines of code developed in an iteration.
   [ ] A short development period of a week or two.
   [ ] Critical items that need fixing.
   [ ] Is the sub-unit of an iteration.

4. When we estimate a story as a 2 it means?
   (Mark the box corresponding to your answer.)

   [ ] It takes twice as long to do as a 1.
   [ ] It is twice as significant as a 1.
   [ ] It provides moderate business value.

5. Knowledge about Agile development is difficult to transfer through speech or written communications so using exercises like this one helps in learning the concepts. This type of knowledge is called what?
   (Mark the box corresponding to your answer.)

   [ ] Abstract knowledge
   [ ] Obscure knowledge
   [ ] Obtuse knowledge
   [ ] Explicit knowledge
   [ ] Tacit knowledge

6. Agile development methodology is what type of approach to development.
   (Mark the box corresponding to your answer.)

   [ ] Sense and Respond
   [ ] Rollover and regroup
   [ ] Continuous revision model
   [ ] Management by crisis
   [ ] Management by plan

7. Project velocity refers to …
   (Mark the box corresponding to your answer.)

   [ ] How fast the project is getting done and is calculated by summing the number of stories completed divided by the number of sprints.
   [ ] The change in the number of stories a team get finished over time.
   [ ] Is the sum of the estimates of the stories completed in the sprint.
   [ ] Is the change in the sum of the business value completed sprint to sprint.

8. A lecture would have been more effective in teaching the Agile concepts.
   (Circle the number of your answer.)

   Disagree  2  3  4  5

9. May we use your anonymous responses to this questionnaire for research purposes?
   [ ] Yes you may use my responses.
   [ ] No I do not wish my answers to be used.

For questions regarding this research contact the Office of Responsible Research 800-678-6251 or the Principal Investigator: Rajiv Ramnath 614-330-7617 or by email at ramnath@cse.ohio-state.edu