# Experiences Using a Cyber Learning Environment in CS1 Classrooms

**Dr. Otto Borchert, Missouri Southern State University**

Dr. Borchert received his Ph.D. in Computer Science from North Dakota State University in 2015 and has been teaching introductory computer science courses for four years (at Gonzaga, North Dakota State University, and Missouri Southern State University). His research interests include immersive virtual environments for education, mobile app development, and STEM education through technology. He is currently an assistant professor in the Computer Information Science department at Missouri Southern State University.

**Abigail Byram, Virginia Commonwealth University**

Abigail Byram is an undergraduate senior at VCU interested in improving Computer Science education, especially for underrepresented groups.

**Ms. Debra Mardell Duke, Virginia Commonwealth University**

Debra Duke is an Instructor and the Undergraduate Director in the Computer Science Department of the College of Engineering at Virginia Commonwealth University.Debra received her Master's degree from Virginia Commonwealth University in 2006. She worked as a Software Developer at mDimension Technology making use of her experience in several programming languages, database design and development, software engineering, and web application development. She began teaching computer science at Reynolds Community College in Richmond, Virginia in 2009 and moved to VCU in August 2016. Debra has served on the advisory board for Lighthouse for Computer Science (LH4CS). The goal of the Lighthouse project is to improve computer science diversity through faculty professional development. In addition, she is a member of the Advisory Council for the Deep Run High School's Center for Information Technology in Glen Allen, Virginia, where she provides program support and assists in curriculum development for their technology-based preparatory program for future computer scientists.

**Mr. Alex David Radermacher, North Dakota State University**

Alex Radermacher is a lecturer at North Dakota State University. He teaches introductory programming courses as well as upper-level software engineering courses including the department's Senior Capstone Design course. His research interests include student learning, software testing, and software development processes.

**Mrs. Mourya Reddy Narasareddygari, North Dakota State University**

Mourya Reddy is a Phd Candidate in Software Engineering at North Dakota State University. Her present research work is related to Computer Science and Software Engineering education. Her research is focused on experiences at using a wide array of learning engagement strategies (e.g., Gamification, Social Interaction . . . ) in the context of a Cyber learning platform.

**Dr. Gursimran Singh Walia, North Dakota State University**

Gursimran S. Walia is an associate professor of Computer Science at North Dakota State University. His main research interests include empirical software engineering, software engineering education, human factors in software engineering, and software quality. He is a member of the IEEE Computer Society. Contact him at gursimran.walia@ndsu.edu

# Experiences Using a Cyber Learning Environment in CS1 Classrooms

## Abstract

The Software Engineering and Programming Cyber Learning Environment (SEP-CyLE) is a web-based platform to supplement standard course materials in CS1, CS2, software engineering, and software testing courses. In SEP-CyLE, students complete learning objects (LOs) and tutorials that cover topics which are not always discussed directly in lecture or in labs. SEP-CyLE also includes a series of learning and engagement strategies to encourage and instruct students. This experience report provides a brief background on the platform as well as information on how SEP-CyLE can be used by students and faculty. The paper continues by relaying specific successes and failures of using SEP-CyLE in CS1 courses at two separate research universities. The paper concludes with a discussion of the process for developing new learning objects for SEP-CyLE and their use in CS1 courses with an eye towards future modules and work to be done.

## 1. Introduction

This paper describes an implementation of the Software Engineering and Programming Cyber Learning Environment (SEP-CyLE) platform in two American universities, identifying best practices and potential pitfalls for others who wish to use it in their classroom. SEP-CyLE is a virtual learning environment that combines learning and engagement strategies with learning objects to create an online web-based platform for computer science education. One specific strength of the platform is its ability to provide bite-sized portions of material for students to consume, specifically with respect to testing in introductory computer programming courses.

Due to the ubiquitous nature of software in the 21st century and the increasing drive towards automation, computer science and software engineering workers are in high demand. This demand and outpouring of opportunities attracts a variety of students from different backgrounds to postsecondary computing programs. However, recent research [1] indicate that there are several knowledge deficiencies in the learning outcomes of students who graduate from computer science and software engineering courses. Many students don't meet industry expectations for technical proficiency. This outcome can be traced to a lack of comprehension of principal programming ideas leading to the creation of poorly developed software. In addition to this, universities are striving to improve their first year retention rates. To help overcome these problems, it is crucial that computer science instructors make use of pedagogical strategies that help students improve their understanding of programming concepts and become better programmers. One such tool that implements these strategies through a cyber-learning platform that incorporates an array of learning engagement strategies (e.g., collaboration, social networking, gamification), is SEP-CyLE.

SEP-CyLE attempts to overcome these technical and retention issues in three main ways. First, it breaks large concepts like software programming and testing concepts into smaller learning objects providing a less overwhelming experience for students. Second, integrated learning and engagement strategies show that software testing and other foundational programming concepts are relevant. Finally, as a standalone virtual environment SEP-CyLE provides a way for students to learn about software programming and testing concepts separate from the classroom environment allowing the instructor to focus on other material.

The remainder of the paper is organized as follows. We begin by presenting additional information related to SEP-CyLE and a brief background on related projects. We then describe the experience of using SEP-CyLE in CS1 classrooms at two different research universities. Next, we provide a discussion related to developing content for SEP-CyLE. We conclude with a discussion and description of future research.

## 2. SEP-CyLE – Software Engineering and Programming Cyber Learning Environment

SEP-CyLE was initially designed and developed by Clarke et al. in 2010 as WRESTT - A Web Repository of Testing Tutorials with a cyberlearning environment [2]. The main objective of WRESTT was to improve the testing knowledge of computer science and software engineering students with the help of digital learning content that included learning objects (LOs) and testing tool tutorials. Initially, WRESTT was developed in two versions. The first version contained primarily tutorials for testing tools and frameworks such as JUnit – a unit testing framework [3], SWAT – Simple Web Automation Toolkit [4], and Cobertura – a code coverage tool [5]. Eventually, a second version of WRESTT was developed to support collaborative learning, allowing students to work in virtual teams and earn virtual reward points. The reward points (called virtual points in WRESTT) were not necessarily part of a student's course grade but could serve as motivation for the students to interact with the system beyond what was required by their course instructor. This second version was also different from the first in that it included tutorials and quizzes on the core concepts of software testing. It is this second version of WRESTT (which has since been renamed as SEP-CyLE - Software Engineering and Programming Cyber Learning Environment) that is the focus of this paper. SEP-CyLE continues to undergo development and has become a configurable cyberlearning environment that contains new and growing digital learning content related to software programming and testing concepts. A diagram of these features is shown in Figure 1, while a screenshot of the current instructor interface is given in Figure 2.

The current version of SEP-CyLE offers an assortment of learning and engagement strategies. These strategies were developed in order to increase student motivation and help them learn content through a variety of experiences. So far, three different strategies have been implemented including collaborative learning, gamification, and social interaction.

**Collaborative features:** These features allow students to be arranged into small teams. Once all members of a team complete a learning object, they are given bonus points based how fast they complete the learning object. Higher points are awarded to teams that finish first. As we discuss below, this is not true collaborative learning, but at least motivates students to apply individual effort to the task.

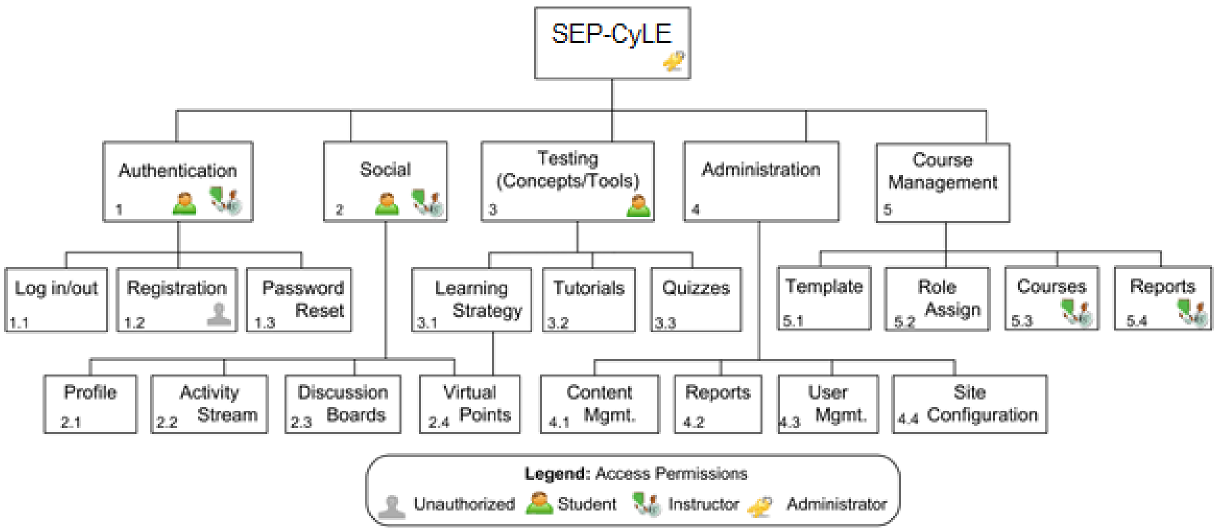


Figure 1: Block Diagram of SEP-CyLE Features

**Gamification Features:** SEP-CyLE incorporates gamification elements to improve student engagement and motivation through playful and context-centric activities. The gamification elements that are used in SEP-CyLE are virtual (reward) points and leaderboards.

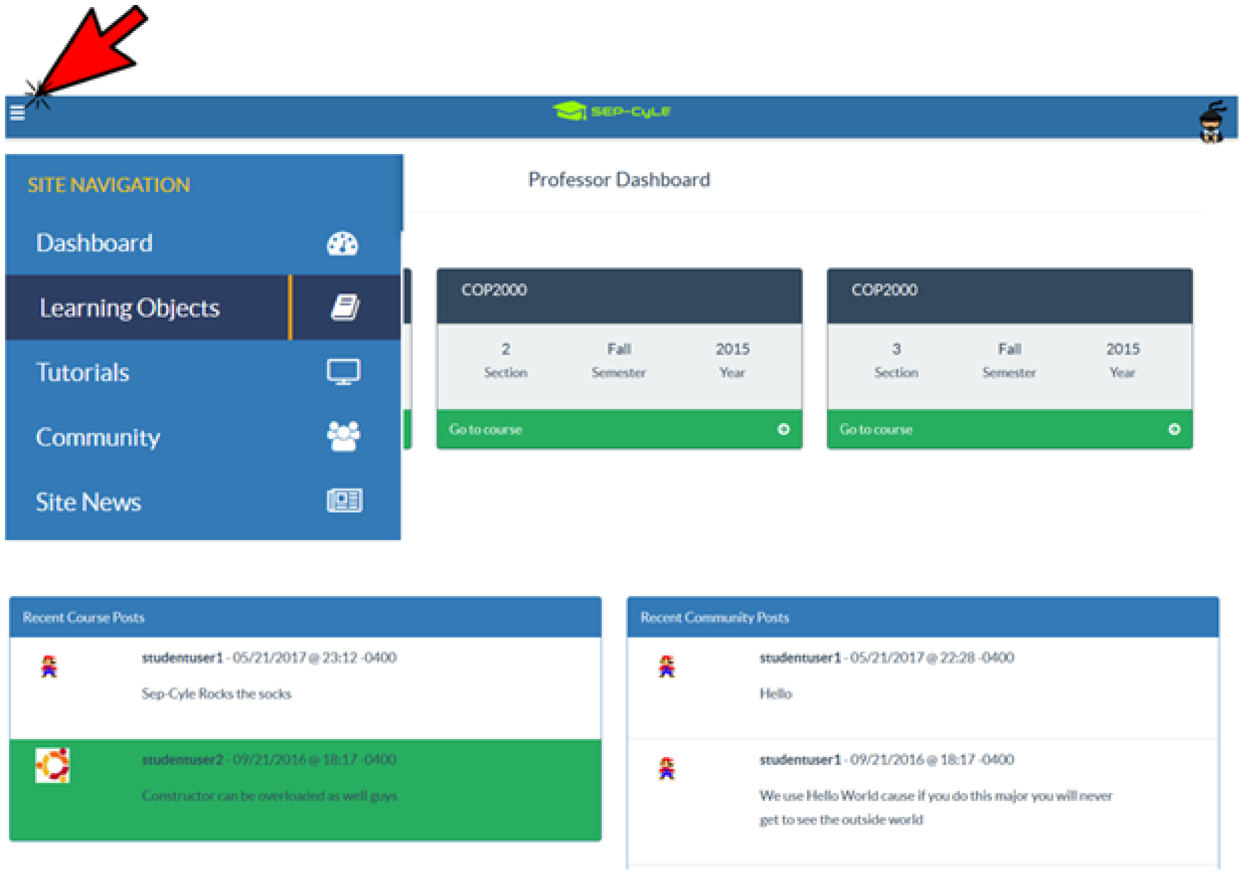
Students earn virtual points by completing certain tasks individually or as part of team. These tasks include completing LOs or engaging in some social activities in SEP-CyLE such as writing on discussion boards or leaving reviews and comments on LOs.

Leaderboards provide details about virtual points earned by other students in the course. This helps students know where they are standing in the class and increases engagement by encouraging the student to earn more points to place higher on the leaderboard.

**Social Features:** These features allow students to modify their profile (e.g. add a picture or modify other details) and to engage in conversations on course discussion boards as well as a discussion forum for the entire SEP-CyLE community. For all these activities, each student can be awarded virtual points, depending on the learning and engagements strategies enabled by the instructor.

In addition to these learning and engagement strategies, there are two fundamental features of SEP-CyLE: Learning Objects and Tutorials.

Learning Objects (LO): The main educational aspect of SEP-CyLE is the Learning Object, which are small content modules designed to be completed in no more than fifteen minutes. An LO can be described as a small self-contained chunk of knowledge, which is created and vetted by experts in that area. Each LO consists of two main sections: content and assessment. The content section includes information about the topic to be explained and is represented in various forms including text, audio, and video. The assessment section consists of a practice quiz that students can take to gauge their understanding of the material as well as an official quiz that is graded and reported by the system. At the time of this publication SEP-CyLE contains more than 45 LOs.



studentuser1 - 05/21/2017 @ 23:12 -0400

Sep-Cyle Rocks the socks

studentuser2 - 09/21/2016 @ 18:17 -0400

Constructor can be overloaded as well guys

studentuser1 - 05/21/2017 @ 22:28 -0400

Hello

studentuser1 - 09/21/2016 @ 18:17 -0400

We use Hello World cause if you do this major you will never get to see the outside world

Figure 2: Instructor View. Courses the instructor is teaching is on the top row. Discussion boards are shown on the bottom. Options for site navigation are shown in the upper left.

Tutorials: Tutorials are videos which provide students with step by step instructions for performing a task or using a tool. These tutorials do not contain assessments like LOs and are meant to be a guide for tools and techniques, rather than a way of testing student understanding. Table 1 lists tutorials that are present in SEP-CyLE as of this writing.

## 3. Related Work

This paper focuses on the use of SEP-CyLE as a method of integrating software testing into the CS1 curriculum at two universities in the United States. Before developing the case study and expounding on the lessons learned, it will be useful to place SEP-CyLE in the software engineering education landscape and to compare it to other learning technologies.

A recent literature review [6] shows mixed results with respect to applying test driven development and teaching software testing techniques in the first programming course. They specifically note, citing other researchers, that course-level software testing requirements can be overwhelming for students [7], perceived by students as irrelevant [8], and takes time away from other material [9]. SEP-CyLE attempts to overcome these difficulties in three main ways. First, breaking large concepts like software programming and testing into smaller learning objects provides a less overwhelming experience for students. Second, learning and engagement strategies demonstrate that software testing is relevant. Finally, as an online virtual environment SEP-CyLE provides a way for students to learn about software testing separate from the classroom environment allowing the instructor to focus on other material.

Table 1. Tutorials in SEP-CyLE

| **Tutorial Name** | **What the student learns** |
|---|---|
| Selenium | How to automate web applications for testing purposes |
| JUnit | How to write and run repeatable tests for Java |
| Code Cover | How to perform automated code coverage tests |
| EclEmma | How to analyze JUnit tests from within the Eclipse environment |
| SWAT | How to interact with several different web browsers |
| Rational Functional Tester | How to automate flexible tests for frequent application user interface changes |
| Cobertura | How to identify Java code that is lacking test coverage |
| Robotium | How to write automatic black-box UI tests for Android applications |
| Cobertura for Web Apps | How to use Cobertura to generate code coverage reports for a web application |
| Mocking for Java with Mockito | How to use Mockito, a framework to create stubs and spies |

Another strength of SEP-CyLE is that it is a multi-institutional tool - SEP-CyLE is used at 49 institutions of higher learning across the globe. Instructors from research universities, regional teaching colleges, community colleges, and both public and private institutions have all either attended a workshop or used SEP-CyLE in their classroom. Further exploration and evaluation of learning environments in a variety of situations is critical to expanding their reach and effectiveness [6]. This paper's contribution lies in relating two case studies: one at North Dakota State University and one at Virginia Commonwealth University, showing lessons learned while implementing the software. This information will assist instructors in adopting SEP-CyLE at their own institutions, increasing dissemination of the platform.

Commercial learning management systems (LMS) occupy a different niche from SEP-CyLE. Programs like Blackboard, Canvas, and Desire2Learn offer university-wide support for instructional classroom management. SEP-CyLE focuses on deploying reusable learning objects and learning and engagement strategies to teach software engineering skills and techniques to students. However, since LMS's are an integral part of most university communities, SEP-CyLE does support the export of student scores to a CSV file, so that instructors can load that information into an organization-wide LMS. Building Learning Tools Interoperability (LTI – for integration between SEP-CyLE and organization-wide LMS's) is in progress.

Other large efforts have been undertaken to develop entire courses for software testing and software engineering. Websites like KhanAcademy and Codecademy provide complete programming tutorials. Massively Open Online Course websites like Udacity and Coursera provide complete course reproductions. Personalized and real-time online learning environments for computer science education have also been designed for programming and have had some success [10, 11, 12, 13]. SEP-CyLE instead focuses on improving a typical university face-to-face lecture, with small, bite-sized learning objects, not a full replacement for course materials. Unlike most of the personalized learning environments, it has been used in a wide variety of universities around the world.

## 4. Implementation – Virginia Commonwealth University

SEP-CyLE was integrated into four sections of CS1 at Virginia Commonwealth University over the course of an academic year, with an average enrollment of 50 students per section. Roughly two thirds of students were Computer Science majors, most of the remaining students were Mathematics, Bioinformatics or other STEM majors, with a small number of Social Sciences and Humanities majors in each section. The results of a survey conducted at the beginning of each section indicated that about half of the students had at least some prior programming experience obtained outside of the course.

SEP-CyLE was introduced in the second week of the course by providing a demonstration of the website including how to log in, how to access and complete LOs, and how to leave comments on LOs. Engagement strategies in SEP-CyLE were also introduced including the discussion board, the point system, and student teams. Students were encouraged to participate for extra practice and to broaden their knowledge of computing beyond topics covered in class.

In two of the four sections, weekly participation in SEP-CyLE LOs was an optional extra credit opportunity. Each week, the student with the most accumulated points in SEP-CyLE was announced to the class and awarded a small prize in the form of candy and laptop stickers prior to a brief introduction of the following week's LO. In the other two sections, the weekly completion of the assigned LO was a required portion of the course's homework grade. An introduction to the following week's LO was also presented weekly, but without announcing a top-scoring student. To collect comparative data, one section was given SEP-CyLE with all features (gamification, social interaction, and collaborative learning) enabled. Another section was given access to the website with all of these features disabled. In the other two sections, only gamification and only social interaction were enabled, respectively.

The sequence of LOs assigned throughout these sections were: Hand Tracing Sequential Code, Pair Programming, Statement Coverage, Hand Tracing Method Calls, Debugging, Programming Coding Standards, Introduction to Software Testing - 1, 2, and 3, Introduction to UML, and CS1 Unit Testing-1. This sequence was designed to align the LO with the material covered in class at the time. About half of these LOs were directly related to course material, so SEP-CyLE integrated well into the curriculum. In future semesters, we plan to develop more LOs to match this university's CS1 curriculum. The concepts in some LOs, for example, software testing, more closely follow this university's CS2 curriculum. Considering this, SEP-CyLE will also be introduced as a supplementary resource into CS2 to fit course curriculum and offer a review on CS1 concepts.

### 4.1 What Worked and What Didn't Work

All study participants were invited to discuss their experience using SEP-CyLE and 32 students attended the focus group sessions. The participants indicated that the social and collaborative features of SEP-CyLE did not particularly encourage use of the website. Students indicated that if they did not already know other team members assigned to their group within SEP-CyLE, they were often uninterested in reaching out to engage with them. Instead students said that discussions and teamwork on SEP-CyLE would be more engaging if relationships were formed in person before collaborating on the website or if they were allowed to self-select their team. Students stated that the gamification point system encouraged continued usage, especially when these points were integrated into the course either for a grade or for competition. Focus group responses and student outcomes suggest that enabling the gamification feature both encouraged higher participation in SEP-CyLE and lead to more positive student outcomes in the course. Because of this, we suggest utilizing the gamification feature when integrating SEP-CyLE into a course; especially when online points are linked with real-world rewards such as extra credit or recognition.

Students in the focus groups self-reported that they were not particularly interested in using SEP-CyLE initially, when it was only listed as an external resource with bonus points given for completing LOs. One focus group participant commented, “I did all the extra credit at the end, before the final”. According to focus group participants, students felt that they retained more information from LOs that were directly related to material currently discussed in lectures. In particular, they liked “having examples of what we’re learning” and an opportunity to “get extra practice and have a better understanding” of course topics. Some focus group participants indicated a lack of retention for the content of LOs in topics which were not directly related to course materials. One participant commented that she felt the “lack of baseline knowledge on the topic” hindered the instructional value of LOs used to supplement the course materials. Considering this, it is suggested to integrate the website into the course by granting credit and/or assigning LOs as the corresponding topics are covered in lectures.

## 5. Implementation – North Dakota State University

SEP-CyLE was used at North Dakota State University in four sections of the CS1 introductory programming course with an approximate enrollment of 40 per section. These students were primarily CS majors in their freshman year, but the course also includes a sizeable portion of students majoring in Engineering, Mathematics, Statistics, and other STEM disciplines who tend to be sophomores or juniors just as often as they are freshmen. Based on previous surveys of the student population in these classes, approximately half have no prior exposure to programming at all and only a very tiny minority have taken a high school programming class or equivalent. There were a few steps that had to be completed before we were able to use SEP-CyLE. First, we needed to contact the administrators for a new account. No option currently exists to self-create your own account or your own classes. Once the four sections were created by the SEP-CyLE system administrator, we needed to enroll our students. This can be done one student at a time on a web-based form or by submitting a specially formatted CSV file. SEP-CyLE does not integrate with any standard learning management system, which would make it easier to import students into the course.

We chose to experiment with the learning and engagement strategies described earlier during our time using SEP-CyLE. The different combinations of features included (1) using all strategies, (2) using all strategies except gamification, (3) using all strategies except social interaction, and (4) using all strategies except collaboration. Each section of CS1 was configured to use a different set of learning and engagement strategy. Students were placed into teams of three or four within their own section using the built-in team building functionality of SEP-CyLE. This functionality allows students to be placed into teams manually or randomly. We chose to randomly assign students into groups.

Students were introduced to SEP-CyLE during the first class period of the semester. They first started by completing a short multiple-choice pre-test that was used by researchers to establish a baseline for the students' abilities. A computer science graduate research assistant gave a short demonstration of the SEP-CyLE website, while students in the course followed along on lab computers. Students were provided with instructions on how to log in, navigate to assigned learning objects, view relevant tutorials, and how to communicate using discussion boards. Students were then given the rest of the period to work on the first learning object "Introduction to NetBeans".

LOs were assigned approximately every week throughout the semester, although there were no LOs assigned in the final weeks of the semester. Students were reminded of the current learning object assignments verbally during lecture and via announcements added to Blackboard, our university-wide learning management system. The progression of learning objects used throughout the semester was: Introduction to NetBeans IDE, Introduction to Version Control, Advanced NetBeans IDE Features, Fixing Program Errors, Creating Unit Tests, NetBeans GUI Builder, Debugging, White Box Testing, Advanced Debugging, and SOLID Design Principles. Most of these LOs provided supplemental information related to topics discussed as part of the course, but that were not present in the course text book or otherwise available to students who missed demonstrations or discussion during lecture periods.

SEP-CyLE provided more information for students who were struggling with the testing techniques and concepts on homework assignments, as only a minimal amount of lecture time is devoted to testing topics. This progression also matched the course assignments well, although we ended up having to implement some new learning objects as discussed below. Students were offered extra credit for their participation in SEP-CyLE, but it was not required.

At the conclusion of the course, students completed a final post-test given as an attachment to the final exam. Results of these studies comparing the learning and engagement strategies are discussed in [14,15].

## 5.1 What Worked and What Didn't Work

Approximately two-thirds of the way through the course, students were offered extra credit to meet with an external evaluator in a focus group. Students told the evaluator that while they enjoyed using SEP-CyLE, they didn't realize they were in teams with other students. Similarly, the other collaborative features of the platform such as the discussion boards went almost entirely unused. Students who used SEP-CyLE found it to be useful, especially for the testing components of homework assignments. Anecdotally, students would tell other students about SEP-CyLE after finding a useful piece of information in the environment which leads us to believe it can be of assistance in the classroom.

Although SEP-CyLE is quite useful for students and has a wealth of content already available, instructor-level functionality is less well developed. SEP-CyLE does not contain an ability to integrate with Blackboard, Moodle, Canvas, or other learning management systems (LMS). It does provide functionality to export grades and other student data to an Excel spreadsheet, but does not do so in a format that directly imports into an LMS.

It was useful being able to direct students to a resource focused on software testing when they were having issues implementing testing techniques and strategies in their homework assignments. Additionally, SEP-CyLE provided a good deal of information related to the different software tools (NetBeans, SVN, etc.) used as a part of the course that are not discussed in the textbook. As SEP-CyLE content matures, we may consider making SEP-CyLE a graded portion of the course.

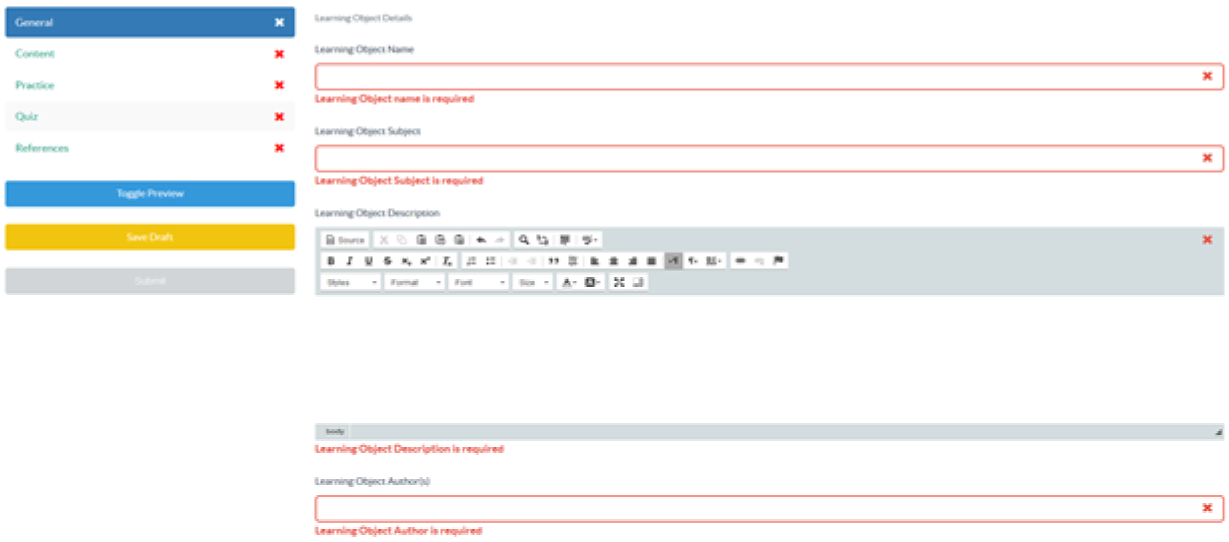
## 6. Developing Learning Objects

Although SEP-CyLE contains a large variety of existing LOs, one of the largest draws for using SEP-CyLE was the ability for the course instructors to create their own modules to supplement the existing course syllabus and structure, rather than attempting to shoehorn in existing modules. This worked well as it ensured that every week, students were viewing a learning object that was highly relevant to the course material and allowed SEP-CyLE to serve as a supplement for topics where the course instructors felt the textbook lacked an appropriate amount of depth. The Learning Objects we developed include:

- SOLID Principles - This learning object introduces the design principles that help in making object-oriented software more understandable, flexible and maintainable.
- Advanced Debugging - This Learning Object provides information about three features in the NetBeans IDE, including: the “Step Into” expression, watches, and conditional breakpoints.
- Fixing Program Errors in NetBeans - This learning object focuses on identifying and fixing different types of programming errors when using the NetBeans IDE.

To develop a learning object, an instructor must create content related to a specific topic, a practice quiz that students can repeatedly complete, a real quiz used for grading purposes, and references where appropriate. Content is shown through a series of web pages that can include text, images, and videos. Quizzes allow for different types of questions including true-false, multiple choice and multiple answer formats.

Instructors can enter content via an online web editor with WYSIWYG features (Figure 3). The instructor is encouraged to create multiple small pages so that students aren’t overwhelmed with the amount of content. Video and images are inserted using buttons on the editor, while text is simply typed into the interface. Instructors use the same tools to create a series of true-false, multiple choice, and multiple answer question for the quizzes. Instructors can save a draft of the work for return at a later time. They are also able to switch to a preview mode, so they can see what their learning object would look like to a student viewing the module. These features make it painless to add new content, which is a definite advantage of using SEP-CyLE compared to other static learning environments.



Learning Object Details

Learning Object Name

Learning Object name is required

Learning Object Subject

Learning Object Subject is required

Learning Object Description

Learning Object Description is required

Learning Object Author(s)

Learning Object Author is required

Figure 3. Editing a new LO in SEP-CyLE

## 7. Conclusion and Future Work

We will continue to use SEP-CyLE in our classrooms and develop learning objects for our and other students. Some planned modules include: mock classes, GUI design, and events and listeners. Many of the SEP-CyLE tutorials and LOs focus on the Java programming language and the NetBeans IDE. This works well for both institutions in this experience report, but we plan to develop learning objects in other languages and development environments as we find more collaborators.

Another issue with the current version of SEP-CyLE is that students who are using the collaborative learning engagement strategy aren't actually collaborating. They are completing the same problems individually and their group score reflects how many quiz questions each person gets right. One enhancement might be creating a system that would require them to work collaboratively to solve a more complicated problem or to engage in other activities such as reviewing each other's code.

Another problem is the lack of an integrated IDE within SEP-CyLE. We would like to see the ability to have students work on small code problems (or eventually entire programming assignments) in SEP-CyLE where the system automatically checks student's solutions. Based on the results of the automated evaluation, learning objects can be assigned to students who are having difficulties. The overarching idea is to create a continuous practice, assessment, and feedback loop that would help students throughout the hands-on learning process.

The developers of SEP-CyLE are also working on a version of the cyberlearning environment for other disciplines. It is called STEM-CyLE, a configurable learning and engagement cyberlearning environment that contains digital learning content in all areas of STEM. It combines all of the features of SEP-CyLE described in this paper with other STEM disciplines.

We continue to look for more participants to use SEP-CyLE and to develop content for the platform. Workshops are given approximately annually to teach instructors how to use it in their classrooms. At these workshops, instructors learn about the research behind SEP-CyLE, brainstorm ideas on how to integrate SEP-CyLE into their classrooms and complete tutorials on how to use both the student and instructor side of SEP-CyLE. To learn more about SEP-CyLE and the workshops, visit https://stem-cyle.cis.fiu.edu.

**Acknowledgements**

This work is supported in part by the National Science Foundation under grants DUE-1225742, DUE-1525112, and DUE-1525414. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

**References**

[1] A. Radermacher, G.S. Walia, and D. Knudson. *Investigating the skill gap between graduating students and industry expectations*. ICSE Companion, 2014.

[2] P. J. Clarke, A. A. Allen, T. M. King, E. L. Jones, and P. Natesan. *Using a Web-based Repository to Integrate Testing Tools into Programming Courses.* In Proceedings of the ACM International Conference Companion on Object Oriented Programming Systems Languages and Applications Companion (OOPSLA '10). ACM, New York, NY, USA, pp. 193–200, 2010.

[3] E. Gamma and K. Beck. JUnit, 2008. http://www.junit.org/.

[4] Ultimate Software. SWAT, 2009. http://sourceforge.net/projects/ulti-swat/.

[5] Cobertura Team. Cobertura, May 2010. http://cobertura.sourceforge.net/.

[6] A. Luxton-Reilly, I. Albluwi, B. Becker, M. Giannakos & A. Kumar, L.M. Ott, J. Paterson, M. Scott, J. Sheard, and C. Szabo. *Introductory Programming: A Systematic Literature Review.* Proceedings Companion of the 23rd Annual ACM Conference on Innovation and Technology in Computer Science Education, July 02 - 04, 2018, Larnaca, Cyprus, pp 55-106, 2018

[7] W. Marrero and A. Settle. *Testing first: emphasizing testing in early programming courses.* In Proceedings of the 10th Annual SIGCSE Conference on Innovation and Technology in Computer Science Education (ITiCSE'05). ACM, NewYork, NY, USA, 4–8. 2005.

[8] V. Isomöttönen and V. Lappalainen. *CSI with games and an emphasis on TDD and unit testing: piling a trend upon a trend.* ACM Inroads 3, 3 (2012), pp. 62–68, 2012.

[9] V. Thurner and A. Böttcher. *An "objects first, tests second" approach for software engineering education.* In IEEE Frontiers in Education Conference (FIE'15). IEEE, pp. 1–5, 2015.

[10] S. Choy and S. Ng, *An interactive learning environment for teaching and learning of computer programming*, IEEE International Conference on Advanced Learning Technologies, 2004. Proceedings., Joensuu, 2004, pp. 848-849.

[11] S. Chookaew, P. Panjaburee, D. Wanichsan, P. Laosinchai. *A Personalized E-Learning Environment to Promote Student's Conceptual Learning on Basic Computer Programming.* Procedia - Social and Behavioral Sciences. 116, pg. 815-819, 2014.

[12] C. Hill and B. M. Slator. *Computer Science Instruction in a Virtual World*. World Conference on Educational Media, Hypermedia and Telecommunications (ED-MEDIA 2000), June 26-July 1, Montreal, Quebec, Canada, 2000.

[13] P. Bhattacharya, M. Guo, L. Tao, Y. Fu and K. Qian, *A Collaborative Interactive Cyber-learning Platform for Anywhere Anytime Java Programming Learning*, 2011 IEEE 11th International Conference on Advanced Learning Technologies, Athens, GA, pp. 14-16, 2011.

[14] MR Narasareddygari, G. Walia, and A. Radermacher. *Using Gamification and Cyber Learning Environment to Improve Student's Learning in an Introductory Computer Programing Course —an empirical case study*, 125th Annual ASEE Conference, Salt Lake City, Utah, 2018.

[15] MR Narasareddygari, G. Walia, O. Borchert, and A. Radermacher. *Evaluating Learning Engagement Strategies in a Cyber Learning Environment during Introductory Computer Programming Courses-- An Empirical investigation,* 125th Annual ASEE Conference, Salt Lake City, Utah, 2018.