The Journal of Computing Sciences in Colleges

Papers of the 32nd Annual CCSC Rocky Mountain Conference

October 20th-21th, 2023
Metropolitan State University of Denver
Denver, CO

Baochuan Lu, Editor
Southwest Baptist University

Pam Smallwood, Regional Editor
Regis University

Volume 39, Number 2 **October 2023**

The Journal of Computing Sciences in Colleges (ISSN 1937-4771 print, 1937-4763 digital) is published at least six times per year and constitutes the refereed papers of regional conferences sponsored by the Consortium for Computing Sciences in Colleges.

Table of Contents

The Consortium for Computing Sciences in Colleges Board of Directors

Following is a listing of the contact information for the members of the Board of Directors and the Officers of the Consortium for Computing Sciences in Colleges (along with the years of expiration of their terms), as well as members serving CCSC:

Scott Sigman, President (2024), ssigman@drury.edu, Mathematics and Computer Science Department, Drury University, Springfield, MO 65802.

Karina Assiter, Vice President/President-Elect (2024), KarinaAssiter@landmark.edu, Computer Science, Landmark College, Putney, VT 05346.

Baochuan Lu, Publications Chair (2024), blu@sbuniv.edu, Division of Computing & Mathematics, Southwest Baptist University, Bolivar, MO 65613.

Brian Hare, Treasurer (2023), hareb@umkc.edu, School of Computing & Engineering, University of Missouri-Kansas City, Kansas City, MO 64110.

Cathy Bareiss, Membership Secretary (2025), cathy.bareiss@betheluniversity.edu, Department of Mathematical & Engineering Sciences, Bethel University, Mishawaka, IN 46545.

Judy Mullins, Central Plains Representative (2023), mullinsj@umkc.edu, University of Missouri-Kansas City, Kansas City, MO (retired).

Michael Flinn, Eastern Representative (2023), mflinn@frostburg.edu, Department of Computer Science & Information Technologies, Frostburg State University, Frostburg, MD 21532.

David R. Naugler, Midsouth Representative (2025), dnaugler@semo.edu, Brownsburg, IN 46112.

David Largent, Midwest Representative(2023), dllargent@bsu.edu, Department of Computer Science, Ball State University, Muncie, IN 47306.

Mark Bailey, Northeastern Representative (2025), mbailey@hamilton.edu, Computer Science Department, Hamilton College, Clinton, NY 13323.

Shereen Khoja, Northwestern Representative(2024), shereen@pacificu.edu, Computer Science, Pacific University, Forest Grove, OR 97116.

Mohamed Lotfy, Rocky Mountain Representative (2025), mohamedl@uvu.edu, Information Systems & Technology Department, College of Engineering & Technology, Utah Valley University, Orem, UT 84058.

Tina Johnson, South Central Representative (2024), tina.johnson@mwsu.edu, Department of Computer Science, Midwestern State University, Wichita Falls, TX 76308.

Kevin Treu, Southeastern Representative (2024), kevin.treu@furman.edu, Department of Computer Science, Furman University, Greenville, SC 29613.

Bryan Dixon, Southwestern Representative (2023), bcdixon@csuchico.edu, Computer Science Department, California State University Chico, Chico, CA 95929.

Serving the CCSC: These members are serving in positions as indicated:
Bin Peng, Associate Editor, bin.peng@park.edu, Department of Computer Science and Information Systems, Park University, Parkville, MO 64152.
Ed Lindoo, Associate Treasurer & UPE Liaison, elindoo@regis.edu, Anderson College of Business and Computing, Regis University, Denver, CO 80221.
George Dimitoglou, Comptroller, dimitoglou@hood.edu, Department of Computer Science, Hood College, Frederick, MD 21701.
Megan Thomas, Membership System Administrator, mthomas@cs.csustan.edu, Department of Computer Science, California State University Stanislaus, Turlock, CA 95382.
Karina Assiter, National Partners Chair, karinaassiter@landmark.edu, Landmark College, Putney, VT 05346.
Deborah Hwang, Webmaster, hwangdjh@acm.org.

CCSC National Partners

The Consortium is very happy to have the following as National Partners. If you have the opportunity please thank them for their support of computing in teaching institutions. As National Partners they are invited to participate in our regional conferences. Visit with their representatives there.

Gold Level Partner
Rephactor

Welcome to the 2023 CCSC Rocky Mountain Conference

Welcome to the 32nd annual conference of the Rocky Mountain (RM) Region of the Consortium for Computing Sciences in Colleges. The CCSC RM region board members are grateful for the authors, presenters, speakers, attendees, and students participating in this year's conference.

This year we received 14 paper submissions on a variety of topics, of which 9 papers were accepted for presentation in the conference. Multiple reviewers, using a double-blind paper review process, reviewed all submitted papers for the conference. The review process resulted in an acceptance rate of 64%. In addition to the paper presentations, there are five peer reviewed tutorials/workshops. This year is the first time we accepted scholarly posters. The review process resulted in two posters for presentation. We truly appreciate the time and effort put forth into the reviewing process by all the reviewers. Without their dedicated effort, none of this would be possible. A special thank you goes to co-Submission chair Karina Assister.

The CCSC RM region board would like to thank our national gold level partner Rephactor, as well as the Association for Computing Machinery in-cooperation with SIGCSE.

We hope you enjoy the conference and take the opportunity to interact with your colleagues and leave both enthused and motivated. As you plan your scholarly work for the coming year, we invite you to submit a paper, workshop, tutorial, or panel for a future CCSC RM region conference, or to serve as a reviewer or on the CCSC RM region board. Please encourage your colleagues and students to participate in future CCSC RM region conferences.

Mohamed Lotfy, PhD
Utah Valey University
Conference Chair

2023 CCSC Rocky Mountain Conference Steering Committee

Mohamed Lotfy
Conference chair .... Utah Valley University, UT
Ed Lindoo
Treasurer .... Regis University, CO
Pam Smallwood
Editor .... Regis University, CO
Karina Assiter
Submission Co-chair .... Landmark College, VT
Mohamed Lotfy
Submission Co-chair .... Utah Valley University, UT
Dan McDonald
Webmaster .... Utah Valley University, UT
Jenny Nehring
Publicity chair .... Utah Valley University, UT
Ed Lindoo
Registrar .... Regis University, CO
Jody Paul
Site Co-chair .... Metropolitan State University, CO
Thyago Mota
Site Co-chair .... Metropolitan State University, CO
Mohamed Lotfy
Program Chair .... Utah Valley University, UT
Michael Leverington
Student Posters Co-chair .... Northern Arizona University, AZ
Troy Taysom
Student Posters Co-chair .... Utah Valley University, UT
Kodey Crandall
Student Programming Competition Co-chair .... Utah Valley University, UT
Dave Loper
Student Programming Competition Co-chair .... Utah Valley University, UT

Regional Board — 2023 CCSC Rocky Mountain Region

Mohamed Lotfy, Board Representative .... Utah Valley University, UT
Ed Lindoo, Treasurer .... Regis University, CO
Pam Smallwood, Editor .... Regis University, CO
Dan McDonald, Webmaster .... Utah Valley University, UT

The Generation of a Large Bank of Randomized Questions in a Discrete Structures Course*

Jose Cordova
Department of Computer Science
University of Louisiana at Monroe
Monroe, LA 71209
cordova@ulm.edu

Abstract

This paper describes a system designed to generate a large bank of questions containing randomized elements for certain topics in an introductory discrete mathematics course. The resulting questions introduce random parameters into questions designed to test a student's mastery of various learning outcomes in propositional logic. The system stores randomly generated questions in an XML-based file format that can be imported into a learning management system for assignment as homework exercises and/or quiz questions. The question bank, which includes hundreds of questions of various structural types in six areas of propositional logic, has been used successfully in three sections of our discrete structures course.

1 Introduction

The effort described in this paper is part of a larger study designed to ascertain the effect of randomized vs. non-randomized homework assignments in various areas of computer science. A previous paper [2], which focused on our data structures course, reported our initial findings that there was little or no correlation between the performance of students in non-randomized homework assignments and their performance in corresponding code-completion exam questions.

The above-mentioned paper also discussed possible reasons leading to the apparent lack of impact of homework exercises on students' mastery of learning objectives, as indicated by their performance on exam questions. One possibility is that, given a uniform set of homework questions presented to a group of students, the correct answers are shared among students within the same course or possibly among students taking the course in different sections or different semesters. The natural research question to be examined is whether the introduction of randomized elements in homework questions affects the results significantly. To help with this research, the author has developed software to generate a large bank of questions in an introductory discrete structures course. This paper describes the nature of the questions generated, the general approach used by the software, and preliminary experiences of using the test bank in an actual classroom setting.

2 Background

While previous research studies have investigated the effectiveness of online homework assignments as predictors of test scores ([4], [5], [6]), little emphasis has been placed on studying the effects of homework randomization. In contrast, Chen et al [1] studied the impact of question randomization on asynchronous exams in an effort to limit collaborative cheating. Interestingly, the researchers found that the effects of collaborative cheating were significantly reduced by using even relatively small question pools consisting of three or four problems.

Commonly used Learning Management Systems (LMS) –such as Moodle and Canvas– have traditionally provided instructors with the ability to compose questions in which a numerical answer is computed from other numeric value(s) drawn randomly from a specified range when the question is presented to the student. This type of calculated formula question allows an instructor to write a question once while effectively generating multiple versions of the question. However, in computer science, there are several types of student outcomes which cannot be assessed adequately by calculated formulae using numeric values. For instance, a typical question in an introductory discrete structures course requires students to select the correct symbolic logic translation of an English sentence. The following example is taken from a popular introductory textbook in discrete structures for computer science majors [3].

Let s = "stocks are increasing" and i = "interest rates are steady." Write the statements below in symbolic form using the symbols $\neg$, $\wedge$, $\vee$ and the letters s

and i to represent component statements.

a. Stocks are increasing but interest rates are steady.

b. Neither are stocks increasing, nor are interest rates steady.

When giving electronic assessments using an LMS, an instructor could provide students with a list of choices –each containing an expression in symbolic form– including the correct answer and several distractors. However, if the intent is to assign randomized assessments, one would need to generate multiple similar questions by modifying either the component statements or the structure of the expression, or both. Such a task would require a significant amount of effort if performed manually. Furthermore, if such an effort were to be undertaken for various student learning outcomes, the amount of time required of the instructor would be prohibitive.

To help address these obstacles, we have developed software to generate arbitrarily large numbers of questions, such as the one above, in a format that can be imported into the Moodle LMS. The sections that follow describe the methodology used and the resulting structure of the generated questions.

3 Methodology

When designing a system to generate a bank of questions to be used in randomized assessments, one can include randomness in at least two ways: *parameter randomization* and *structural randomization*. In parameter randomization, the structure of the question is consistently duplicated, but different parameter values are used randomly, as is the case with the calculated formula question types discussed in the previous section. In contrast, with structural randomization, both the parameter values and the form of the question (as well as the solution) change from question to another.

Consider, for instance, a learning outcome from our introductory discrete structures course related to the book example presented in the previous section:

the student will be able to select the correct symbolic logic translation of an English sentence.

When applying parameter randomization, one can generate multiple questions of the form:

Given the statements p: <p> and q: <q>, select the correct symbolic representation for the sentence: <p> but <q>

where <p> and <q> are English statements such as "stocks are increasing" and "interest rates are steady." By replacing the values of <p> and <q> with randomly selected statements, one can programmatically generate many questions, each containing a different English sentence. Of course, in all cases, the student is simply asked to recognize and select the conjunction (p $\wedge$ q) from a list of different choices. To introduce structural randomization, one can modify the form of the question (and the corresponding correct answer) as in the following:

> *Given the statements p: <p> and q: <q>, select the correct symbolic representation for the sentence: <p> only if <q>*

If the system draws random values for p and q from a list of n distinct statements, it is possible to generate n(n-1) variations of each question form, since there are n choices for p and n-1 choices for q. Furthermore, by systematically generating negations of the original statements, the number of possible variations increases to 2n(2n-2), given the constraint that the value of q should be neither p nor $\sim$p. The resulting question bank can be imported into the LMS and used as the basis for homework exercises in which a subset of the questions is selected at random.

4 Implementation

The process described above has been used to generate questions of various structural types in several areas of our introductory discrete structures course. In this paper, we focus on student outcomes and associated questions related to the propositional logic area. The process involves, as its first step, the construction of a data file containing an arbitrary number of statements of the form *<subject> is <object>*. While various sentence structures could be used, having a simple and uniform structure facilitates the syntactic manipulations to be implemented by the software. A sample file might contain statements such as:

```
the weather in Chicago is windy
the food in Italy is delicious
the architecture in Paris is stunning
the color of the new dress is blue
```

The data file, which could be constructed manually or generated programmatically, can contain any number of statements, although as indicated before, the number of possible question variations involving two variables grows quadratically with respect to the number of statements in the source file. In practice, we have used data files containing approximately 30 statements. Furthermore, different data files could be used to generate alternative question banks with minimal effort.

For a question type with two variables, such as the examples presented in the previous section, the question-generation software module randomly selects values for <p> and <q>, randomly constructs the negation of either <p> or <q>, and constructs the question in a format compatible with the Moodle import feature. The system repeats this process as many times as necessary to generate the desired number of questions, which are then stored in an output file. Figure 1 contains a sample randomly generated question in Moodle XML format (some of the optional XML elements have been removed in the interest of saving space and aiding in comprehension).

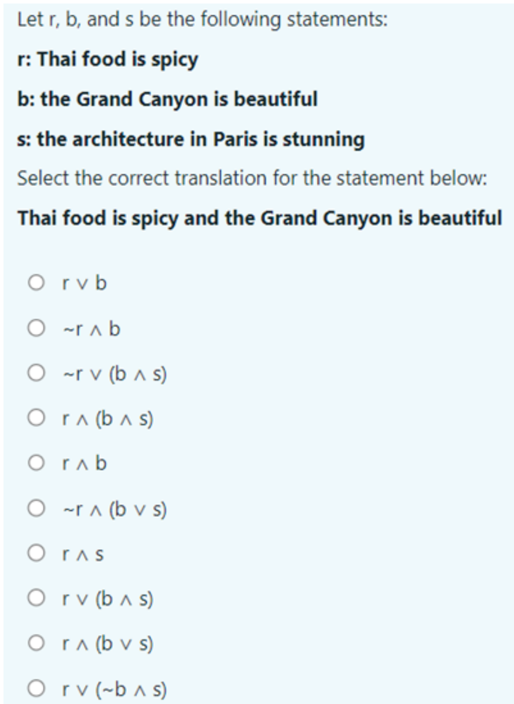
It should be noted that, although the correct answer (shown last in the code snippet) involves the conjunction of two statements, the software selects and presents three random statements to increase the number of possible false choices. The question options also specify that a single answer should be selected and that the possible answers should be shuffled every time a question is rendered. Figure 2 contains a snapshot of the question when rendered in a Moodle quiz. By randomly populating the values of the individual statements and shuffling the order in which choices are presented, there is greater difficulty in copying another student's answers.

After being stored in the LMS question bank, the generated questions are assigned randomly to students as homework exercises or quiz problems. When a sufficiently large number of questions is generated, the probability of having two or more students presented with the same question is very low. In addition, the random selection of exercises from a large set of questions facilitates the possibility of granting a student multiple attempts without repeating questions from one attempt to the next. Slight modifications to the software allow for the generation of pools of alternative questions whose correct answers match some of the other logic expressions, to be used as distractor choices in the sample question shown in Figure 2. Consequently, a homework assignment or other assessment that draws random questions from the various pools effectively introduces both parameter randomization and structural randomization.

Although not shown in Figure 1, the Moodle XML format allows the inclusion of general question feedback as well as individual feedback for each of the distractors. Figure 3 presents another randomly generated question in which the student is asked to apply known logical equivalences to select the two statements that are equivalent to the original implication. As explained in the previous example, parameter randomization is achieved by randomly selecting (and possibly negating) statements for the antecedent and consequent.

```xml
<question type="multichoice">
    <name> <text>Translate Sentence to Logic 21 </text> </name>
    <questiontext format='html'>
        <text><![CDATA[
            <p>Let r, b, and s be the following statements:</p>
            <p><strong>r: Thai food is spicy </strong></p>
            <p><strong>b: the Grand Canyon is beautiful </strong></p>
            <p><strong>s: the architecture in Paris is stunning </strong></p>
            <p>Select the correct translation for the statement below:</p>
            <p><strong> Thai food is spicy and the Grand Canyon is beautiful </strong></p>]]>
        </text>
    </questiontext>
    <defaultgrade>1.00</defaultgrade>
    <single>true</single>
    <shuffleanswers>true</shuffleanswers>
    <answernumbering>none</answernumbering>
    <answer fraction="0" format="html"><text><![CDATA[<p>~r &and; (b &or; s)</p>]]></text></answer>
    <answer fraction="0" format="html"><text><![CDATA[<p>r &or; b</p>]]></text></answer>
    <answer fraction="0" format="html"><text><![CDATA[<p>r &or; (b &and; s)</p>]]></text></answer>
    <answer fraction="0" format="html"><text><![CDATA[<p>r &or; (~b &and; s)</p>]]></text></answer>
    <answer fraction="0" format="html"><text><![CDATA[<p>r &and; (b &or; s)</p>]]></text></answer>
    <answer fraction="0" format="html"><text><![CDATA[<p>r &and; (b &and; s)</p>]]></text></answer>
    <answer fraction="0" format="html"><text><![CDATA[<p>r &and; s</p>]]></text></answer>
    <answer fraction="0" format="html"><text><![CDATA[<p>~r &and; b</p>]]></text></answer>
    <answer fraction="0" format="html"><text><![CDATA[<p>~r &or; (b &and; s)</p>]]></text></answer>
    <answer fraction="100" format="html"><text><![CDATA[<p>r &and; b</p>]]></text></answer>
</question>
```

Figure 1: Sample Question in Moodle XML Format.



Let r, b, and s be the following statements:

r: Thai food is spicy

b: the Grand Canyon is beautiful

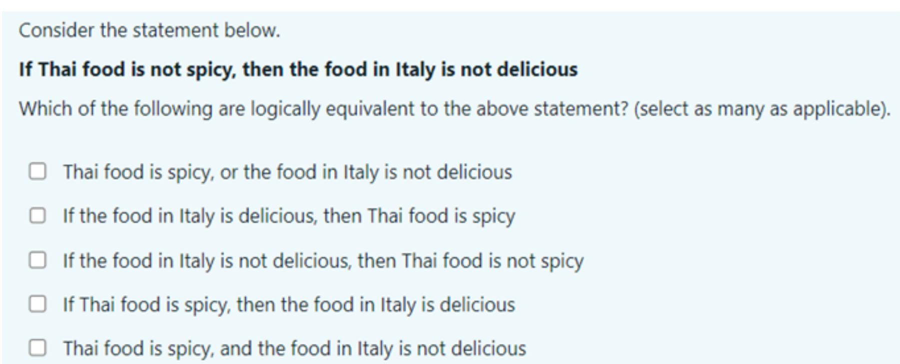
s: the architecture in Paris is stunning

Select the correct translation for the statement below:

Thai food is spicy and the Grand Canyon is beautiful

- ○ r ∨ b
- ○ ~r ∧ b
- ○ ~r ∨ (b ∧ s)
- ○ r ∧ (b ∧ s)
- ○ r ∧ b
- ○ ~r ∧ (b ∨ s)
- ○ r ∧ s
- ○ r ∨ (b ∧ s)
- ○ r ∧ (b ∨ s)
- ○ r ∨ (~b ∧ s)

Figure 2: Rendering of Sample Question in a Moodle Quiz.



Consider the statement below.

If Thai food is not spicy, then the food in Italy is not delicious

Which of the following are logically equivalent to the above statement? (select as many as applicable).

- ☐ Thai food is spicy, or the food in Italy is not delicious
- ☐ If the food in Italy is delicious, then Thai food is spicy
- ☐ If the food in Italy is not delicious, then Thai food is not spicy
- ☐ If Thai food is spicy, then the food in Italy is delicious
- ☐ Thai food is spicy, and the food in Italy is not delicious

Figure 3: Parameter Randomization with Multiple Correct Answers.

The process described in this section has been used to generate thousands of questions that test the student's ability to perform the following tasks in the context of propositional logic:

- Translate English sentences into symbolic logic expressions.
- Translate logic expressions into corresponding English sentences.
- Select the correct negation of English sentences involving disjunctions and conjunctions.
- Identify the contrapositive, converse, and inverse form of an English sentence containing an implication.
- Identify a logically equivalent sentence of an English sentence containing an implication.
- Select the correct negation of an English sentence containing an implication.

The question bank was used in three sections of our discrete structures course in the Spring 2023 semester totaling almost one hundred students. The large number of questions allowed us to assign homework exercises that draw random questions from one or more structural types in such a manner that no two students had the same questions. The same process was used when administering quizzes and tests in a proctored environment.

5 Summary and Avenues for Further Research

We have described the design and general implementation of a software system that generates questions to assess several learning outcomes in the area of propositional logic within the context of an introductory course in discrete structures. The generated questions have been utilized to assign homework exercises as well as proctored quizzes and tests in three sections of the course.

As explained in the introduction, the system was developed as part of a wider effort to study whether the randomization of homework assignments has a measurable effect on students' performance. While this paper has focused on the propositional logic area, we are extending the system to generate randomized questions to assess outcomes in the areas of predicate logic and combinatorics. As we expand the study to other areas of computer science, we will explore opportunities for randomizing questions in data structures and algorithms.

As mentioned in the paper, the system currently generates question files in Moodle XML format, which was selected because of its flexibility and ease of import. We plan to modify the software by adding an option to output questions using alternative formats that can be imported by other commonly used LMS platforms.

References

[1] Binglin Chen, Matthew West, and Craig Zilles. "How much randomization is needed to deter collaborative cheating on asynchronous exams?" In: *Proceedings of the Fifth Annual ACM Conference on Learning at Scale*. 2018, pp. 1–10. DOI: `10.1145/3231644.3231664`.

[2] Jose Cordova et al. "On the correlation between homework problems and test code-completion questions in a data structures course". In: *Journal of Computing Sciences in Colleges* 36.5 (2021), pp. 180–188.

[3] Susanna S Epp. *Discrete mathematics with applications*. Cengage learning, 2010.

[4] Mingyu Feng and Jeremy Roschelle. "Predicting students' standardized test scores using online homework". In: *Proceedings of the Third (2016) ACM Conference on Learning@ Scale*. 2016, pp. 213–216. DOI: `10.1145/2876034.2893417`.

[5] Christiane Frede and Maria Knobelsdorf. "Exploring how students perform in a theory of computation course using final exam and homework assignments data". In: *Proceedings of the 2018 ACM Conference on International Computing Education Research*. 2018, pp. 241–249. DOI: `10.1145/3230977.3230996`.

[6] Andrew Grodner and Nicholas G Rupp. "The role of homework in student learning outcomes: Evidence from a field experiment". In: *The Journal of Economic Education* 44.2 (2013), pp. 93–109. DOI: `10.2139/ssrn.1892173`.

Teaching and Learning With Virtual Reality*

Daniel C. Cliburn
Department of Computer Science
University of the Pacific
Stockton, CA 95211
dcliburn@pacific.edu

Abstract

Virtual reality (VR) is an exciting field with numerous application areas, one of which is education. It has never been easier for instructors to teach with and for students to learn about VR technologies. This paper provides an overview of many ways that VR technologies are being integrated into computing education. Software tools are described that support teaching and learning about VR, and the author's experiences teaching VR and using VR technologies are presented. The paper concludes with a discussion of some of the barriers to wide scale adoption of VR technologies in computing education.

1 Introduction

Virtual reality (VR) technologies have been around for decades, but only recently has the more affordable cost of high quality VR equipment allowed these technologies to be easily accessible to the average consumer. Faculty and students at practically every academic institution can now interact in the "metaverse," 3D virtual worlds accessible through the Internet and often in VR[7]. VR experiences can allow students to practice or observe concepts that would be difficult to do otherwise, or interact in ways not possible through other online mediums. Many published research studies report learning gains when VR technologies are used in educational settings[14].

The purpose of this paper is to provide an overview of many ways that VR technologies are being used in computing education. The next section defines the field of VR and discusses previous work to incorporate VR technologies into educational settings. Then, several tools for developing web-accessible VR content are described. Next, the author's experiences teaching with and about VR are presented. Finally, the paper concludes with a discussion of barriers to wide scale adoption of VR technologies in educational settings and directions for future work.

2 Background

Sherman and Craig[27] describe five elements of a VR experience: the participants, the creators, a virtual world, immersion, and interactivity. Clearly, there would be no need for VR content without someone to experience it, and there would be no VR content without someone to create it. Unfortunately, too often those with the domain knowledge necessary to conceptualize an effective educational VR experience have not had the expertise to create the experience. Similarly, those with the expertise to create VR applications often have not had sufficient domain knowledge to conceptualize an effective educational VR experience. Developing a partnership between educators and VR content creators that is beneficial to both has been challenging.

Virtual worlds are 3D computer generated spaces that allow interactions such as selecting and manipulating virtual objects, exploring virtual spaces, and interacting with other users. Immersion is the element of a VR experience described by Sherman and Craig[27] that may be most confusing to students, in part, because the term immersion can refer to multiple concepts. Sherman and Craig describe mental immersion as a "state of being deeply engaged," such as how you may feel when reading a good book or playing a great video game. Those in the VR community often refer to this phenomenon as presence. Sherman and Craig describe physical immersion as "bodily entering into a medium," which involves the use of technology devices (such as a VR headset) to present feedback to the user's senses. The goal of a true VR experience is to allow participants to interact with a virtual world through immersive technology devices while feeling a sense of presence greater than what is possible through traditional computing equipment (such as a computer monitor, keyboard, and mouse). However, there is certainly a place in educational settings for what is sometimes referred to as "desktop VR" or "non-immersive VR," which offers a lower level of physical immersion through traditional computing equipment, but is more broadly accessible since it does not require access to special VR technology devices[24].

Educators have been discussing the use of VR technologies for decades[20, 22]. VR technologies have been used to create educational experiences in fields such as anatomy[12], art history[6], astronomy[5], and criminology[13]. Computing educators have been using VR technologies to teach computer graphics[15], sorting algorithms[2, 25], and interdisciplinary collaboration[21]. Students have found VR learning experiences to be more engaging and preferable to similar non-immersive learning tools[25]. Cui and colleagues[12] report that after experiencing their immersive learning sessions, students with lower spatial abilities were able to score comparatively on assessments to students with higher spatial abilities, suggesting that at least for some types of content, VR might be particularly useful to students with lower spatial abilities.

Bricken[3] was among the first to discuss the concept of a virtual reality learning environment (VRLE), which can provide students with virtual active learning experiences that can be shared with others. VRLEs can motivate and engage students[11, 23, 16], and their usage has been discussed for application areas such as special education[17], industry[18], and engineering education[28]. VRLEs have been used to hold online class meetings[10, 11, 23] as an alternative to other forms of distance education. Many modern VRLEs allow students to represent themselves as avatars, and communicate with each other through voice and text chat. Figure 1 shows an example of the author's VRLE developed using Mozilla Hubs.

Figure 1: A virtual reality learning environment created with Mozilla Hubs.

3 Tools for Developing Web-Accessible VR

While game engines, such as Unreal Engine (unrealengine.com) and Unity (unity.com), are used to develop much of the commercial VR content available today, many other tools are available for creating web-accessible VR experiences. One of the major benefits of creating web-accessible VR content is that it is easy to share with others. Students can simply open the browser app on their VR headset and enter the web address given to them by their instructor to join an immersive VR class session or other educational VR experience. Another advantage of these tools is that they also allow for non-immersive access, meaning that someone without a VR headset can still experience the content through a browser on their computer.

Mozilla Hubs (`hubs.mozilla.com`) supports the development of custom virtual worlds accessible online through a web browser or a VR headset. Students can customize avatars to represent themselves in the virtual space (see Figure 1). Hubs supports screen sharing, so instructors can present lecture slides during a virtual class session. Hubs also supports voice chat and 3D spatial sound, so someone's voice becomes louder the closer an avatar is to theirs. This makes it possible for students to split up into groups and go to separate locations in the virtual space, so they can hold conversations not easily overheard by others. No coding experience is necessary to create a VRLE in Hubs. Instead, Mozilla supplies a tool called Spoke, for customizing Hubs spaces. Mozilla recommends a maximum capacity of 25 guests per room in Hubs.

Virbela Frame (`learn.framevr.io`) is similar to Mozilla Hubs in many ways. Frame supports the creation of VRLEs accessible through a web browser or VR headset, with no coding necessary. Students can join virtual classrooms using avatars, communicate through voice chat, and instructors can share their screen to present lecture slides during virtual class sessions. For a fee, Frame can accommodate up to 100 users at once.

A-Frame (`aframe.io`) supports development of VR applications that run in a web browser using HTML and JavaScript. Students can use a tool called Glitch to edit their code and deploy it online for free. A-Frame projects can be experienced in VR through the browser app in a VR headset. The A-Frame website contains numerous examples, tutorials, and documentation to help students get started creating their own custom browser based VR applications.

PlayCanvas (`playcanvas.com`) is a browser-based game engine that can be used to develop VR applications for the web. Like A-Frame, custom scripts can be written with JavaScript and applications can be experienced in VR through the browser app in a VR headset. The PlayCanvas Editor looks similar to Unity's interface, so students with Unity experience should be able to learn PlayCanvas quickly. An unlimited number of public projects can be hosted on the PlayCanvas website for free (projects can be made private for a fee). Numerous tutorials and other learning resources are available on the PlayCanvas website.

4 Teaching and Learning with VR

The author first became interested in VR technologies while a graduate student in the year 2000. In his early years as a faculty member, he focused on how to teach VR concepts to undergraduate students at primarily undergraduate institutions. The cost of the technologies necessary for teaching immersive VR were substantially more than today, creating significant challenges. Most VR education at the author's institution took place as a unit in an undergraduate computer graphics course[9], and focused primarily on how to create VR applications. A few years later, the author began holding occasional class sessions in Second Life (`secondlife.com`), which was used as a non-immersive VRLE[10].

In 2013, the Oculus Rift DK1 was introduced, which led to a dramatic reduction in the cost of high-quality VR headsets. VR headsets have become affordable and quite prevalent (at the time of this writing, a Meta Quest 2 can be purchased for under $400). With game engines such as Unity and Unreal Engine, developing VR content has never been easier, particularly for those without significant programming experience. During this period, the author began to spend less class time on how to develop VR applications, and more time on foundations of VR and evaluation of VR content.

In 2018, the author began teaching a cross-disciplinary VR course[8] that has been taken by students from Computer Engineering, Computer Science, Education, Engineering Management, Psychology, Mechanical Engineering, and Media X (a major focusing on the intersection of arts, technology, and emerging media). The first third of the course covers applications of VR, the second third on foundations of VR, and the final third on evaluating VR experiences. Working in cross-disciplinary teams, students developed and then evaluated VR applications created for the Oculus Rift and HTC Vive using the Unity game engine.

In 2021, the author obtained funding to purchase Oculus Quests for every student enrolled for the VR course and for the first time most class sessions were held in immersive VR[4]. A VRLE was created using Mozilla Hubs, and students could join class sessions through the browser app in their Oculus Quests. Thus, students were fully immersed in VR while learning about VR. However, it should be noted that students were required to be on campus to work with their assigned teams on their final course projects. While students felt that learning about VR in VR was a positive experience, some students indicated that wearing their Quest for long periods was uncomfortable, and it was very difficult to take notes during class sessions while wearing the Quest.

In fall of 2023, for the first time a completely virtual option of the VR course will be offered. The goal is to make the course as broadly accessible to as many students as possible, and from as many majors as possible. All class sessions will be held in a VRLE, which students can attend through a browser in their VR headset or their personal computer. Students will not have to travel to campus to attend any class sessions or to complete course assignments. Students will have the option to complete final course projects individually, or in teams. Students will also have the option to complete final course projects using immersive VR equipment available on campus or complete web-based projects using their own equipment and Mozilla Hubs, Virbela Frame, A-Frame, or PlayCanvas. Students that choose team-based final projects will develop virtual reality applications proposed by faculty and staff at the author's institution. Students that choose individual final projects will develop VR applications that they propose themselves. Instead of a final exam, all students will present their final project at a virtual reality showcase at the end of the term. Students may present in person on campus, or virtually, whichever option they prefer and that best fits their final project.

5 Conclusion

In 1991, Bricken[3] identified cost, usability, and fear as barriers to the adoption of VR technologies in educational settings. The cost of VR equipment has dropped substantially since 1991, and the number of users experiencing social VR applications in the metaverse has never been greater. Thus, cost and fear are less likely to be barriers to the adoption of VR in educational settings now than in 1991. However, usability may still be a concern. In a recent study, Pirker and colleagues[26] identified "design of the user interface" as a common issue with educational VR applications for computer science education. Furthermore, as described earlier, some tasks, such as note-taking, that are straight forward in the real world are challenging in VR.

One of the greatest barriers today to the adoption of immersive VR technologies in computing education may be determining appropriate usages with quantifiable benefits. Liu and colleagues[19] have begun to investigate the types of learners that may benefit most from educational experiences in VR. Other researchers report recently that, while VR is often cited as improving attitudes of learners, increasing motivation, and providing an efficient learning environment, more evidence-based research should be conducted to explore the effectiveness of VR in computing education[1]. Many studies investigating educational uses of immersive VR are short term, and do not provide students with multiple opportunities to experience the educational VR content[14]. Pirker and colleagues[26] identified exploration into potential use cases of VR in classroom settings as an important direction for future work.

Despite these barriers and limitations, VR remains a topic of great interest to many computing students and educators. VR technologies have never been more accessible, and developing VR content has never been easier. Identifying effective ways to teach VR, utilize immersive technologies in the classroom, and construct educational VR experiences remain important areas for future work.

References

[1] Friday Joseph Agbo et al. "Application of virtual reality in computer science education: A systemic review based on bibliometric and content analysis methods". In: *Education Sciences* 11.3 (2021), p. 142.

[2] Akhan Akbulut, Cagatay Catal, and Burak Yıldız. "On the effectiveness of virtual reality in the education of software engineering". In: *Computer Applications in Engineering Education* 26.4 (2018), pp. 918–927.

[3] Meredith Bricken. "Virtual reality learning environments: potentials and challenges". In: *Acm Siggraph Computer Graphics* 25.3 (1991), pp. 178–184.

[4] Keely Canniff and Daniel Cliburn. "Teaching virtual reality in virtual reality". In: *2022 8th International Conference of the Immersive Learning Research Network (iLRN)*. IEEE. 2022, pp. 1–5.

[5] Hubert Cecotti. "A Serious Game in Fully Immersive Virtual Reality for Teaching Astronomy Based on the Messier Catalog". In: *2022 8th International Conference of the Immersive Learning Research Network (iLRN)*. IEEE. 2022, pp. 1–7.

[6] Hubert Cecotti et al. "Virtual reality for immersive learning in art history". In: *2020 6th International Conference of the Immersive Learning Research Network (iLRN)*. IEEE. 2020, pp. 16–23.

[7] Peter Allen Clark. "The Metaverse has already arrived. Here's what that actually means". In: *Time Mag* (2021).

[8] Daniel C Cliburn. "A cross-disciplinary course on virtual reality". In: *2018 IEEE Frontiers in Education Conference (FIE)*. IEEE. 2018, pp. 1–5.

[9] Daniel C Cliburn. "Incorporating virtual reality concepts into the introductory computer graphics course". In: *2006 37th Technical Symposium on Computer Science Education (SIGCSE)*. ACM. 2006, pp. 368–372.

[10] Daniel C Cliburn and Jeffrey L Gross. "Second Life as a medium for lecturing in college courses". In: *2009 42nd Hawaii international conference on system sciences*. IEEE. 2009, pp. 1–8.

[11] Murat Çoban and İdris GOKSU. "Using virtual reality learning environments to motivate and socialize undergraduates in distance learning". In: *Participatory Educational Research* 9.2 (2022), pp. 199–218.

[12] Dongmei Cui et al. "Evaluation of the effectiveness of 3D vascular stereoscopic models in anatomy instruction for first year medical students". In: *Anatomical sciences education* 10.1 (2017), pp. 34–45.

[13] F Jeane Gerard et al. "Work-In-Progress—CrimOPS—Gamified Virtual Simulations for Authentic Assessment in Criminology". In: *2022 8th International Conference of the Immersive Learning Research Network (iLRN)*. IEEE. 2022, pp. 1–3.

[14] David Hamilton et al. "Immersive virtual reality as a pedagogical tool in education: a systematic literature review of quantitative learning outcomes and experimental design". In: *Journal of Computers in Education* 8.1 (2021), pp. 1–32.

[15] Birte Heinemann, Sergej Görzen, and Ulrik Schroeder. "Teaching the basics of computer graphics in virtual reality". In: *Computers & Graphics* 112 (2023), pp. 1–12.

[16] Hsiu-Mei Huang, Ulrich Rauch, and Shu-Sheng Liaw. "Investigating learners' attitudes toward virtual reality learning environments: Based on a constructivist approach". In: *Computers & Education* 55.3 (2010), pp. 1171–1182.

[17] Horace HS Ip and Chen Li. "Virtual reality-based learning environments: Recent developments and ongoing challenges". In: *Hybrid Learning: Innovation in Educational Practices: 8th International Conference, ICHL 2015, Wuhan, China, July 27-29, 2015, Proceedings 8*. Springer. 2015, pp. 3–14.

[18] Vasiliki Liagkou, Dimitrios Salmas, and Chrysostomos Stylios. "Realizing virtual reality learning environment for industry 4.0". In: *Procedia Cirp* 79 (2019), pp. 712–717.

[19] Jiaxu Liu et al. "Which Types of Learners Are Suitable for the Virtual Reality Environment: A fsQCA Approach". In: *2022 8th International Conference of the Immersive Learning Research Network (iLRN)*. IEEE. 2022, pp. 1–5.

[20] Paul Moore. "Learning and teaching in virtual worlds: Implications of virtual reality for education". In: *Australasian Journal of Educational Technology* 11.2 (1995).

[21] Eric Nersesian et al. "Interdisciplinary Collaboration Approaches on Undergraduate Virtual Reality Technology Projects". In: *2020 IEEE Integrated STEM Education Conference (ISEC)*. IEEE. 2020, pp. 1–8.

[22] Max M North, Joseph Sessum, and Alex Zakalev. "Immersive visualization tool for pedagogical practices of computer science concepts: A pilot study". In: *Journal of Computing Sciences in Colleges* 19.3 (2004), pp. 207–215.

[23] Eileen A O'Connor and Jelia Domingo. "A practical guide, with theoretical underpinnings, for creating effective virtual reality learning environments". In: *Journal of Educational Technology Systems* 45.3 (2017), pp. 343–364.

[24] Sujni Paul and Saif Hamad. "The Role of Virtual Reality in Story telling and Data Visualization for motivating students in learning programming". In: *2020 Seventh International Conference on Information Technology Trends (ITT)*. IEEE. 2020, pp. 169–173.

[25] Johanna Pirker et al. "The potential of virtual reality for computer science education-engaging students through immersive visualizations". In: *2021 IEEE Conference on Virtual Reality and 3D User Interfaces Abstracts and Workshops (VRW)*. IEEE. 2021, pp. 297–302.

[26] Johanna Pirker et al. "Virtual reality in computer science education: A systematic review". In: *Proceedings of the 26th ACM symposium on virtual reality software and technology*. 2020, pp. 1–8.

[27] William R Sherman and Alan B Craig. *Understanding virtual reality: Interface, application, and design*. Morgan Kaufmann, 2018.

[28] Diego Vergara, Manuel Pablo Rubio, and Miguel Lorenzo. "On the design of virtual reality learning environments in engineering". In: *Multimodal technologies and interaction* 1.2 (2017), p. 11.

Like a Bee to a Honeypot: A Bug Bounty Capstone Project*

Steven Fulton, Matthew Dickerman, Madison Gillan,
Krittawata Su-uthai, Alison Thompson, and Peter Ye
Department of Computer and Cyber Sciences
United States Air Force Academy
USAFA, CO 80841
{steven.fulton, c23matthew.dickerman,
c23madison.gillan, c23krittawata.su-uthai,
C23alison.thompson, c23peter.ye}@afacademy.af.edu

Abstract

Our research focuses on the use of a commercial bug bounty program as part of a senior capstone project for Computer Science and Cyber Science majors. We look at the use of such programs as fulfillment of the accreditation requirement for a major project which requires integration and application of knowledge and skills acquired in earlier course work. We approached the project by using the Cyber Attack Methodology to systematically test vulnerabilities and identify possible areas for concern for the corporation. We also provided an overview of our experience and associated findings for other universities that may wish to use a similar program as a capstone experience. This paper discusses the process we used to analyze the corporate environment, as well as the general findings of our attempt.

1 Introduction

At the United States Air Force Academy (USAFA), seniors who are majoring in Computer Science and Cyber Science can take advanced computing courses which include cyber warfare, cyber defense, and other related topics (`https://www.usafa.edu/department/computer-science/`). While these classes have proven useful in gaining an educational background, students may not have an opportunity to work on large multi-dimensional projects until their senior year when they are given an opportunity to work on a two semester capstone project. USAFA's Computer Science and Cyber Science programs are accredited by the Accreditation Board for Engineering and Technology (ABET) which have a student outcome that focuses on the ability to function effectively as a member or leader of a team engaged in activities appropriate to the program's discipline (`https://www.abet.org/accreditation/accreditation-criteria/criteria-for-accrediting-computing-programs-2019-2020/`). We use our capstone as an opportunity to meet this requirement.

The use of bug bounties to find and patch security flaws in software applications has grown in popularity in the field of cybersecurity since 1983, when a Silicon Valley start-up offered anyone to "get a bug if you find a bug"[7]. These initiatives incentivize those who discover and report flaws, thus giving engineers useful feedback and enhancing the software's overall security. Companies may choose to perform penetration testing (pen testing) by hiring one ethical hacker or by crowd sourcing the process, making it available to a large number of ethical hackers via a bug bounty (paying for the vulnerabilities reported by participating ethical hackers)[3]. In our case, a group of students participated in a public bug bounty for a large corporate organization, thus putting their technical and security skills to the test in a practical setting. This project was a chance for these cadets to develop their technical skills in a controlled setting, while gaining real-world experience in the field of cybersecurity. In this paper, we'll examine the value of practical hacking and bug finding experiences for college students, and how the experiences might help them become ready for careers in cybersecurity. We also review the process used in the bug bounty event and discuss our findings.

2 Background and Related Work

We were recently approached by a large corporation to consider creating a team to participate in a bug bounty for our capstone project. The team utilized the 5-phase Cyber Attack Methodology[11] to formulate a plan of attack, which included different attack vectors to potentially gain access. We opted to use this methodology as it is taught in our core computer science class and we felt that the clarity and simplicity of the methodology fit our project goal. Our plan of attack focused on attack vectors, such as web-based attacks and social engineering, throughout the semester to attempt to gain and maintain access.

Unlike traditional universities, the United States Air Force Academy is a federal entity. It was important that we address some possible concerns prior to the start of the bounty. The use of United States Government purchased computer systems and software could be seen as governmental overreach in an attempt to attack a private organization. To overcome this concern, we asked the company to provide baseline hardware and software configurations for the attack machines. Furthermore, the faculty mentor required that the students create rules of engagement to ensure that the students were aware of the limits of their ability to break into the system. Otherwise, the only limits placed upon the team were the same limits defined by the corporation's published bug bounty program. Finally, since our students are in fact, government personnel, they are not able to accept any monetary 'bounty' from any of their findings and we had to come to an understanding regarding any possible bounty.

Along with the research conducted on the company, our team also conducted research focused on similar programs offered at other universities and companies. These articles helped to establish a framework for how we would conduct our project and how we should aim to gain access into the company's systems. Tjaden and Tjaden[12] and more recently Whitman and Mattord[13] describe college-level courses focused on defending computer networks and systems. While these courses include some penetration testing, they are more focused on internal defense rather than attacking external companies. Tjaden and Trajen[12] most closely describe our capstone project, as it focuses on an undergraduate course that allows students to practice penetration testing on actual applications owned and used by the university. However, this differs from our capstone project in that it takes place in a classroom setting for graded events, rather than having a team dynamically focused purely on performing penetration testing on a company.

In 2006, Irvine, Rose, and Fably[5] hosted a workshop discussing practical and experimental approaches to information security education, in which they discussed topics such as case studies[1] and threat modeling[8] to provide capstone experiences in computer information security. At the same workshop, it was discussed that there needs to be an ethical understanding prior to teaching students how to use tools to break into computer systems[10]. Pauli and Engebretson[9], a mere six years later, emphasized the importance of hands-on learning, which supports the need for developing capstones that include active learning components, such as our the bug bounty project.

In a master's thesis in 2018, Christian[2] pointed out how bug bounties could be the future of vulnerability research if organizations are mature enough to run a fully public bounty program. By 2022, Kapoor, Penton and Pierpont[6] were describing how the University of Florida has implemented a bug bounty program outside the computer security world, in which identified bugs associated with class structure (syllabi, course website, etc) were being submitted by students to present formative course feedback. Students would receive points toward their course if they identified a course issue.

Harper[4] discusses the use of bug bounty programs by the Department of Defense to crowd source their computer vulnerability identification by encouraging friendly hackers to "probe for and identify vulnerabilities", allowing for a more secure environment.

While much exists in literature about the pros and cons of bug bounties, it is important to note that our capstone project differs from the articles in that it is being used as an undergraduate capstone class and, in some ways, is focused on skill-building rather than actually finding bugs. We view this as a learning opportunity for each member of our team to gain hands-on experience in cyber offensive and defensive fields, which will be crucial in our future careers.

3 Team and Class Structure

Our capstone team was made up of five students: three Computer Science majors, one Cyber security major and a double Computer Science/Cyber Science major. The course is a two semester, three credit hour per semester course, which is a requirement for Computer Science and Cyber Science majors. Students are provided the opportunity to choose their capstone at the beginning of the first semester and with rare exceptions must remain with the course for the entire year. Some basic team compatibility testing is performed at the start of the first semester to understand the compatibly of the team. Each team is matched with a faculty mentor whose specialty is focused on the subject area of the topic being evaluated. In this case, the faculty member had not only published on cybersecurity topics, but also worked in the cyber career field prior to teaching at our institution.

The team met 40 times a semester for a two hour block of time. The expectation was that while much of the work could be accomplished during this time frame, the students would be required to complete tasks outside of the normal working time.

4 Approach

In the beginning of the project, the team consulted the partner company to create a definitive list of rules of engagement (ROE) that the team will follow throughout the project. The set of rules included everything from how we planned the attacks, which computers were going to be used in the attack, which networks we were going to use, which new tools were going to be used, and the like. There were also rules which were placed on us by the bug bounty itself, such as no physical destruction of company property and no denial of service to employees or customers.

We began the bug bounty by gathering information and conducting research to gain more insight into the company's structure and the tools available to use. For example, we were able to identify and sort thousands of DNS domains belonging to the organization, find important individuals at the company to begin a map of corporate personnel, and research possible malware available for use in attacking the systems. For the individuals at the company, we found that it was most beneficial to find them on websites such as LinkedIn, GitHub, and Facebook. We also researched possible penetration software and other tools which could be of use. We used Burp Suite, a commercially available software package, to attempt to perform the pentesting attacks on the corporation. We also decided to use GoPhish as a resource in email campaigns.

From this research, we were able to gain a more solid background on the company, malware tools available, and more to ensure that the project was successful after this reconnaissance stage. We were then able to organize an attack log, lists of company contacts for phishing, and sorted scans/domain lists for web exploitation. This organization allowed us to become more successful in the attack portions of the project and coordinate attacks with one another. We decided to take a three pronged approach to the system: Technical attacks against the corporate resources, a social engineering campaign (primarily using Linked In), and a phishing campaign to see how likely the employees were to open email and click on a link. This three pronged approach in demonstrated in Figure 1.

4.1 Network Mapping and Scanning

The team was able to find and sort through over 21,000 DNS addresses. The team utilized a divide-and-conquer method of approaching this task and was able to sort domains by type including log-in pages, non-existent pages, error pages, and more. By accomplishing this task, we had a strong foundation to begin web-based attacks. Additionally, we researched individuals working for the corporation to create a web of employees for easier phishing attacks. We conducted this research through social media sites, such as LinkedIn, to receive employee names and job titles. Our team was able to find more than 2,000 employees in various locations across the country, therefore making it much easier to find an individual that would be willing to click a link in a phishing campaign or false web page.

From this research, our team was better able to begin attacks against the

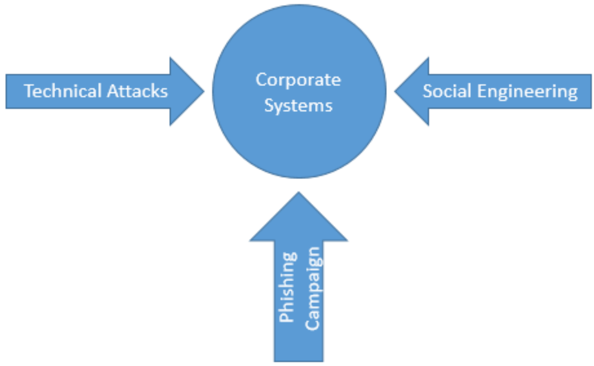


Figure 1: Corporate Attack Plan

organization. We had solidified targets for phishing including names, job titles, and email addresses, which were expanded throughout the semester. Using Burp Suite, we began identifying web-based attacks and possible malware usage on the domains that were initially sorted.

4.2 Phishing

We executed a phishing campaign that involved sending emails to employees of the targeted organization. To make the emails appear more legitimate, we also created fake LinkedIn accounts that were used to lure employees into providing sensitive information. Despite evidence of spam filtering, our phishing emails were successfully delivered to the employees. We further investigated how the company handles incoming emails, and discovered a DKIM vulnerability that allowed our emails to get into the company's system undetected. This vulnerability underscores the importance of robust email security measures to prevent phishing attacks from being successful.

4.3 Port Exploitation

After using the scan function on Burp Suite, we identified several IP addresses that were flagged as vulnerable. The program shows multiple levels of vulnerabilities including high, medium, and low vulnerabilities. The team focused on the high and medium vulnerabilities and found approaches to exploit the vulnerabilities.

4.4 Web Exploitation

We identified several IP addresses that were flagged as vulnerable. The program shows multiple levels of vulnerabilities including high, medium, and low vulnerabilities. The team focused on the high and medium vulnerabilities and found approaches to exploit the vulnerabilities.

HTTP request smuggling is a technique for interfering with the way a web site processes sequences of HTTP requests that are received from one or more users. Request smuggling vulnerabilities are often critical in nature, allowing an attacker to bypass security controls, gain unauthorized access to sensitive data, and directly compromise other application users. The site had been taken down and was presenting a 403 error, suggesting that the website understood our request but refused to answer it. We attempted (unsuccessfully) to bypass 403, in order to try HTTP smuggling. Multiple bypassing attacks were used on all 403 pages found, but nonetheless, the servers blocked the connection after time limit has expired. Overall, no vulnerabilities were found.

Cross-Origin Resource Sharing (CORS) is an HTTP header-based mechanism that allows a server to indicate any origins (domain, scheme, or port) other than its own, from which a browser should permit loading resources. After scanning various domains with Burp Suite, we found one vulnerability on one domain. However, the server forbade clients from accessing the input box and HTTP header.

SQL injection is a common attack vector that uses malicious SQL code for back-end database manipulation to access information that was not intended to be displayed. There were multiple attempts to exploit the user input injection, however, no vulnerabilities were found.

A brute force login attack was attempted by using a dictionary attack. There were four main sites that were attacked: an older site that had not been updated since 2013, a Cisco Network Management service, and two FTP servers. The sites were discovered in our initial reconnaissance, and it is believed that the organization did not have knowledge that they were public facing. The attacks on these sites were conducted utilizing Burp Suite Intruder, with the IP Rotate Extension enabled, to make it appear that the login attempts were coming from different IP addresses. We quickly discovered that we needed to create a DNS domain for Burp Suite intruder to rotate, so we created a similar domain to the organization's domain. We attempted more than one million username and password attempts with one success occurring with the default credentials for a Cisco network management service. We were disappointed to discover a second login page was behind the initial page. Shortly after we discovered this access, GET requests began using RSA verification, so it is likely that the organization discovered our attempts and put a blocking system in place.

5 Conclusion

During the capstone, we took several actions to access an organization's system. We divided our tactics into social engineering, web-based attacks, and malware. Under social engineering, we created a phishing campaign and forged LinkedIn accounts to lure employees for information. Our emails were delivered successfully, and we found a DKIM vulnerability as our email entered the company's system. In web-based attacks, we scanned over 1,000 IP addresses and discovered 57 non-standard ports accessible across four sub-nets registered to the organization by using nmap and other available tools. Additionally, we found numerous hidden Cisco product sign-on pages with default credentials enabled and one SSH connection allowing anonymous login. In web exploitation, we used the Burp Suite scan function to identify vulnerabilities in IP addresses and exploit high and medium vulnerabilities. We attempted HTTP smuggling and Cross-Origin Resource Sharing, but the server forbade clients from accessing the input box and HTTP header. We also did SQL injection, 403 bypasses, and brute force login but did not find any vulnerabilities. We learned about the company's robust security measures and gained insights into improving our hacking techniques for future college-level projects.

While we did not successfully exploit all the vulnerabilities we identified, we gained important insights into the company's security measures. For example, we learned that the company had implemented robust spam filtering for incoming emails, making it difficult for phishing campaigns to succeed. We also found that the company had successfully implemented multi-factor authentication, which helped to prevent unauthorized access to sensitive information. These insights allowed us to understand the company's security posture better and identify potential improvement areas.

Using an active bug bounty event as our capstone project allowed us to perform pentesting on a true networked environment, as opposed to a lab environment with simulated vulnerabilities. Our team was able to identify several vulnerable areas which needed to be hardened by the corporation. The information was shared with the company to allow for them to secure their system.

6 Future Research

Based on the results from this capstone, there are several recommendations for future research that could help penetrate a company's cybersecurity defenses. We realized the need to start phishing campaigns and social engineering early to allow more time for our team to pivot if needed and for the responses from the employees. It is crucial to avoid spending too much time on a single vulnerability and instead focus on a broad range of vulnerabilities to ensure comprehensive protection. Frequent scanning for exploitation is important to ensure that new vulnerabilities are quickly detected and exploited. Finally, we discovered the value of taking advantage of real-world events. In our case, employee layoffs were announced shortly after we started our bug bounty, and we found that employees were reaching out to us to find answers regarding their employment, thinking that we were actually part of the company. Hackers can take advantage of the uncertainty and fear among employees in such a situation to launch phishing campaigns.

"*The views expressed in this article, book, or presentation are those of the author and do not necessarily reflect the official policy or position of the United States Air Force Academy, the Air Force, the Department of Defense, or the U.S. Government.*"

References

[1] Shiva Azadegan et al. "Undergraduate Computer Security Education". In: *Practical and Experimental Approaches to Information Security Education* (2006), p. 17.

[2] Joseph L Christian. "Bug bounty programs: Analyzing the future of vulnerability research". PhD thesis. Utica College, 2018.

[3] Aaron Yi Ding, Gianluca Limon De Jesus, and Marijn Janssen. "Ethical hacking for boosting IoT vulnerability management: A first look into bug bounty programs and responsible disclosure". In: *Proceedings of the Eighth International Conference on Telecommunications and Remote Sensing*. 2019, pp. 49–55.

[4] Jon Harper. "Silicon Valley Could Upend Cybersecurity Paradigm". In: *National Defense* 101.759 (2017), pp. 32–34.

[5] "Practical and Experimental Approaches to Information Security Education". In: *Proceedings of the Seventh Workshop on Education in Computer Security (WECS7)*. Ed. by Cynthia Irvine, Matthew Rose, and Naomi Falby. 2006-01. URL: `http://hdl.handle.net/10945/35011`.

[6] Amanpreet Kapoor, Andrew Penton, and Hamish Pierpont. "Eliciting course feedback through a bug bounty program". In: *Proceedings of the 27th ACM Conference on on Innovation and Technology in Computer Science Education Vol. 2*. 2022, pp. 595–596.

[7] P Marks. “Bounties mount for bugs”. In: *Commun. ACM* (2018). URL: https://cacm.acm%20.org/news/230582-bounties-mount%20-for-bugs/fulltext.

[8] Jide B Odubiyi and Casey W O’Brien. “Information security attack tree modeling”. In: *Practical and Experimental Approaches to Information Security Education* (2006), p. 29.

[9] Josh Pauli and Patrick Engebretson. “Filling your cyber operations training toolbox”. In: *IEEE Security & Privacy* 10.5 (2012), pp. 71–74.

[10] Scott J Roberts and Andrew L Reifers. “Adding When, Where,and Why To How”. In: *Practical and Experimental Approaches to Information Security Education* (2006), p. 39.

[11] Edward Skoudis and Tom Liston. *Counter hack reloaded: a step-by-step guide to computer attacks and effective defenses*. Prentice Hall PTR, 2005.

[12] Brian Tjaden and Brett Tjaden. “Training students to administer and defend computer networks and systems”. In: *Proceedings of the 11th annual SIGCSE conference on Innovation and technology in computer science education*. 2006, pp. 245–249.

[13] Michael E Whitman and Herbert J Mattord. “Instructional Perspective on Cyber Defense: From Collegiate Competition to Capstone Course”. In: *Information Security Education Journal, Vol 2, Number 1* (2014).

Digital Circuit Projects for an Accelerated Online Undergraduate Computer Architecture Course*

Bin Peng and John Cigas
Computing and Mathematical Sciences
Park University
Parkville, MO 64152
bpeng@park.edu, cigas@acm.org

Abstract

Digital Logic is an essential topic in the Computer Science curriculum. This paper introduces a set of circuit-building projects to teach digital logic and circuits in an eight-week online version of an undergraduate computer architecture course. Those projects cover circuit design and building in a software simulator by progressing through gate-level design, sub-circuits, and circuit component-level integration. Students commented positively on gaining a deeper understanding from those hands-on projects. Student performance and feedback are presented and discussed.

1 Introduction

Digital Logic is an essential topic in the Computer Science curriculum. The Computer Science Curricula 2013[2] recommends digital logic and digital systems as a required topic for the Architecture and Organization knowledge area, which "develops a deeper understanding of the hardware environment upon which all computing is based, and the interface it provides to higher software layers." At the beginning of their computing curriculum, computing students learn that logic gates are basic building blocks for digital circuitry. However, many may not fully understand how digital circuits implement computing and control functionalities computers need until a computer architecture or organization course.

The authors teach an undergraduate-level computer architecture course at a four-year liberal arts institution. The course covers combinational and sequential circuits and how such circuits are used to build fundamental computer components like the arithmetic-logic unit, registers, central processing unit, and memory. As an active learning strategy[1, 7], the authors designed a set of projects on designing and building combinational and sequential circuits in a software simulator. Students commented positively on the active learning approach and gaining a deeper understanding from building those circuits themselves.

The remainder of this paper is organized as follows. Section 2 explains the background of the work. Section 3 covers the setup of the projects within the course. Section 4 describes each project in detail. Student performance and feedback are discussed in Section 5. Section 6 offers concluding remarks and possible ideas for adoption.

2 Background

The computer architecture course at the authors' institution is an undergraduate-level course that covers data representation, logic and circuits, memory system organization, and assembly-level machine organization. The course uses a bottom-up approach in which logic gates and circuits are covered right after data representation. The course has two prerequisites, discrete mathematics and an introductory programming course. The course content is driven by the institution's student profile and degree requirements. Most of our students are non-traditional and part-time, taking classes in an accelerated online 8-week format as their schedules allow. Most have started their degree programs at one or more institutions and have transferred a sometimes significant portion of their overall bachelor's degree courses. Most of these students are currently employed. Some seek a career change, while others need to complete their degree for career advancement. To serve that population, the department offers a broad computing degree with several concentrations sharing a common set of core classes. The architecture course is required in several concentrations, ranging from a more traditional computer science track to an information technology track, specifically designed to serve students who prefer less mathematics in the curriculum. Because of these characteristics and constraints, projects in our online courses need to be software-based and completable in the compressed, 8-week format of our online courses.

The projects described in this paper differ from other sets of classroom activities, such as those using a physical breadboard[4] or Raspberry Pi[9], Logisim simulations of a complete ALU[6], or web-based circuit evaluations[5]. The online, 8-week course format precludes using physical hardware, which is difficult to obtain and support outside a lab environment. The 8-week limitation makes ALU-based projects too challenging as students don't have enough time to build up skills and then complete a significant project. Web-based circuit evaluations are helpful for initial learning and perhaps scaffolding, but lack the cohesiveness of a meaningful project.

3 The Course Organization

Here is the overall schedule of the 8-week version of our computer architecture course:

Unit 1: Number Systems; Two's Complement
Unit 2: IEEE 754; Boolean Algebra Review
Unit 3: Combinational Circuits
Unit 4: Sequential Circuits
Unit 5: Memory Hierarchy
Unit 6: von Neumann Architecture
Unit 7: x86 Architecture
Unit 8: Review and Final Exam

The three projects progress through gate-level design, using sub-circuits, and circuit component-level integration in Units 3, 4, and 5. Unit 2 of the course provides a quick review of logic gates and the concept of combinational circuits, as those topics have been covered in discrete mathematics, which is one of the prerequisite courses. Unit 3 introduces designing and building combinational circuits and covers combinational circuits used in computers. The first project, therefore, is placed in Unit 3 for students to practice designing and building a combinational circuit in Logisim (`http://www.cburch.com/logisim/`). After learning about sequential circuits in Unit 4, the second project covers building a sequential circuit and feeding its output to a combinational circuit. Finally, in Unit 5, students build a memory reading circuit using Logisim components.

Before adding those projects, units 3 - 5 focused only on theory and reading circuits, with no significant reinforcement activities.

4 The Projects

4.1 Project 1: A BCD-to-Decimal Circuit

The first project is to build a combinational circuit to light up a seven-segment display to display decimal values 0 to 9. The circuit takes a four-bit input, ABCD in Figure 1 (A being the most significant bit), in Binary Coded Decimal (BCD) format. The seven outputs of the circuit, a to g, will connect to the seven segments of a seven-segment display.

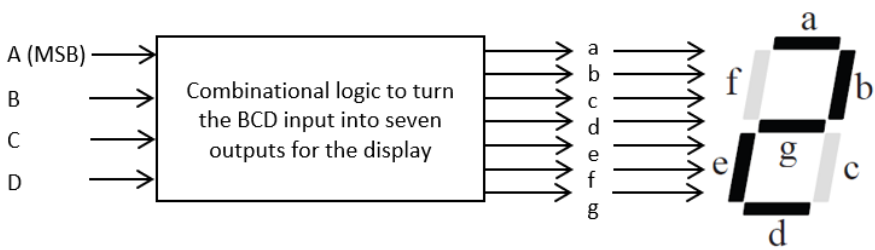


Figure 1: Block Diagram of the BCD-to-Decimal Circuit

This project originates from a Boolean expression simplification exercise in Stallings[10]. A seven-segment display can display a hexadecimal digit, 0 to 9, and A to F. In our adaptation, this project's input runs only from 0000 to 1001 for decimal digits 0 to 9. This restriction turns the last six combinations of the four-bit input, i.e., 1010 to 1111, into don't care conditions. We picked this BCD-to-decimal version as it lets students practice designing a circuit with don't care conditions. A hidden reason for this design is that it is harder to find the solution to a BCD-to-decimal circuit online than a BCD-to-hex circuit.

Students complete this project in three steps. First, students revise the truth table of a seven-segment display to turn the last six input combinations into "don't care" conditions. They then simplify the Boolean expression of each output variable. Next, students build a combinational circuit based on their step 1 result in Logisim. Finally, they connect a seven-segment display to their circuit.

This project is time-consuming, so a warning about the time needed is included at the beginning of the project requirement. To prepare students for this project, the lecture of this unit shows an example of designing and building a combinational circuit with four inputs and four outputs. The example circuit is a BCD incrementer that requires the same four-bit BCD code input as the project and increments the BCD code to generate a plus-one result. The lecture walks through the truth table with don't care conditions, simplifying the Boolean expressions for two out of the four output variables, and creating the circuit for those two outputs.

4.2 Project 2: A Counter-Driven Two-Decimal-Digit Display Circuit

After learning about sequential circuits, the second project builds a sequential circuit and feeds its output to a combinational circuit. This project is a circuit that displays decimal numbers 0 to 15 and rewinds to 0. It is composed of a sequential sub-circuit and a combinational sub-circuit. First, the project uses a counter sub-circuit (step 1 in Figure 2) to generate binary numbers 0000 to 1111 automatically. Next, the output of the counter sub-circuit is fed to a second sub-circuit (step 2 in Figure 2) to convert the binary number to decimal, but as two decimal digits in BCD format, i.e., the output would be two sets of BCD numbers. Finally, the output of the step 2 circuit connects to two Hex Digit displays. A splitter is used before each Hex display component to combine the four bits for a BCD number (four wires, one bit per wire) into a wire carrying four bits.

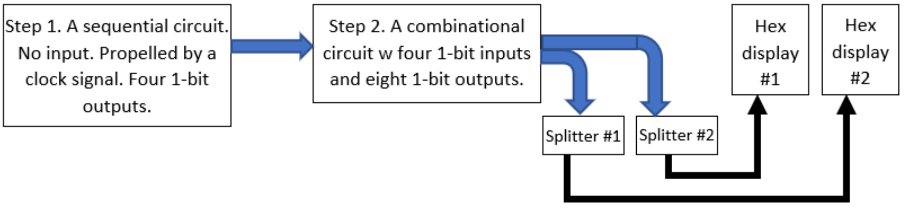


Figure 2: Block Diagram of the Counter-Driven Two-Decimal-Digit Display Circuit

This project is composed of three steps. In step 1, students build an asynchronous counter using D flip-flops based on the block diagram of a cascaded divide-by-two circuit from Tarnoff[11]. In step 2, students design and build a combinational circuit to convert a 4-bit binary number to its decimal value expressed in two sets of BCD codes. Step 2 instruction provides the truth table for such a circuit; students complete the simplification and circuit building. In step 3, students combine the step 1 and step 2 circuits by loading the step 2 circuit as a circuit library and connecting the two as Figure 2 shows. The project includes a basic explanation of splitters and directs students to the Logisim documentation for additional details.

4.3 Project 3: A Memory Reading Circuit

After working with gates and self-built circuits, students advance into building a circuit using circuit components from the software simulator. Project 3 is a revised version of a nifty assignment[8] and features a sequential circuit to simulate memory reading. This circuit will read a block of data from a memory device and display the data on a user terminal (Figure 3). The block of data will be a string of ASCII characters. During design time, students pick a string of their choice and hard-code the ASCII values into the memory device. A counter component will generate memory addresses automatically, just that students need to determine the number of bits needed for the addresses based on the length of their string. The memory device is a ROM component, and the user terminal is simulated with a TeleTYpewriter (TTY) component.

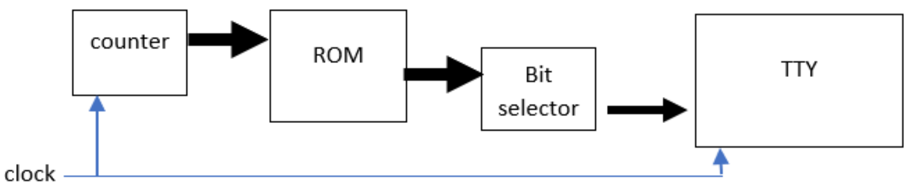


Figure 3: Block Diagram of the Memory Reading Circuit

This project is designed to be simpler compared to the first two. Its unit, unit 5 of the course, covers memory hierarchy and cache memory, and cache memory weighs more in this unit. Nevertheless, this memory reading circuit integrates data representation and interfacing memory with other components in a computer. As simple as it is, this project wraps up the discussion at the digital circuit level before the class moves into the next unit on CPU architecture.

5 Student Performance, Feedback, and Analysis

The authors tested those projects in three online sessions in 2022, with 13 students in S2 2022 (Spring 2022), 15 in F1 2022, and 11 in F2 2022. Table 1 tabulates the average and standard deviation of student scores in those projects and the course total. Those projects accounted for 15% of the course total. The same instructor taught all three sessions and graded all coursework.

Rather than course scores, a finer-grained analysis of the final exam based on core learning outcomes would be more appropriate. However, we are still trying to ascertain the right level of problems on the final, so that is not a helpful measure at the current time.

Table 1: Project and Course Total Grades From Three Online Sessions in 2022

Class Session	Project 1		Project 2		Project 3		Course Total	
	avg	stdev	avg	stdev	avg	stdev	avg	stdev
S2 2022 (13 students)	77.7%	27.4%	87.4%	19.6%	89.8%	26.1%	73.0%	14.0%
F1 2022 (15 students)	79.6%	24.9%	91.4%	10.4%	93.0%	12.0%	79.0%	14.1%
F2 2022 (11 students)	84.4%	16.4%	80.8%	21.7%	91.5%	11.6%	78.0%	14.0%

Overall, students performed well in all three projects, despite the compressed schedule. Project 1 grades were generally lower than the other two since it was the first digital logic project; students had just started on Boolean expression simplification and the simulator software. The relatively high standard deviations are due to projects that received minimal credit, which is not unusual for our 8-week courses, where students will skip assignments when they have other commitments and later just try to keep up with the rest of the course. Final course grades, which include exams, tend to be lower than project grades. This situation is also not unusual.

Students liked the hands-on approach of those projects and how the projects helped visualize the concepts. Here is a student reflection on their understanding of simplification and combinational circuits at the end of Project 1:

> I still am a little lost as to how this all works. My brain isn't creative enough to make the connections but, using Logisim is awesome because I can see what is actually happening when I click each input on of [or] off. That really helps me to visualize how the gates are working.

The projects were engaging. Students gained a deeper understanding of how digital circuits were designed and worked. Building a working circuit was satisfying:

> My understanding of simplification and combinational circuits after this project has improved. If I didn't do the K-map and discovered the simplest form, my circuit would've had so many gates compared to what I might have had! It is an amazing feeling to be able to complete and visualize a simple yet, complex circuit displaying numbers from 0 to 9.

> These projects help me understand the information in the lecture

> much more because I actually get to create the circuits and see how they behave.
>
> I have a better understanding of the bigger picture of digital circuits. This project put together all the components we have been working on the last few weeks and finally displayed a result in the TTY output. Being able to see a tangible final result really cemented my understanding of how each of these basic digital circuits come together to create an output.

At the end of the course, students even volunteered more comments on those projects in a graded asynchronous discussion. As a way to wrap up the course, the discussion in the last unit asked students to respond to one of the following four questions:

Q1. What was the hardest material from last week? What did you do that helped or didn't help? The purpose of this discussion question is for you to reflect on your work last week and share some tips (what failed, what worked, ...) with the class.

Q2. Summarize your learning in this course. Were there any topics you found more interesting than others, or topics you felt were stressed too much?

Q3. Post a question for the class to answer. Your question may be something you'd like to discuss more on or a made-up question to test your peers' understanding of a specific topic.

Q4. Post a question about the final exam that you're not clear about.

Of the 39 students from the three sessions combined, 22 took advantage of Q2 to talk about topics that interested them (Table 2). Among the 22 students who picked Q2, 18 talked about those digital logic projects. Of those 18 students, 15 cited those projects as their favorites or that helped their learning. Among the other Q2 students, two students had mixed feelings, and a third one spoke negatively as they had a hard time connecting the dots.

Students liked the projects for their hands-on nature. Two students said the projects helped visualize the abstract concepts. Seven students talked about how they gained a better or deeper understanding of the material. Nine students considered the projects "fun", "cool", "fascinating", or they "enjoyed" the projects. One student talked about "a great sense of accomplishment", and another said those projects sparked their curiosity to learn more. Creating circuits, even via a software simulator, helped visualize abstract concepts from the book. Designing and building circuits mimicking real computer components or functionalities was relevant and intriguing. Seeing a working circuit of a certain complexity from their own creation, even with issues at times, was gratifying. Here are excerpts of some of the positive comments. Only the portions relevant to those projects are included.

Table 2: Student Discussion of Those Projects in the Last Unit of the Course

Class Session	Number of students that picked Q2	Number of Q2 responses that talked about the projects	Number of remaining Q2 responses
S2 2022	7 out of 13	6 (4 positive, 2 mixed)	1 (talked about other topics covered in the course)
F1 2022	9 out of 15	7 (6 positive, 1 negative)	2 (positive about the whole course but did not list any specific topics)
F2 2022	6 out of 11	5 (all positive)	1 (talked about other topics covered in the course)

> Well, the course was a bit technical for me but I enjoyed it a lot. I learned from this course a lot, especially the digital logic projects. The most interesting topic to me was K-Maps and boolean simplification.

> I immensely enjoyed using circuits, clocks, and d-latches. Seeing and creating circuits that a computer would use to produce a counting-up counter was great.

> I had the most fun when we had to actually design the circuits, minus the part where we had to simplify the K-maps, especially unit 4 where there where [were] multiple maps for the different outputs. Then triple checking to make sure I got the grouping correct then when it came to testing the circuit realizing that there were still some mistakes in the groupings. The lesson that had to be the most fun had to be unit 5 when we converted our string of characters to hexadecimals then having the TTY display the corresponding character.

Among the less favorable comments, one student liked building circuits, but had a hard time with simplification. The second mixed-feeling comment wanted more connections between those circuits and real-world applications. The student who made the negative comment said they didn't grasp the connection between k-map (simplification) and circuits. This student may have been helped if they didn't ignore the grading comments and the instructor's offer to meet virtually since the first project.

As an initial evaluation of the efficacy of the assignments for our students, we looked at data from the Major Field Test (MFT) in Computer Science[3], an assessment instrument from ETS that we use as part of our ongoing program

Table 3: ETS Major Field Test Scores

Student Cohort	Number of Test Takers	Percent Correct – Digital Logic Question 1	Percent Correct – Digital Logic Question 2
AY2022-2023*	62	45.2	34.4
AY2021-2022	59	35.6	20.3
National**	-	64.4	41.3

* Does not include summer term.
** ETS Comparative Data population. Data from September 2015 thru June 2022.

evaluation. In the academic year 2021-22, we started offering the MFT in two of our operating systems courses, as these courses were likely to come toward the end of a student's degree program, and one of the courses uses computer architecture as a prerequisite.

Looking at the data in Table 3, there are two items on the MFT that specifically relate to digital logic/digital systems. Of the 59 students in operating systems in AY 2021-22 who tested, 35.6% answered the first question correctly and 20.3% answered the second correctly. This is in comparison to 64.4% and 41.3% of overall correct answers at the national level. Looking at the results for AY 2022-23, which is the first time students from the updated computer architecture course took the MFT, the average scores of the 62 students rose to 45.2% and 34.4%. These results are very encouraging, though still preliminary. The increases appear to be significant, especially in light of the fact that only about 1/3 of the MFT takers during AY 2022-23 had used the updated projects. Note that of the 62 students, only 4 had not taken computer architecture prior to taking an operating systems course, so we did not analyze that situation separately.

6 Conclusion

The authors built digital logic projects to engage students in an accelerated online format of their computer architecture course. Most students responded positively to the projects and gained a good amount of learning from the projects. Even though the first project is heavy, time-wise, the projects are attainable even in an accelerated eight-week online setting. While they may be easy for some students, such projects may work well with students in a moderate-level introductory computer architecture course. For an advanced class, instructors may simplify the project instructions to let students figure out some details and connections on their own.

References

[1] Charles C Bonwell and James A Eison. *Active learning: Creating excitement in the classroom. 1991 ASHE-ERIC higher education reports.* ERIC, 1991.

[2] Joint Task Force on Computing Curricula – Association for Computing Machinery (ACM) and IEEE Computer Society. *Computer Science Curricula 2013: Curriculum Guidelines for Undergraduate Degree Programs in Computer Science.* New York, NY, USA, 2013. DOI: 10.1145/2534860.

[3] ETS. *ETS Major Field Tests - Computer Science Test Description.* 2015. URL: https://www.ets.org/content/dam/ets-org/pdfs/mft/comp-sci-test-description.pdf.

[4] James Feher. "Providing a digital logic lab experience in a computer architecture course: nifty assignment". In: *Journal of Computing Sciences in Colleges* 25.5 (2010), pp. 337–341.

[5] Ville Karavirta et al. "Interactive exercises for teaching logic circuits". In: *Proceedings of the 2016 ACM Conference on Innovation and Technology in Computer Science Education.* 2016, pp. 101–105.

[6] Matias Lopez-Rosenfeld. ""Tell me and I forget, teach me and I may remember, involve me and I learn": changing the approach of teaching Computer Organization". In: *2017 IEEE/ACM 1st International Workshop on Software Engineering Curricula for Millennials (SECM).* IEEE. 2017, pp. 68–71.

[7] Jeffrey J. McConnell. "Active Learning and Its Use in Computer Science". In: *SIGCSE Bull.* 28.SI (Jan. 1996), pp. 52–54. ISSN: 0097-8418. DOI: 10.1145/237477.237526. URL: https://doi.org/10.1145/237477.237526.

[8] Bin Peng. "Building a memory reading circuit". In: *Journal of Computing Sciences in Colleges* 34.4 (2019), pp. 114–116.

[9] Michael P Rogers and Charles Hoot. "Getting ahead with a hat: reengineering a computer organization course". In: *Journal of Computing Sciences in Colleges* 34.4 (2019), pp. 42–51.

[10] William Stallings. *Computer Organization and Architecture.* 10th. New York, NY: Pearson, 2015.

[11] David Tarnoff. *Computer Organization and Design Fundamentals: Examining Computer Hardware from the Bottom to the Top.* Lulu.com, 2007.

A Stakeholder Visualization Tool Study*

Johanna Blumenthal and Richard Blumenthal
Department of Computer and Cyber Sciences
Regis University
Denver, CO 80221
{jblumenthal, rblument}@regis.edu

Abstract

Professional codes of ethics and conduct provide a convenient pedagogy for teaching students to consider social, ethical, and professional issues related to computing. Case studies offer a means for using these codes to analyze such issues, which include the importance of identifying all stakeholders impacted by a computing decision. The research in this article suggests that the introduction of a stakeholder visualization tool into a professional code's text-based case study can positively affect the number of stakeholders considered by students.

1 Introduction

Computing professionals are expected to analyze computing applications. Historically, this analysis focused on the technical capabilities of the computing solution (performance, reliability, availability, etc.). Today, there is a growing expectation that such an analysis will also include an examination of the social context in which the application resides including its impact on society. As this expectation has grown, educators have been called to teach students how to perform social impact analysis of computing applications. This requirement has been formalized in the learning outcomes of curricular recommendations for computing-related programs, such as Cybersecurity, Data Science, Software Engineering, Information Systems, and Computer Science[8, 10, 17, 18, 26].

Case studies are commonly used to demonstrate how to determine the social context and impact of a computer solution (e.g., [5, 12]). CARE is a case study analysis approach developed for use with the *The ACM Code of Ethics and Professional Conduct* (The-Code), which can be used for a social context and impact analysis, and "is designed to inspire and guide such social and ethical conduct for all computing professionals including... students"[13]. Each step in the CARE approach suggests examination of certain aspects of a presented case study via exploratory questioning, similar to the Socratic method.

Although Socratic style methodology has oft been used in the classroom as an organizing method for deeper discussion of a topic, the quality and effectiveness of the discussion in this approach relies heavily upon a facilitator to ensure that the direction of the questions and answers broadens and deepens the understanding of the topic[4]. Without such experienced guidance, are the CARE questions enough for students to engage in the deep analysis desired? If not, would a simple tool aid in deepening the questions-answer consideration?

The researchers set about to answer these questions through a pre-post style study in which students were asked to use questions from the CARE approach before and after being presented with a simple Stakeholder Visualization Tool (SVT), which is depicted in (Figure 1). The SVT is designed to prime the thinking of students in a manner that would have them consider stakeholders based upon different spheres of life (the sectors) and different proximity scopes (the levels). Three primary research questions are addressed in this article:

> **R1**: To what extent does the SVT affect the number of stakeholders considered by students in a CARE case study?
>
> **R2**: To what extent does the SVT affect the number of additional details students would seek in order to provide a greater understanding of the situational context in a CARE case study?
>
> **R3**: To what extent can students appropriately expand the SVT?

R1 and R2 directly follow from two of three questions posed in the Consider step of the CARE process, which is described in Section 3.3.

2 Background

The SVT developed by the authors in this research has been highly influenced by the Computer Science Curriculum 1991 (CS'91), literature arising from CS'91 (e.g., [14, 19, 20]), and the progression of case studies for use in applying The-Code (e.g., [5, 13]). The design of SVT and motivation for R1 and R2 has been influenced by research in educational and cognitive psychology. A summary of these influencing factors is given in the remainder of this section.

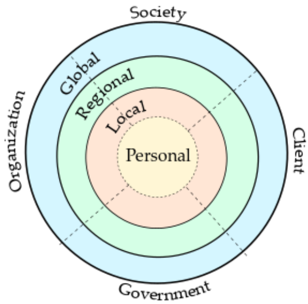


Figure 1: Stakeholder Visualization Tool

2.1 CS Curriculum

Considerations of Social, Ethical, and Professional (SEP) issues were first required in the CS'91 curricular recommendations[28], have continued through the subsequent recommendations in 2001 and 2013[9, 26], and are currently expanded upon in the proposed 2023 recommendations[16]. Accrediting agencies have also followed these recommendations[1, 27]. The introduction of SEP requirements in 1991 led to a flurry of activity focused on how to address these new SEP learning outcomes including the ImpactCS project.

In 1991, NSF funded the ImpactCS project to further develop the knowledge areas that ought to be included in the SEP curricular guidelines[14, 19, 20]. One product to come out of ImpactCS was a listing of case studies to be used in this manner[15]. ImpactCS also introduced a conceptual framework for educators to categorize computing case studies with respect to SEP issues suggesting that "every ethical and social concern occurs at a particular level of social analysis". Three dimensions of such social analysis were identified: technical, social, and ethical with a two-dimensional table visualization where rows specified Levels of Social Analysis and columns Responsibility and Ethical Issues.

The introduction of required SEP topics into the curriculum included a focus for computing students to "uphold general professional standards"[28] and led ImpactCS to note that "a careful study and application of professional codes of ethics is crucial to ethical practice in computer science. It is also important to present students with examples of authentic situations to analyze in the context of a code of professional ethics"[19]. In turn, this brought a focus on The-Code into required computer science education.

2.2 The Code

Guidelines for a *Professional Conduct in Information Processing* were first adopted by the ACM in 1996[23] with revisions in 1972 and 1992[2]. The most recent 2018 revision is The-Code version[13]. In 2018, **Ethical Principle 1.1** was amended to "a computing professional should contribute to society and to human well-being, *acknowledging that all people are stakeholders in computing*"[13]. Additionally, the CARE process along with three fictionalized case were also introduced. The case studies are intended to illustrate how The-Code can be applied as a framework for analyzing ethical dilemmas using CARE. CARE is a four-step process: (**C**onsider stakeholders and consequences, **A**nalyze how the code applies to the context, **R**eview possible actions, and **E**valuate decisions and future impact)[13]. Two questions, of three, from the Consider step are used in this article's research (see **Q1** and **Q2** in Step 4 of Section 3.2) along with the first CARE case study.

2.3 Cognitive and Education Theory

Priming is an effect where "environmental stimuli may affect subsequent responses by activating mental constructs without conscious realization"[29]. While the SVT in the current study is designed to serve as a prime, it also designed to serve as a hierarchical concept map. "Concept mapping is an active, creative, visual and spatial learning activity in which concepts are organized according to their hierarchical relationships"[24]. Together, priming and concept maps have been demonstrated to improve long-term retention of information and transfer of knowledge when solving future problems.

The Picture Superiority Effect (PSE) is a phenomenon where pictures are better recognized and recalled than text-based labels as a result of what is believed to be the duel-encoding of pictures in human memory[22]. This encoding results in pictures being more perceptually rich than text. Evidence against the PSE effect occurs when such images are only presented for brief periods of time, such as when several pictures are presented every second[11], which is not the case with the SVT presentation in this research. As a result, illustrations have been shown to be more effective than text alone for problem solving transfer[21], which requires students to solve problems that are different from those presented during instruction. In the current research, no instruction is given for problem-solving with the transfer being the answering of the CARE stakeholder and additional context questions. Naturally, the SVT plays the role of the illustration annotating the text in the case study.

Ultimately, the stakeholder identification problem can be treated as a student learning outcome. That is, students should learn to better identify all stakeholders in a problem. Research on multiple external representations (MERs) of information suggests the inclusion of the SVT, in addition to the text-based case study descriptions and questions, would positively affect stakeholder identification. Specifically, the use of multiple isomorphic external representations of the same task space has been shown to be effective in enhancing learner performance[3]. Of specific interests in the current research is MER support for making the task easier by changing its nature[30]. In the current research, the use of both text and the SVT provides a MER of potential stakeholders.

In learning environments, researchers have demonstrated that individuals also have varied learning styles (e.g., [25]). The inclusion of the SVT into the text-based case study from The-Code provides additional support for visual learners beyond the priming, concept mapping, and MERs previously introduced.

2.4 Prior Research

Preliminary results for research Question R1 were reported in a SIGCSE poster session[6]. The current work has expanded the number of subjects in the study, the stakeholders examined, and added the R2 and R3 questions.

3 Methodology

The authors designed an experiment and survey to analyze the effect of introducing the SVT into the Consider phase of the The-Code CARE process.

3.1 Participants

Twenty-three undergraduate computer science students participated in this study. The students were enrolled in one of six courses (CS1, Algorithmic Complexity, Operating Systems, Web & Database, Unix, and Computer Ethics).

3.2 Procedure

The survey was conducted using an online Learning Management System tool associated with the student's course and consisted of the following steps.

Step 1: Voluntary consent to participate in the experiment was obtained.

Step 2: Students were given Ethical Principle 1.1 from The-Code.

Step 3: Students were given the two-paragraph Malware Disruption case study from The-Code. They were not informed that the ethical principle in Step 2, nor the case study in this step, were taken from The-Code.

Step 4: With the case study still displayed, students were then asked two successive questions taken from the Consider step of the CARE process:

> **Q1**: "Who are the relevant actors and stakeholders?"
>
> **Q2**: "What additional details would provide a greater understanding of the situational context?"

These questions serve as controls for the research experiment.

Step 5: Students were then shown the SVT depicted in Figure 1 with the following explanation: "Consider the following image that suggests various communities of stakeholders effected by a computing decision, where *Personal* consists of you and your family, *Organization* consists of the organization you are working for, *Client* consists of the customers from whom you and your organization are creating a computing solution, *Government* consists of the governing bodies in which your computing solution will be created and/or deployed, and *Society* represents general societal interests in your computing solution". No other information about the SVT or how to use it was provided.

Step 6: With the SVT now displayed, students were re-asked Q1 and Q2,

> **Q3**: "Who are the relevant actors and stakeholders?"
>
> **Q4**: "What additional details would provide a greater understanding of the situational context?"

Hence, the SVT serves as the manipulation in this study/experiment.

Step 7: The subjects were subsequently asked the following successive questions. "With respect to stakeholder diagram,

> **Q5**: What levels might you add or subdivide besides Personal, Local, Regional, and Global?,
>
> **Q6**: What additional sectors might you add or subdivide besides Society, Client, Organization, and Government?
>
> **Q7**: Did you find this diagram easy to understand, why or why not?
>
> **Q8**: In the future how likely might you be to consider this stakeholder diagram, when considering all stakeholders that might be affected by an ethical computing decision?"

The responses to Q8 were restricted to a Likert-like scale of: Not Very Likely, Not Likely, Not Sure, Somewhat Likely, and Very Likely.

3.3 Coding Protocol

A coding protocol was created and independently used by the researchers to determine the number of stakeholders appearing in the Q1-Q6 answers. The protocol identifies ten "Named-Stakeholders" explicitly mentioned in the CARE case study: Generic Services, Generic's clients (web-based retail, malware, spam, and botnet), ISPs, security organizations, other-governments, and two response teams (security vendors and government). The SVT depicts five contextual sectors: Society, Client, Government, Organization, and Personal. Each sector contains three proximity levels (local, regional and global) depicted in the SVT. With respect to these sectors and levels, the protocol was conservative in identifying different stakeholders. For example, a specific answer of "Generic's Clients" or "Society" is coded only as one stakeholder, while an answer specifically enumerating multiple clients or sector levels, such as "web-based retail, malware" or "local government, regional government" is coded as two different stakeholders. For question Q3, only additional stakeholders not already mentioned in question Q1 for a given subject were counted.

With respect to what "situational context details" students would seek in Q2 and Q4, the protocol counted additional "Mentioned-Stakeholders" prompted by the SVT in Q4 that were not mentioned in Q2. For example, if "advertisers" was mentioned, it was not counted since it is not a level or sector found in the SVT. Finally, with respect to what additional SVT sectors or levels the students might add in Q5 and Q6, the answers "national" and "regional" were coded as different responses (see the Discussion section).

4 Results

To verify the stakeholder identification coding protocol accuracy, an agreement percentage inter-rater reliability assessment was performed. The two raters identified a combined total of 157 references to stakeholders in the students' answers to questions Q1 and Q3. There was a 97.5% (153/157) agreement on these references with a 100% agreement for the 101 stakeholder references in question Q1 and a 92.9% (52/56) agreement for Q3. The disagreements resulted when one rater judged two references as different stakeholders, while the other rater judged them as referring to the same stakeholder: two occurrences of "Generic Services" vs. "Organization", "governments of" vs. "governments hosting", and "global vs. society". In the following statistical tests, these disagreements were rectified by conservatively using the rater's judgment with the lower number of identified additional stakeholders in Q3. Additionally, the raters had 100% agreement on the number of additional stakeholders about which contextual details were sought by the student in the answers to Q4 versus Q2. Finally, the two raters also had a 100% agreement on the number of new sector or level additions given in the answers to questions Q5 and Q6.

Test-1: A single-sample paired difference t test[7], with null hypothesis H_0 of no difference from ten (the number of case study Named-stakeholders) was conducted to determine how many Named-stakeholders were referenced in Q1. There was a significant decrease difference for the referenced Named-Stakeholders ($M = 2.87, SD = 1.80$); $t(22) = -19.11, p < 0.01$. A large Cohen's effect size of ($D = 3.99$) was found. These results suggest the students failed to reference most of the Named-Stakeholders as actual stakeholders.

Test-2: A paired-samples difference t test[7], with null hypothesis H_0 of no increase in referenced stakeholders, was conducted to compare the number of stakeholders identified prior to the SVT presentation in Q1 and after its presentation in Q3. There was a significant increase difference for the prior Q1 ($M = 4.39, SD = 4.42$) and after Q3 ($M = 6.65, SD = 4.47$) conditions; $t(22) = 5.27, p < 0.001$. A large Cohen's effect size of ($D = 1.10$) was found. These results suggest the SVT manipulation affects an increase in the total number of stakeholders identified in the CARE case study. Additionally, 74% (17/23) of students identified additional stakeholders in their Q3 answers.

Test-3: A single-sample paired difference t test, with null hypothesis H_0 of no increase, was conducted to determine whether the SVT resulted in students suggesting additional situational context criteria in their Q4 answers. There was a significant increase difference for the additional situation criteria Q4 condition ($M = 0.57, SD = 1.24$); $t(22) = 2.19, p < 0.05$. A medium Cohen's effect size of ($D = 0.46$) was found. Since the sample size is small $N = 23$ yet statistical significance was found, the medium effect suggests the SVT affects an increase in the situational context criteria introduced in the answers to Q4.

Test-4: A single-sample paired difference t test, with a null hypothesis H_0 of no increase, was conducted to determine whether the SVT affected students referencing "Mentioned-Stakeholders" specifically depicted in the SVT in Q3. There was a significant increase difference for the after SVT condition ($M = 1.91, SD = 1.81$); $t(22) = 5.06, p < 0.01$. A large Cohen's effect size of ($D = 1.06$) was found. These results suggest that the SVT affects an increase in the Mentioned-Stakeholders considered in the case study, as depicted in the SVT. Additionally, 74% (a different 17 of 23 than in Test-1) of the students referenced "Mentioned-Stakeholders" in the SVT. Of these seventeen, there was an mean increase of 2.6 Mentioned-Stakeholders referenced.

Test-5: A single-sample paired difference t test, with a null hypothesis H_0 of no increase, was conducted to compare the combined number of additional levels and sectors suggested in the answers to questions Q5 and Q6. There was a significant increase difference for the suggested sectors and levels in the Q5-Q6 condition ($M = 1.61, SD = 1.90$); $t(22) = 4.06, p < 0.01$. A large Cohen's effect size of ($D = 0.85$) was found. These results indicate that students were able to suggest additional sectors and levels not found in the SVT. Additionally, 74% (a different 17 of 23 from Tests 1 and 3) of students suggested additional levels and sectors. These suggestions and the (multiple) number times they were suggested are: national vs. regional (7), public vs. private (2), individual, family, community, legislative vs. executive government, victims, criminal, physical environment, moral, socioeconomic, legal damages, and internet users. Furthermore, two subjects who did not suggest any additional levels or sectors indicated that the SVT adequately covered all stakeholder groups, while a third indicated the SVT didn't work even though they did reference additional stakeholders from the SVT in Q4.

Tests 1, 3, and 5 were repeated by comparing the N=12 students in a Lower Division condition, in which they survey was taken during courses taught to Freshman and Sophomores, and in an Upper Division condition given during Junior and Senior courses, referred to as *Tests- 6, 7* and *8*. No statistical difference was found for these tests. The results with the N=12 Lower Division condition listed first were:
Test-6 $M = 2.33, SD = 2.15$ vs. $M = 2.91, SD = 1.92, t(21) = 0.68, p = 0.50$;
Test-7 $M = 2.35, SD = 2.50$ vs. $M = 0.73, SD = 1.27, t(21) = 1.91, p = 0.07$;
Test-8 $M = 1.92, SD = 2.47$ vs. $M = 1.27, SD = 1.01, t(21) = 0.81, p = 0.4$.

For Q7, fourteen students indicated the SVT was easy to understand, seven it wasn't clear, and one stated it was not easy. The remaining two students indicated that the diagram was too generic. Though, another student stated that it was presented at the right level of detail. The results from how likely the students were to use the SVT again in Q8 are: Not Very Likely (4), Not Likely (1), Not Sure (8), Somewhat Likely (5), and Very Likely (5).

5 Discussion

Overall, the results of the study indicate that the SVT helps students to engage in the two CARE "Consider" questions tested in a more depthful manner by listing more stakeholders and asking for more information about additional stakeholders. Furthermore, the results are encouraging to the researchers in so far as most students were able to use the SVT as intended without any detailed instructions as to how it was designed to be used. The researchers believe that additional instruction may increase the effects found in these results.

Interestingly, most of the students struggled in the control question to list all of the explicitly Named-Stakeholders by only naming an average of 2.9 out of the 10 stakeholders (Test-1). This may indicate a general lack of familiarity among these students with stakeholder and issue spotting in a written scenario. After being exposed to the SVT, most students did list additional stakeholders that they had not previously listed in Q1. One student even explicitly stated that the SVT was subjectively "not helpful", while still listing additional stakeholders after seeing it. Of those who did reference additional stakeholders, the average number of stakeholders added was 2.26 (Test-2).

Although the particular case study chosen for this experiment came directly from The-Code, it contains Named-Stakeholders and synonyms that are depicted in the SVT: organization, client(s), and government(s). This may indicate that "having seen these words" or priming alone is not the only source of the effect described in the results. The researchers are curious to repeat the experiment with a different case study that does not explicitly use stakeholders depicted in the SVT to see if the effect size is greater after seeing the SVT than when stakeholders are explicitly stated in a less direct manner.

When comparing Q2 vs. Q4 answers, which called upon students to state "what additional contextual details were important for analyzing the case scenario", the researchers only focused on additional information related to stakeholders sought by the student in Q4 that was not stated as sought in Q2 and that was directly related to the SVT, as determined by labels depicted in the SVT, as well as agreed upon protocol synonyms for such words. The researchers wanted to know if the SVT prompted the students to seek additional information expressly about stakeholder groups alluded to in the SVT. The results suggest that students did, on average, seek more information about stakeholders alluded to in the SVT. (Test-3).

In addition to analyzing the total number of stakeholders listed in Q3 vs. Q1, the researchers also coded the number of Mentioned-Stakeholders. The number of Mentioned-Stakeholders increased by an average of 1.91 after the students were exposed to the SVT. Even some students who did not have an additional number of stakeholders when comparing Q1 and Q3, changed the way they worded their responses in order to align with the language of the SVT, which indicates they were influenced by the SVT.

Although the authors purposefully did not give explicit instructions on how students should use the SVT to come up with additional stakeholders in the case study, most students appeared to understand the general intent of the SVT. This is indicated by the increase in stakeholders referenced, which is discussed above (Test-2) as well as the result that most students were able to appropriately extend the stakeholder diagram in a manner aligned with its general intended approach (Test-5). Instruction on how to use the SVT to identify stakeholders, including an example using a case study, may increase the effect provided by the SVT. Although most students understood the general purposes of the SVT, with fourteen stating it was clear or easy to use, seven students expressed some confusion about the SVT, parts of the SVT, or how to apply it to the case study. The authors assume that explicit instructions would help reduce this confusion.

Student responses indicate that there may be some confusion between the SVT Client and Organization sectors (however this confusion also exists in Software Engineering discussions of customer, client and organization, where these "roles" are often filled by the same stakeholder but can have different stakeholders too). Responses also indicate that some students were unsure about the "Personal" sector/level in the center of the SVT and how to use that, while several appropriately referenced and extended the Personal sector/level. Clarification of the Personal aspect of the SVT in future research should prove helpful, especially since this level was included to increase student engagement with the SVT.

The comparison of the Lower to Upper Division conditions suggests that educational maturity isn't a factor in identifying stakeholders. The Upper Division group included three students who were enrolled in an ethics and social good computing course. It would be interesting in future research to collect more data to determine whether such a course affected stakeholder identification versus use of the SVT.

There is a concern that the small sample size of N=23 might be biasing the data. This is especially true of the comparison of the Lower and Upper Division conditions in Tests 6-8, since the sample size was twelve and eleven students, respectively. Although somewhat of an aside, in future research, the authors would strongly consider some type of non-coercive incentive for students to participate in such research, such as offering gift-card for a free coffee drink or equivalent.

There are several avenues of future research the authors are interested in exploring. The research cited in the Background section suggests that the SVT may also increase the long-term benefits of using such a diagram. For example, after using the SVT multiple times, a type of automaticity may take place in which the SVT is integrated in the reasoning process when identifying stakeholders such that it no longer needs to be explicitly shown. The authors are also interested in exploring variations of the SVT and other visual tools. Finally, the SVT focused on one ethical principal from The-Code and one step in The CARE process. It is likely that similar visualization tools could help with other principles in The-Code and steps in the The CARE approach to examining ethical computing case studies.

In conclusion, the results of the experiment suggest that fairly simple visual tools can be used to help students engage in deeper analysis when considering all of the potentially relevant stakeholders in case studies with the SVT having a positive affect for research questions R1-R3.

6 Acknowledgements

The authors would like to thank the anonymous reviewers for their feedback, the Internal Review Board of Regis University, and the students who volunteered to participate in this survey.

References

[1] ABET. Criteria for accrediting computing programs. https://www.abet.org/accreditation/accreditation-criteria/criteria-for-accrediting-computing-programs-2021-2022/.

[2] ACM. Acm code of professional conduct (1972), 1972. https://ethics.acm.org/code-of-ethics/previous-versions/1972-acm-code/.

[3] Sharon Ainsworth. Deft: A conceptual framework for learning with multiple representations. *Learning and Instruction*, 16:183–198, 2006.

[4] Hanna M. Altorf. Dialogue and discussion: Reflections on a socratic method. *Arts & Humanities in Higher Education*, 18(1):60–75, 2019.

[5] Ronald E. Anderson, Deborah G. Johnson, Donald Gotterbarn, and Judith Parrolle. Using the new acm code of ethics in decision making. *Communications of the ACM*, 36(2):98–106, 1993.

[6] Richard Blumenthal and Johanna Blumenthal. Consider visualizing society within the acm code of ethics [poster]. In *Proceedings of 51st ACM Technical Symposium on Computer Science Education*, page 1292, 2020.

[7] Charles H. Brase and Corrinne P. Brase. *Understandable Statistics: Concepts and Methods*. Houghton Mifflin Company, Boston, MA, USA, 2003.

[8] Diana L. Burley and Matt Bishop. Cybersecurity curricula 2017, 2017. https://www.acm.org/binaries/content/assets/education/curricula-recommendations/csec2017.pdf.

[9] James H. Cross, Gerald Engel, Eric Roberts, and Russell Shackelford. Computing curricula 2001 computer science, 2001. https://www.acm.org/binaries/content/assets/education/curricula-recommendations/cc2001.pdf.

[10] Andrea Danyluk and Paul Leidig. Computing competencies for undergraduate data science curricula, 2021. https://www.acm.org/binaries/content/assets/education/curricula-recommendations/dstf_ccdsc2021.pdf.

[11] P Fraisse. Motor and verbal reaction times to words and drawings. *Psychonomic Science*, 12:235–236, 1968.

[12] Damian Gordon, Michael Collins, and Dympna O'Sullivan. The development of teaching case studies to explore ethical issues associated with computer programming. In *UKICER*, pages 1–9, 2021.

[13] Don Gotterbarn and Marty J. Wolf. The code: The acm code of ethics and professional conduct, 2018. https://www.acm.org/code-of-ethics.

[14] Chuck Huff and C. Diane Martin. Computing consequences: A framework for teaching ethical computing. *CACM*, 38(12):75–84, 1995.

[15] ImpactCS. Impactcs (impact computer science), 1995. http://computingcases.org/general_tools/curriculum/impactcs.html.

[16] Amruth N. Kumar and Rajendra K. Raj. Cs202x acm/ieee-cs/aaai comuter science curricula, 2023. https://csed.hosting.acm.org/.

[17] Rich LeBlanc and Ann Sobel. Software engineering 2014: Curriculum guidelines for undergraduate degree programs in software engineering, 2014. https://www.acm.org/binaries/content/assets/education/curricula-recommendations/is2020.pdf.

[18] Paul Leidig and Hannu Salmela. Is202: A competency model for undergraduate programs in information systems, 2014. https://www.acm.org/binaries/content/assets/education/se2014.pdf.

[19] C. Diane Martin, Chuck Huff, Don Gotterbarn, and Keith Miller. Implementinag a tenth strand in the cs curriculum. *CACM*, 39(12):75–84, 1996.

[20] C. Diane Martin and Elaine Y. Weltz. From awareness to action: Integrating ethics and social responsibility into the computer science curriculum. *ACM Computers and Society*, 29(2):6–14, 1999.

[21] Richard E. Mayer, Kathryn Steinhoff, Gregory Bower, and Rebecca Mars. A genertative theory of textbook design: Using annotated illustrations to foster meaningful learning of science text. *Educationa Technology Research and Development*, 43:31–41, 1995.

[22] Allan Paivio. *Images in Mind: The Evolution of a Theory*. Harvester Wheatsheaf, Birmingham, United Kingdom, 1991.

[23] Donn B. Parker. Rules of ethics in information processing. *CACM*, 11(3):198–201, 1968. https://ethics.acm.org/code-of-ethics/previous-versions/1966-acm-code/.

[24] Angelo J. Pintoi and Howard J. Zeitz. Concept mapping: A strategy for promoting meaningful learning in medical education. *Medical Teacher*, 19(2):114–121, 1997.

[25] Frank Romanelli, Eleanora Bird, and Melody Ryan. Learning styles: A review of theory, application, and best practices. *American Journal of Pharmaceutical Education*, 73(1), 2009. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2690881/.

[26] Mehran Sahami and Steve Roach. Computer science curricular 2013. https://www.acm.org/binaries/content/assets/education/cs2013_web_final.pdf.

[27] Seoul Accord. The seoul accord. http://www.seoulaccord.org/.

[28] Alan B. Tucker and Bruce H. Barnes. Computing curricula 1991: Report of the acm/ieee-cs joint curriculum task force. *CACM*, 34(6):68–84, 1991.

[29] Evan Weingarten, Qijia Chen, Maxeall McAdams, Jessica Yi, Justin Hepler, and Dolores Albarracin. From primed concepts to action: A meta-analysis of the behavior effects of incidentally-presented words. *Psychological Bulletin*, 142(5):472–497, 2015.

[30] Jiajie Zhang and Donald A. Norman. Representations in distributed cognitive tasks. *Cognitive Science*, 18:87–122, 1994.

ML Production Systems Course at a Polytechnic PUI*

Ronald J. Nowling
Electrical Engineering and Computer Science
Milwaukee School of Engineering
Milwaukee, WI 53202
nowling@msoe.edu

Abstract

Machine learning-powered web services have entered the mainstream and related experience is highly valued in industry. A course titled "ML Production Systems" on the implementation and operation of software services that incorporate machine learning is described. The course was designed around a term-long group project to implement a spam classification service. The project was supported by lectures in topics such as RESTful services, data storage systems, and data processing systems along with machine learning on time series data. The design of the class, a reflection from its first offering at a polytechnic primarily undergraduate institution (PUI), and the availability of open-source course materials are described. With few courses like it currently available, this work aims to stimulate the proliferation of similar courses at other colleges and universities.

1 Introduction

Machine learning (ML) plays an integral role in data-driven, user-serving applications such as online advertising[22, 4, 26, 11, 24], social media platforms[11, 21, 20], streaming media[6, 3, 18], and e-commerce[29, 1]. Developing machine learning-powered services requires knowledge across a wide range of technologies and practices. The development and training of machine learning models needs to be adjusted to handle data sets that change in time. Architects must be familiar with the tradeoffs, in terms of latency and throughput of storage and data processing components, in order to meet performance requirements and consider failure modes of distributed systems to meet reliability requirements. Developers and software reliability engineers (SREs) need to adapt practices and processes for software deployment, operation, and monitoring, often referred to as developer operations (DevOps), to the nuances of machine learning. ML production systems is a growing yet critical area of study needed to realize the full potential of machine learning for many applications.

Although around for a decade or longer, the practice and knowledge have primarily been driven by a small number of companies. Companies such as Google[24, 33, 27, 2], Meta[11], Uber[7, 31], and LinkedIn[8, 28, 17] developed and described critical technologies and architectural approaches that were later adopted by the wider industry and published highly-cited experience reports. Industry is beginning to converge on terminology, concepts, knowledge, and practices, enabling the development of educational resources. At least eight books[5, 10, 32, 9, 19, 15, 30, 13] and self-study materials[25] have been published since 2018. This knowledge has started to translate into the classroom as evidenced by the recent development of two courses[14, 16]. Students will expect access to similar coursework to support industry careers, and faculty will be tasked with developing such courses.

This paper describes the content and materials of a new, group-based, project-focused elective course titled "ML Production Systems" and reflects on the experience of teaching the course to senior undergraduate students at a polytechnic, primarily undergraduate institution (PUI). The core of the course is an implementation of an end-to-end spam classification system, using popular, freely-available, open-source software. The project was supported by lectures, example code, and tutorials. The described materials are released under open-source licenses for use by other instructors.

2 Description of the Course

In a 10-week quarter course, students learned to design, implement, deploy, and monitor ML production systems. The course was organized around a five-part end-to-end spam classification system project. An architectural diagram of the system is presented in Figure 1.

Students were given a script that simulates users receiving and interacting with emails and evaluates predictions against known labels using the trec07p spam classification data set [23] and hand-picked philosophy and history texts

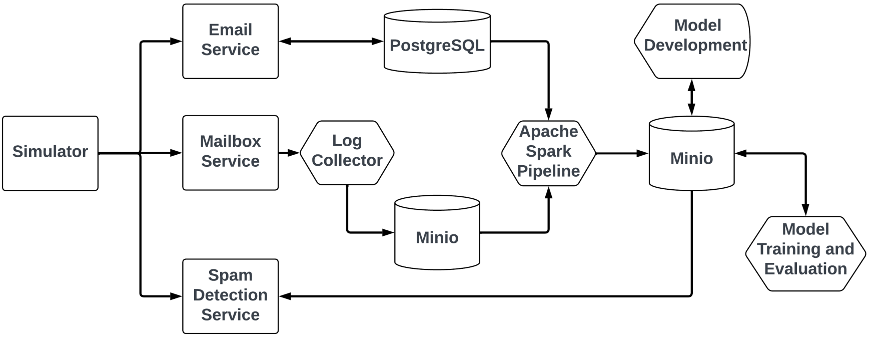


Figure 1: **Architecture of the Spam Classification System.** Square boxes indicate services, cylinders indicate databases, and hexagons indicate pipelines. The model development component represents an interactive process performed by a user. Arrows indicate the direction of data movement.

from Project Gutenberg as input. Students used the script to populate their system, test the prediction service, and experiment with data distribution shift and model retraining. The project was supplemented with code examples and handouts. Students were given two weeks to complete each part and asked to demo their working solutions to the instructor.

One of the primary goals of the course was to expose students to technology commonly used in industry. To accomplish this, the course and project were designed using only freely-available, open-source software. PostgreSQL was used as the relational database, and Minio was used as the object store. Python was chosen as the primary programming language primarily due to the availability of the scikit-learn library for machine learning. Jupyter notebooks, Pandas, and matplotlib were used for model development. Additional Python libraries used include the Flask web framework, marshmallow for validation, psycopg2 for accessing PostgreSQL, and boto3 for accessing an object store. Lastly, the data pipeline was implemented using Apache Spark and Scala. Due to concerns about costs and complexity, it was decided to focus on local development (at least initially), rather than using a cloud provider. Therefore, the university assigned every student a personal laptop with a moderately powerful CPU, 16 GB RAM, and a 256 GB SSD configured with Microsoft Windows 10.

The project was supported by two lectures per week. The lectures covered topics such as implementing RESTful services using the Flask framework, different types of database and storage services, data processing infrastructure, development and evaluation of machine learning models, continuous delivery

(CD) of machine learning models (so-called MLOps), and monitoring of the system.

2.1 Submissions and Feedback

Projects were graded primarily through student demonstrations. Students were asked to store their project code in private GitHub repositories and share the repositories with the instructor. For archival purposes, students were asked to submit links to the repositories to the learning management system (LMS).

Through these demonstrations, given during lab time, the instructor primarily assessed technical correctness. This allowed the instructor to identify and provide feedback on any errors that could impact later projects. If errors were encountered during the demo, the students were asked to make changes and demo again. Grades were assigned based on the successful demos. Suggested due dates were provided to students but flexibility was provided if students needed more or less time for projects.

2.2 Availability of Materials

The course materials are available under the open-source Creative Commons Attribution-ShareAlike 4.0 International (CC BY-SA 4.0) and Apache Software License v2.0 licenses and archived in the `cs4981-2023q3` branch of the GitHub repository located at:
`https://github.com/msoe-dise-project/ml-prod-sys-course`
The course and its materials are initatives of the Milwaukee School of Engineering Data-Intensive Systems Education (MSOE DISE) project (`https://msoe-dise-project.github.io`) led by the author.

3 Student and Instructor Experiences

Two sections of the course were offered by a single instructor in the 2022-2023 Winter quarter. Thirty-six Computer Science (CS) and twelve Software Engineering (SE) majors (48 total) with senior standing enrolled. All students had taken required courses in full-stack web applications, database systems, and software engineering tools (e.g., version control with git). The CS majors had taken additional required courses in machine learning and data science, including a junior practicum, while the SE majors had taken courses such as software architecture, software verification, and a junior practicum experience. Students organized themselves into 14 groups and ensured that each group included at least one CS major to support the machine learning-specific aspects of the course project.

Student learning was assessed primarily based on satisfaction of the technical project requirements. 46 of the 48 students successfully completed the course, meaning that the students' groups submitted work for and earned passing grades on all five projects. The two students who were not successful did not attend class or submit any work.

Students were asked to describe what they learned, what went well in the class, and what could be improved as part of their final project reports. Textual analysis was performed using a two-pass coding approach[12]. In the first pass, the feedback was reviewed to identify topics. The feedback was reviewed in a second pass to count the number of groups explicitly mentioning each topic with a positive or negative sentiment (see Table 1).

Table 1: Topics and number of mentions in student feedback.

	Positive Mentions	Negative Mentions	Not Mentioned
Overall Experience	10	0	4
ML outside of notebooks	9	0	5
Learned new technologies	12	0	2
Review/integration of previous classes	9	0	5
Environment setup/debugging	1	6	7
Mixed-skills team experience	5	0	9
Preparation for industry roles	4	0	10
Valuable addition to curriculum	4	0	10

Ten of the fourteen groups described their overall experience as positive; none of the groups described their overall experience as negative. The most commonly mentioned contributors to the positive experiences including seeing how ML is used outside of notebook environments (9 positive mentions), learning new technologies (12 positive mentions), review and integration of technologies and concepts learned in previous classes (9 positive mentions), and the experience of working in mixed CS/SE teams (5 mentions). Lastly, four groups described the class as good preparation for industry roles and suggested the class should be a required course for both majors. The only negative mention was of challenges with setting up and debugging the environments by six groups.

4 Discussion

Despite the overall positive experience of the students and instructor, there are multiple opportunities to improve the course. In its current 10-week quarter format, there was no time to include a number of relevant topics. For example, production systems are now frequently designed around event-driven architectures and implemented using streaming technologies. Streaming systems have unique considerations in terms of implementing data analyses on unbounded streams, required computational resources, and how to handle failure. Companies have also adopted experiment trackers, like MLflow, to facilitate reproducibility in ongoing efforts by data scientists to improve model performance. Lastly, feature stores were recently introduced as a new type of service that wraps high-throughput and low-latency data stores into a single API to enable centralization and reuse of feature calculations.

This course is one of the few places in the curriculum where the CS majors were exposed to topics in software engineering, such as software architecture, developing requirements, documentation practices, and software testing. There are substantial opportunities to improve coverage of these topics in the course. For example, the students submitted their work by sharing private GitHub repositories with the instructor, but none of the student groups documented their projects. This means the students did not follow the standard practice of creating `README.md` Markdown files that summarize the purpose of the software in the repository and provide instructions for installing dependencies and running the software. It would have been even better if students had documented dependencies on other services and data, any interfaces or data exported by the system, and the internal operation of the component (including any relational data or file schemas). The students did not appear to implement any formal code review process (e.g., creating pull requests and asking for code reviews), although several groups reported practicing peer programming in their reflections. Lastly, there was no evidence that students implemented any manner of automated tests or continuous integration for their repositories. Given the importance and widespread usage of these practices in industry, their incorporation would better prepare students for industry roles.

From the assessments of the group projects, it was not possible to determine if there were variations in student learning or contributions to the projects. Assessments of individual students such as exams or quizzes would be beneficial for assessing the effectiveness of student learning from the lectures. Similarly, students could be asked to reflect on their contributions and that of group members.

Challenges with setting up the environment and installing dependencies was the only topic with significant negative mentions by multiple student groups in the final reports. In three cases, the instructor created and shared additional instructions with the class; these instructions were incorporated into the assignment descriptions for use by future groups. Nonetheless, the effort needed to set up and debug environments reflective of tasks commonly performed by industry software engineers. The instructor considered the required work and effort spent debugging environment issues to be good preparation for industry roles and was hesitant to simplify that component of the course.

5 Conclusion

In this paper, a course focused on the implementation and operation of ML-powered services was described. This course is not the first of its type but, to the author's knowledge, only the third such course to be described. Maybe most importantly, this course was successfully implemented in a substantially different environment from those of previous courses. The other courses were taught at very selective institutions, Carnegie Mellon University (CMU)[16] and Stanford University[14]. Further, while the CMU course[16] was targeted at graduate students, both the Stanford course[14] and the course described here were taught to senior-level undergraduate students. The author's institution is a regional primarily-undergraduate institution (PUI) that is selective, but considerably less so than either CMU or Stanford. Nonetheless, as a polytechnic, students are required to take a significant number of computer science and software engineering courses, which, as seniors, the students had already completed most of. Future work should evaluate the feasibility of offering a similar course at a liberal arts college or comprehensive regional institution with fewer required courses and/or a mix of junior and senior students.

Few universities offer such courses currently, but there are good reasons to believe that these types of courses will become more common in the next few years. The author's goal in sharing the course materials and describing student experiences is to encourage the development of similar courses at a wide-range of universities and colleges. As the number of available courses increases, there will be further opportunity for dialogue on technical best practices for implementing such systems, effective teaching practices, and ways to interface with related fields such as software engineering and algorithmic fairness.

References

[1] Xavier Amatriain and Justin Basilico. "Recommender Systems in Industry: A Netflix Case Study". In: *Recommender Systems Handbook*. Ed. by Francesco Ricci, Lior Rokach, and Bracha Shapira. Boston, MA: Springer US, 2015, pp. 385–419.

[2] Eric Breck et al. "The ML test score: A rubric for ML production readiness and technical debt reduction". In: *2017 IEEE International Conference on Big Data (Big Data)*. Dec. 2017, pp. 1123–1132.

[3] Oscar Celma. "The Exploit-Explore Dilemma in Music Recommendation". In: *Proceedings of the 10th ACM Conference on Recommender Systems*. RecSys '16. Boston, Massachusetts, USA: Association for Computing Machinery, Sept. 2016, p. 377.

[4] Deepayan Chakrabarti, Deepak Agarwal, and Vanja Josifovski. "Contextual advertising by combining relevance with click feedback". In: *Proceedings of the 17th international conference on World Wide Web*. WWW '08. Beijing, China: Association for Computing Machinery, Apr. 2008, pp. 417–426.

[5] Cathy Chen et al. *Reliable Machine Learning: Applying SRE Principles to ML in Production*. en. 1st ed. O'Reilly Media, Oct. 2022.

[6] Gideon Dror et al. "The Yahoo! Music Dataset and KDD-Cup'11". In: *Proceedings of KDD Cup 2011*. Ed. by Gideon Dror, Yehuda Koren, and Markus Weimer. Vol. 18. Proceedings of Machine Learning Research. PMLR, Aug. 2012, pp. 3–18.

[7] Yupeng Fu and Chinmay Soman. "Real-time Data Infrastructure at Uber". In: *Proceedings of the 2021 International Conference on Management of Data*. SIGMOD '21. Virtual Event, China: Association for Computing Machinery, June 2021, pp. 2503–2516.

[8] Sahin Cem Geyik et al. "Talent Search and Recommendation Systems at LinkedIn: Practical Challenges and Lessons Learned". In: *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*. SIGIR '18. Ann Arbor, MI, USA: Association for Computing Machinery, June 2018, pp. 1353–1354.

[9] Noah Gift and Alfredo Deza. *Practical MLOps: Operationalizing Machine Learning Models*. en. 1st ed. O'Reilly Media, Oct. 2021.

[10] Yaron Haviv and Noah Gift. *Implementing MLOps in the Enterprise: A Production-First Approach*. en. 1st ed. O'Reilly Media, Nov. 2023.

[11] Xinran He et al. “Practical Lessons from Predicting Clicks on Ads at Facebook”. In: *Proceedings of the Eighth International Workshop on Data Mining for Online Advertising*. ADKDD’14. New York, NY, USA: Association for Computing Machinery, Aug. 2014, pp. 1–9.

[12] Monique Hennink, Inge Hutter, and Ajay Bailey. *Qualitative Research Methods*. en. Second edition. SAGE Publications Ltd, Feb. 2020.

[13] Geoff Hulten. *Building Intelligent Systems: A Guide to Machine Learning Engineering*. en. 1st ed. Apress, Mar. 2018.

[14] Chip Huyen. *CS 329S*. en. `https://stanford-cs329s.github.io/`. Accessed: 2023-6-4. 2022.

[15] Chip Huyen. *Designing Machine Learning Systems: An Iterative Process for Production-Ready Applications*. en. 1st ed. O’Reilly Media, June 2022.

[16] Christian Kästner and Eunsuk Kang. “Teaching software engineering for AI-enabled systems”. In: *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering: Software Engineering Education and Training*. ICSE-SEET ’20. Seoul, South Korea: Association for Computing Machinery, Sept. 2020, pp. 45–48.

[17] Krishnaram Kenthapadi, Benjamin Le, and Ganesh Venkataraman. “Personalized Job Recommendation System at LinkedIn: Practical Challenges and Lessons Learned”. In: *Proceedings of the Eleventh ACM Conference on Recommender Systems*. RecSys ’17. Como, Italy: Association for Computing Machinery, Aug. 2017, pp. 346–347.

[18] Noam Koenigstein, Gideon Dror, and Yehuda Koren. “Yahoo! music recommendations: modeling music ratings with temporal dynamics and item taxonomy”. In: *Proceedings of the fifth ACM conference on Recommender systems*. RecSys ’11. Chicago, Illinois, USA: Association for Computing Machinery, Oct. 2011, pp. 165–172.

[19] Valliappa Lakshmanan, Sara Robinson, and Michael Munn. *Machine Learning Design Patterns: Solutions to Common Challenges in Data Preparation, Model Building, and MLOps*. en. 1st ed. O’Reilly Media, Nov. 2020.

[20] Jimmy Lin and Alek Kolcz. “Large-scale machine learning at Twitter”. In: *Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data*. SIGMOD ’12. Scottsdale, Arizona, USA: Association for Computing Machinery, May 2012, pp. 793–804.

[21] Zhuoran Liu et al. “Monolith: Real Time Recommendation System With Collisionless Embedding Table”. In: (Sept. 2022). arXiv: `2209.07663 [cs.IR]`.

[22] Quan Lu et al. "A Practical Framework of Conversion Rate Prediction for Online Display Advertising". In: *Proceedings of the ADKDD'17*. ADKDD'17 Article 9. Halifax, NS, Canada: Association for Computing Machinery, Aug. 2017, pp. 1–9.

[23] Gordon Lynam and Cormack Thomas. *TREC 2007 Public Corpus*. `https://plg.uwaterloo.ca/~gvcormac/treccorpus07/about.html`. Accessed: 2023-6-4. 2007.

[24] H Brendan McMahan et al. "Ad click prediction: a view from the trenches". In: *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*. KDD '13. Chicago, Illinois, USA: Association for Computing Machinery, Aug. 2013, pp. 1222–1230.

[25] Goku Mohandas. *Home - Made With ML*. `https://madewithml.com/`. Accessed: 2023-6-4.

[26] Matthew Richardson, Ewa Dominowska, and Robert Ragno. "Predicting clicks: estimating the click-through rate for new ads". In: *Proceedings of the 16th international conference on World Wide Web*. WWW '07. Banff, Alberta, Canada: Association for Computing Machinery, May 2007, pp. 521–530.

[27] David Sculley et al. "Hidden technical debt in machine learning systems". In: *Adv. Neural Inf. Process. Syst.* 28 (2015).

[28] Amit Sharma and Baoshi Yan. "Pairwise learning in recommendation: experiments with community recommendation on LinkedIn". In: *Proceedings of the 7th ACM conference on Recommender systems*. RecSys '13. Hong Kong, China: Association for Computing Machinery, Oct. 2013, pp. 193–200.

[29] Brent Smith and Greg Linden. "Two Decades of Recommender Systems at Amazon.com". In: *IEEE Internet Comput.* 21.3 (May 2017), pp. 12–18.

[30] Jeff Smith. *Machine Learning Systems: Designs that scale*. en. First Edition. Manning, July 2018.

[31] Chong Sun, Nader Azari, and Chintan Turakhia. "Gallery: A machine learning model management system at Uber". In: *Proceedings of the 22nd International Conference on Extending Database Technology (EDBT)*. OpenProceedings.org, 2020.

[32] Mark Treveil et al. *Introducing MLOps: How to Scale Machine Learning in the Enterprise*. en. 1st ed. O'Reilly Media, Jan. 2021.

[33] Martin Zinkevich. "Rules of machine learning: Best practices for ML engineering". In: *https://developers.google.com/machine-learning/guides/rules-of-ml* (2017).

The Structure of a Graduate Defensive Cybersecurity Course*

Mohamed Lotfy
Utah Valley University
Orem, UT 84058
MohamedL@uvu.edu

Abstract

The growing need for cybersecurity professionals is driven by the drastic increase of advanced persistent threat cyber-attacks on critical infrastructure and ransomware attacks. There are still mismatches between industry needs and cybersecurity education. To prepare IT and cybersecurity graduates and meet industry needs, cybersecurity courses must introduce current offensive and defensive tools and practices to secure computing resources, systems, services, data, and network services. These offensive and defensive cybersecurity tools should be introduced and applied in hands-on activities, thus allowing students to gain the needed knowledge of current cybersecurity best practices. In this paper, we provide the structure, components, hands-on assignments, and virtual environment of a graduate defensive cybersecurity course designed to introduce the cyber kill chain model and the DoD information operations 6D doctrine. Student course evaluations and what helped them to learn the most are presented and discussed.

1 Introduction

The need for cybersecurity professionals has been growing drastically in the last decade and is continuing to grow. According to the Cyberseek.org cybersecurity supply/demand heat map, the total national employed cybersecurity work force in 2023 is 1,129,659. The total number of national cybersecurity openings in 2023 at the time of this writing is 663,434[4]. The 2023 eSENTIRE official cybersecurity jobs report mentioned that "according to Cybersecurity Ventures,there will be 3.5 million unfilled jobs in the cybersecurity industry through 2025"[5]. The growing need for cybersecurity professionals is justified due to the exponential increase in critical infrastructure and ransomware attacks. The Center for Strategic & International Studies recorded more than 950 worldwide significant cyber-attacks on government agencies, defense and high-tech companies, or economic crimes with losses of more than a million dollars since 2006[6].

Most of the significant cyber-attacks are advanced persistent threats (APT). APT are clandestine, prolonged, and continuous hacking processes conducted by cyber-criminals and nation-state/sponsored actors to infiltrate a targeted entity for specific gains[11]. APT conducted by nation-state/sponsored actors tends to be more sophisticated, utilize more resources, and are not financially driven like APT conducted by cybercriminals[2]. For example the 2019-2020 SolarWinds APT hack impacted the global supply chain and affected thousands of organizations, including the U.S. government[18]. The 2021 Colonial pipeline attack impacted the U.S. oil and gas supply chain in the south east and impacted many U.S. residents and organizations[10]. APT and cybercrime damages cost is expected to reach $10.5 trillion by 2025[15].

Nowadays, "every IT position is also a cybersecurity position"[5]. IT workers are currently involved, at some level, securing applications, systems, servers, networking and cloud infrastructure, data at rest or in motion, devices, and people[5]. However, there are still mismatches between industry needs and cybersecurity education[3]. To prepare IT and cybersecurity graduates and meet industry needs, IT and cybersecurity courses must introduce current offensive and defensive tools and practices to secure computing resources, systems, services, data, and network services. These current offensive and defensive cybersecurity tools should be introduced and applied in hands-on activities, thus allowing students to gain the needed knowledge of current cybersecurity best practices. Connecting the systems and data networks knowledge with current cybersecurity best practices within a specific deployment context enables students to develop and gain current professional cybersecurity competency.

In the remaining sections, a review of the structure of current cybersecurity graduate courses will be conducted and the cyber kill chain model will be introduced. Then the structure of a graduate Advanced Network Defense and Countermeasures course, as well as the course assignments, the used tools, and the virtual environment infrastructure are presented. Student evaluation and feedback of the course are shared followed by a discussion on the findings and lessons learned from offering the course.

2 Graduate Cybersecurity Courses Review

In recent years, teaching offensive and defensive cybersecurity in computer science, cybersecurity (CSEC), and information technology (IT) programs became central in cybersecurity education. Offensive security, through penetration testing/ethical hacking courses, allows students to gain the needed knowledge and skills of how to attack and access systems and networks using current tools[8, 13]. Connecting the systems and data networks knowledge with how to perform defensive security skills within a specific context enables students to develop and gain needed cybersecurity competency[13]. In addition, the offensive and defensive hands-on assignments let students develop the needed cybersecurity capabilities thus enabling them later to build layered defenses that harden the systems to intrusion.

According to CC2020 curricula guidelines, CSEC and IT competency-based programs need to be organized around the knowledge, skills, and dispositions dimensions to enable student's career readiness. For IT and CSEC graduates to be job ready, the educational course work should mirror the "computing technologies in the work environment"[7]. IT and CSEC program courses need to expose students to current offensive and defensive methodologies, including network port scanning, vulnerability assessment, exploitation, password harvesting and cracking, server hardening, firewall iptables rules, network and host intrusion systems, and honeypot deployment.

Educators use different approaches to teach cybersecurity courses. Some courses are designed using the Committee on National Security Systems proposed reference framework. Other courses follow curricular guidelines based on industry guidelines, for example ABET. The third approach follows the Department of Homeland Security (DHS) educational guidelines[16]. To allow students to acquire the needed hands-on skills, labs are required regardless of the approach. Some courses in graduate cybersecurity programs allow students to acquire and apply the needed offensive and defensive cybersecurity skills in a legal security education (SEED) lab[12]. Other courses use on-site local network of computers, virtual machines (VMs) installed on students' personal laptops, cloud deployed computing environments (labs as a service), and cyber ranges lab infrastructures.

The computer system security (CSS) graduate course discussed in [1] was designed to allow students to learn and practice security concepts, such as authentication, access control, auditing, system hardening and data protection, to ensure the confidentiality, integrity, and availability (CIA) elements in information systems[1]. The hands-on labs were conducted using containers deployed to create an instance of the system, thus allowing students to run vulnerable applications in an isolated environment on the students' host machines[1].

In Moldovan and Ghergulescu[14], the authors described how they delivered the network security and penetration testing module, which is part of a graduate cybersecurity program, using two different approaches. In the first cohort, timed in-class practical tests with a fixed set of questions were used. In the second cohort, students were placed in groups. Each student group used virtual labs to conduct penetration testing, drafted a report, and presented the work in front of the class. The end of module feedback survey results showed that the second cohort, which used group assessment using virtual labs, provided favorable feedback than the first cohort.

Some cybersecurity courses use a case-study approach[3, 20]. Cai[3] shared a model of how to use a case study approach with hands-on labs using cloud-based virtual machines to teach cybersecurity courses. The end of course survey results showed that students responded positively to the case study approach compared to the traditional delivery of the courses. In addition, the case study course feedback showed that student motivation and self-efficacy were improved in the case study course.

In Vykopal[19], the authors discussed how they used two learning environments to allow students to perform the hands-on cybersecurity labs depending on the class size. While the first environment was a cloud based KYPO cyber range platform used for large or multiple classes, the second environment, used for smaller classes, utilized multiple premade images of operating systems to create VMs that were installed on lab computers or the students' own desktop or laptop.

Ngo, Cui, and Chen[17], shared how the authors used various computing infrastructures in cybersecurity courses taught. The infrastructures included on-site local network of computers, VMs installed on students' personal laptops, and Cloud-Lab, a national computing infrastructure deployed in the cloud. According to , "the former two presented various logistical, technical, and administrative challenges in ensuring a seamless and transparent hands-on learning environment for students". Student feedback was positive regarding the used cloud computing environments.

3 The Cyber Kill Chain Model

Hutchins, Clopperty, and Amin[9] proposed the structure and phases of the cyber kill chain (CKC). The CKC model breaks down an intrusion attack into consecutive stages to help cybersecurity analysts determine the patterns and behaviors of the intruders, their tactics, techniques, and procedures (TTP)[9, 2]. The CKC model allows cybersecurity professionals to understand the advanced persistent threats TTP and how to align defensive capabilities to interrupt and stop the chain. The defensive course of actions was derived from the DoD information operations (IO) 6D doctrine—detect, deny, disrupt, degrade, deceive, and destroy. “Defenders must be able to move their detection and analysis up the kill chain and more importantly to implement courses of actions across the CKC” [9].

Figure 1 shows the CKC to 6D course of action mapping from[9].

Phase	Detect	Deny	Disrupt	Degrade	Deceive	Destroy
Reconnaissance	Web analytics	Firewall ACL				
Weaponization	NIDS	NIPS				
Delivery	Vigilant user	Proxy filter	In-line AV	Queuing		
Exploitation	HIDS	Patch	DEP			
Installation	HIDS	“chroot” jail	AV			
C2	NIDS	Firewall ACL	NIPS	Tarpit	DNS redirect	
Actions on Objectives	Audit log			Quality of Service	Honeypot	

Figure 1: Cyber Kill Chain to 6D course of Action Matrix from [9]

To illustrate the benefits of the CKC model and the 6D course of action matrix techniques, [9] shared a case study of three adversary intrusion attempts that were observed by the Lockheed Martin Computer Incident Response Team (LM-CIRT) in March 2009. Using the CKC to analyze the intrusions, which were leveraging a “zero-day” vulnerability, the LM-CIRT network defenders successfully detected and mitigated the intrusions using the 6D course of actions and techniques.

4 Defensive Cybersecurity Course Structure

At Utah Valley University, the IT6740 Advanced Network Defense and Countermeasures course, which follows the CKC model and the 6D course of action techniques, has been taught face-to-face in a 16-week semester format. The course allows students to explore advanced cybersecurity topics in network defense, server hardening, vulnerability assessment, and mitigation scanning.

Students learn advanced network scanning, asset identification, Linux and Windows server hardening, firewall tools, intrusion and host detection concepts and tasks through an applied viewpoint using a hands-on application of 6D doctrine techniques and the use of current tools. The graduate students attending the course were IT, IS, or cybersecurity professionals working in local organizations. The course activities included hands-on assignments, readings, and multiple reports.

4.1 Hands-on Assignments

The hands-on assignments are constructed to allow the student to use current defensive cybersecurity practices and tools. The student uses the CKC model knowledge to apply the detect, deny, and deceive actions of the 6D doctrine within different contexts. Table 1 shows the structure and software/tools used in the course.

Table 1: Course Hands-on Assignments

	Assignment	**Software/Tools used**
1	Setting the Virtual environment	Kali, VMware/Oracle VirtualBox
2	Network Port Scanning using Nmap	Nmap
3	Vulnerability Assessment using Nessus	Nessus, Kali, Windows XP Windows Server 2008, Linux VMs
4	Server Hardening	Kali, Linux CentOS Windows Server 2012r2 VMs
5	Host-Based Firewalls	Kali, Linux CentOS Windows Server 2012r2 VMs
6	Snort NIDS/NIPS Assignment 1	Kali, Security Onion VMs
7	Snort NIDS/NIPS Assignment 2	Kali, Security Onion VMs
8	Snort NIDS/NIPS Assignment 3	Kali, Security Onion VMs
9	Honeypot	Kali and T-Pot Honeypot VMs
10	HIDS Endpoint Security	Kali, Linux (Ubuntu/Debian) Windows Server 2012r2, Security Onion, and Wazuh(OVA)

The course assignments introduce current defensive cybersecurity practices. The second and third hands-on assignment allow the student to identify the different open or closed ports as well as the existing vulnerabilities on the different host VMs. In assignment four the student applies different fixes to harden the different host VMs. In the host-based firewalls assignment five the student reviews the default firewall settings on the Linux and Windows server VMs and used iptables/ufw and the Windows firewall to create a set of more robust rules specifying the use case and justification considering the flow of traffic between clients and services. Assignments six, seven and eight allow the student to install a network intrusion detection system (NIDS), Security Onion (SO), and write rules to detect ICMP, TCP and different file types flowing between the hosts on the sub-net. In assignment nine the student installs and attacks a honeypot, T-PotCE, and uses the honeypot dashboard to view the different attacks. In the last assignment, host intrusion detection system (HIDS), the student installs Wazuh agents on the Linux and windows server hosts to collect Sysmon events and check changes to the file sizes, permissions, owner changes, last modification date, inode and all the hash sums (MD5, SHA1 and SHA256) of the system files and directories ensuring that the agents are reporting to the centralized SO VM manager or Wazuh OVA VM dashboard. Each student completes the assignments individually.

Each assignment, in the course shell assignment module/page, provided detailed descriptions and instructions to enable the student to perform the different tasks. Each assignment module/page included four areas, the purpose and goals of the assignment, the needed software tools and virtual machines, the tasks that should be performed, and the expected deliverables. The student submits a written document or technical report showing screenshots of accomplished tasks, all used commands, and a reflection on the lessons learned and issues encountered while performing the assignment.

4.2 Written Assignments/Reports

To enable the students to demonstrate the acquirement of the advanced defensive cybersecurity CKC skills, each student submits five detailed reports. The written reports allow students to challenge their defensive cybersecurity knowledge and hands-on skills. It enables them to research and investigate resources that can be part of their professional work. Table 2 shows the structure of the written reports used in the course.

Table 2: Course Written Assignments and Reports

	Written Assignment/Report
1	Current Verizon data breach report analysis
2	Recent breach CVEs (Solarwinds, Colonial pipeline, Log4j, etc.)
3	Cyber kill chain analysis
4	Network defense compliance (PCI-DSS, HIPAA, NIST, etc.)
5	Cybersecurity defense strategy and planning

4.3 Course Delivery

Each course shell in the learning management system included the class video recordings, presentations, demonstrations, hands-on learning and report assignments, the course information, course syllabus, grade book, calendar, and the course materials/modules. The recorded class video lectures showing why and how to use the different tools to conduct the defensive cybersecurity tasks were uploaded in the course shell media folder as well as the course Microsoft Teams channel. To keep students on track, the course calendar was populated with all the assignments and their due dates.

Students were encouraged to use the course Microsoft Teams channel and the weekly discussion Q&A forum to answer each other's questions and provide help. To allow the students to acquire the needed competencies and achieve the course outcomes, the faculty provided detailed feedback on each graded assignment. The feedback explained what the student did well, what did the student missed, how the student used the tools to meet the assignment/lab requirements, and any additional resources or tools that should have been used.

4.4 The Virtual environment

Students had to install either VMware (spring 2021 and spring 2022) Workstation Pro, offered free through VMware Academic Software Licensing Program with the university, or VirtualBox (spring 2023) on their laptops or PCs to create their own virtual environment to complete the required hands-on assignments/labs. To conduct the hands-on assignments, each student was provided a Metasploitable 2 Linux by Rapid7, a customized Windows XP and Windows server 2008 VMs, and a SO 16.04.7.3 ISO. In addition, students had to install an Offensive Security Kali, Linux CentOS, Windows Server 2012r2, Linux (Ubuntu/Debian) host, Telekom-security T-PotCE, and Wazuh OVA VMs.

5 Student Course Evaluation and Feedback

At the end of each course, students were provided an online course evaluation form. The overall course evaluation area used a five-point Likert scale to answer the following questions:

Q1: I learned more about the subject as a result of taking this class.
Q2: I learned how this subject can be used outside of the classroom.
Q3: This class challenged me to think in new ways.
Q4: I developed one or more essential skills as a result of this class.

Table 3 shows the results for the IT6740 Spring 2021, Spring 2022, and Spring 2023 overall course evaluation.

Table 3: Overall Course Evaluation Results

	SA(%)	A(%)	N(%)	D(%)	SD(%)	Avg	StdDev	95% CI
Spring 2021 (N=11, n=5)								
Q1	40	20	20	-	20	3.8	1.17	(2.78, 4.82)
Q2	40	40	-	-	20	4.0	1.10	(3.04, 4.96)
Q3	40	20	20	-	20	4.0	0.89	(3.22, 4.78)
Q4	40	20	20	-	20	3.8	1.17	(2.78, 4.82)
Spring 2022 (N=11, n=3)								
Q1	67	33	-	-	-	4.67	0.47	(4.13, 5.20)
Q2	67	33	-	-	-	4.67	0.47	(4.13, 5.20)
Q3	33	67	-	-	-	4.33	0.47	(3.80, 4.87)
Q4	67	33	-	-	-	4.67	0.47	(4.13, 5.20)
Spring 2023 (N=13, n=4)								
Q1	100	-	-	-	-	5.0	0.0	(5.0, 5.0)
Q2	100	-	-	-	-	5.0	0.0	(5.0, 5.0)
Q3	100	-	-	-	-	5.0	0.0	(5.0, 5.0)
Q4	100	-	-	-	-	5.0	0.0	(5.0, 5.0)

Students were offered to answer the following open-ended question "what helped you learn the most". The following were the written responses provided by the students who opted to answer the open-ended question. Each bullet represents the whole received response from each student.

Spring 2021

- "Difficult semester but I got through it."
- "How knowledgeable the professor is on the subject."
- "Google."

Spring 2022

- "The lectures showing how the security software is used and then doing hands-on assignments of those tools helped me understand how they work."
- "What helped me learn the most in this class was reinforcing what we learned in class with the assignments."
- "Working the assignments."

Spring 2023

- "Professor extremely knowledgeable on the subject matter and was really good at explaining everything."
- "I loved the hands on labs. Those were the most interesting and helpful assignments. The papers were interesting, but I felt like I wasn't getting as much hands on experience as I would have liked."
- "The labs were really cool and insightful. There was a broad range of tools and activities so we got a nice taste of many different solutions. It also took away

some of the stress of keeping VMs working, because we only used them for a few weeks at most. Great for a student environment because we're not afraid to try things and make a mess at times."
- "Hands on labs and exercises."

The spring 2021, spring 2022, and the spring 2023 course sections used the same course content, course structure, hands-on assignments, same virtual machines, and same paper/project requirements and rubrics. While in the spring 2021 and 2022 course sections students used VMware to host the virtual machines, in the Spring 2023 course section students used Oracle VM VirtualBox to host the virtual machines. The same faculty taught and managed the three course sections.

6 Discussion

To prepare the future IT and cybersecurity graduates to current and future practices of defensive cybersecurity, the courses need to introduce students to current tools and practices used by cybersecurity professionals to stop and prevent APT. The CKC model coupled with the DoD IO 6D doctrine course of actions enabled the IT/cybersecurity students to be job ready. The submitted graded student work showed that the students applied the CKC model learned knowledge and gained the needed skills to perform current advanced defensive cybersecurity tasks. Also, student graded work showed that the students understood how each tool was used and why the tool usage fit the defensive task within the provided scenario/context, thus allowing them to acquire the needed IT and cybersecurity skills.

Student response to the course evaluation questions showed that the students developed one or more essential defensive cybersecurity skills. Also, the students learned more about advanced defensive cybersecurity because of taking the course. Students learned how the CKC model and the 6D doctrine course of actions can be used to address outside the class cybersecurity issues. Lastly, the students agreed that the course challenged them to think in new ways.

All students managed to build their virtual environment on their laptops and PCs and complete all the assignments/labs successfully. Student written comments, in section 5 above answering the open-ended question, showed that the hands-on experience in their own virtual environment enabled them to learn the different advanced defensive cybersecurity skills. A very small number of students corrupted their VMs or had difficulties creating the VMs but they managed to overcome the faced issues through peer and faculty help. Some students commented in their reflections that the weekly provided walk-through videos helped them to learn the most. Most students highlighted the value of the hands-on assignments on their learning. In addition, some students found the practicality of the course content was another element that allowed students to learn the most. Also, many students mentioned, in many classroom discussions, that they will adopt and implement the different defensive tools used in the course in their current work environment.

7 Conclusions

Advanced defensive cybersecurity courses need to be an integral component in current graduate IT/cybersecurity education. The courses should provide advanced defensive concepts through an applied viewpoint using hands-on application of real-world cybersecurity practices and techniques using current tools in virtual environments that mimic real organizational infrastructures. Allowing students to practice the CKC model and apply the DoD IO 6D course of actions in different scenarios and contexts enables them to develop IT/cybersecurity job readiness. Analysis of student overall course evaluation and qualitative responses showed that having a personal virtual environment to gain hands-on experience with current cybersecurity tools enabled the students to acquire the different advanced cybersecurity defensive skills.

References

[1] Arwa Khalid AlSalamah, José María Sierra Cámara, and Stephen Kelly. Applying virtualization and containerization techniques in cybersecurity education. In *Proceedings of the 34th Information Systems Education Conference, ISECON*, pages 1–14, 2018.

[2] Pooneh Nikkhah Bahrami, Ali Dehghantanha, Tooska Dargahi, Reza M Parizi, Kim-Kwang Raymond Choo, and Hamid HS Javadi. Cyber kill chain-based taxonomy of advanced persistent threat actors: Analogy of tactics, techniques, and procedures. *Journal of information processing systems*, 15(4):865–889, 2019.

[3] Yu Cai. Using case studies to teach cybersecurity courses. *Journal of Cybersecurity Education, Research and Practice*, 2018(2):3, 2018.

[4] Cyberseek. Cybersecurity supply/demand heat map, 2023. Visited June 27, 2023. URL: `https://www.cyberseek.org/heatmap.html`.

[5] eSENTIRE. 2023 official cybersecurity jobs report, 2023. Visited June 27, 2023. URL: `https://www.esentire.com/resources/library/2023-official-cybersecurity-jobs-report`.

[6] Center for Strategic & International Studies. Significant cyber incidents, 2023. Visited June 8, 2023. URL: `https://www.csis.org/programs/strategic-technologies-program/significant-cyber-incidents`.

[7] CC2020 Task Force. *Computing Curricula 2020: Paradigms for Global Computing Education.* Association for Computing Machinery, New York, NY, USA, 2020. https://doi.org/10.1145/34567967 `doi:10.1145/34567967`.

[8] Regina Hartley, Dawn Medlin, and Zach Houlik. Ethical hacking: Educating future cybersecurity professionals. In *Proceedings of the EDSIG Conference ISSN*, volume 2473, page 3857, 2017.

[9] Eric M Hutchins, Michael J Cloppert, and Rohan M Amin. Intelligence-driven computer network defense informed by analysis of adversary campaigns and intrusion kill chains. *Leading Issues in Information Warfare & Security Research*, 1(1):80, 2011.

[10] Sean Michael Kerner. Colonial pipeline hack explained: Everything you need to know, 2022. Visited June 20, 2022. URL: `https://www.techtarget.com/whatis/feature/Colonial-Pipeline-hack-explained-Everything-you-need-to-know?Offer=abVidRegWall\_gate`.

[11] Cheng Chung Kuo, Kai Chain, and Chu Sing Yang. Cyber attack and defense training: Using emulab as a platform. *Int. J. Innov. Comput. Inf. Control*, 14(6):2245–2258, 2018.

[12] Phil Legg, Alan Mills, and Ian Johnson. Teaching offensive and defensive cyber security in schools using a raspberry pi cyber range. In *Journal of The Colloquium for Information Systems Security Education*, volume 10, pages 9–9, 2023.

[13] Mohamed Lotfy. Teaching a penetration testing course during covid-19 - lessons learned. *Journal of the Consortium for Computing Sciences in Colleges*, 37(2):85–99, 2021.

[14] Arghir-Nicolae Moldovan and Ioana Ghergulescu. Leveraging virtual labs for personalised group-based assessment in a postgraduate network security and penetration testing module. In *2020 15th International Workshop on Semantic and Social Media Adaptation and Personalization (SMA*, pages 1–6. IEEE, 2020.

[15] Steve Morgan. Cybersecurity jobs report: 3.5 million unfilled positions in 2025, 2023. Visited June 27, 2023. URL: `https://cybersecurityventures.com/jobs/`.

[16] Djedjiga Mouheb, Sohail Abbas, and Madjid Merabti. Cybersecurity curriculum design: A survey. In *Transactions on Edutainment XV*, pages 93–107. Springer, 2019.

[17] Linh B Ngo, Liu Cui, and Si Chen. Computing infrastructures to support cybersecurity education. In *34th Annual Conference of The Pennsylvania Association of Computer and Information Science Educators*, page 30, 2019.

[18] Saheed Oladimeji and Sean Michael Kerner. Solarwinds hack explained: Everything you need to know, 2023. Visited June 27, 2023. URL: `https://www.techtarget.com/whatis/feature/SolarWinds-hack-explained-Everything-you-need-to-know`.

[19] Jan Vykopal, Pavel Čeleda, Pavel Seda, Valdemar Švábenskỳ, and Daniel Tovarňák. Scalable learning environments for teaching cybersecurity hands-on. In *2021 IEEE Frontiers in Education Conference (FIE)*, pages 1–9. IEEE, 2021.

[20] Xinli Wang and Yan Bai. Introducing penetration test with case study and course project in cybersecurity education. In *Journal of The Colloquium for Information Systems Security Education*, volume 9, pages 6–6, 2022.

Experiences Introducing the POGIL Methodology for Teaching Computer Organization & Architecture*

Pamela M. Smallwood
Department of Computer and Cyber Sciences
Regis University
Denver, CO 80221
psmallwo@regis.edu

Abstract

This paper describes one CS department's experiences with introducing Process Oriented Guided Inquiry Learning (POGIL) activities in CS classrooms. POGIL is an active and collaborative learning methodology, in which students work together in small groups to complete guided activities that help them construct understanding and develop process skills. This paper will discuss what POGIL is, the benefits of POGIL, how POGIL activities were first introduced in CS1, and how POGIL activities were then developed and introduced within a Computer Organization and Architecture course. Informal results and future work will also be discussed.

1 Background

Ideally learning environments (such as college classrooms) would implement methodologies that support the research about how students learn. Research shows people learn better and remember more when they construct their own understanding following an explore-invent-apply learning cycle, when they interact with others, and when they reflect on their performance[12]. Learning outcomes increase as learning environments progress from passive, to active, to constructive, to interactive, so students are interacting with each other to construct understanding of a topic[1]. McConnell[10] discussed the benefits of active and collaborative learning in Computer Science (CS). Process Oriented Guided Inquiry Learning (POGIL) supports all of this research.

1.1 What is POGIL?

In POGIL[17] classrooms, students work together in self-managed 3-4 person teams, completing carefully designed guided-inquiry activities. Each POGIL activity contains several models (e.g., figures, tables, sample code). The models are followed by a series of questions that guide teams through one or more of the following learning cycles: explore the model, discover/invent key concepts, and apply new understanding. Each team member has a rotating role (e.g., manager, spokesperson, recorder, and analyst). The roles help students develop process skills (e.g., communication, teamwork, problem solving, and critical thinking), along with learning content material. Instead of lecturing, the instructor facilitates learning by assisting teams, as needed, while they complete the activities.

1.2 Use and Effectiveness of POGIL

While POGIL has been used in other STEM disciplines for over 20 years[11], use of POGIL within computing disciplines is newer. The 2011-2016 CS-POGIL project[2] began developing POGIL activities for CS. The 2016-2020 IntroCS POGIL project[8] created many introductory Java and Python activities and provided training and mentors for faculty starting to use POGIL in CS1. The CS department at Regis University was able to introduce POGIL within its CS1 courses starting in fall 2018, using previously developed activities and guidance from the IntroCS POGIL project.

Studies on the use of POGIL compared to over traditional lectures [3, 15, 4] have found that students using POGIL had more positive attitudes about the course and instructor while demonstrating higher content mastery, and courses using POGIL had lower attrition rates. Walker and Warfa's meta-analysis of 21 studies comparing POGIL courses to standard lecture courses[16] also found that the odds of passing a course (grade of C or better) were roughly 2 times higher in a POGIL classroom.

Research with CS1 faculty[5] indicated that using POGIL in CS1 courses led to more active learning and engagement in class, a deeper understanding of concepts, and encouraged positive peer-to-peer relationships (which helps develop a sense of community). And more recently, students using POGIL in

CS1 were found to perform better on post-tests at the end of the course, as well as on retention tests given in a follow-on course[9].

These findings tracked with the Regis CS department's experiences when first implementing POGIL in CS1. Compared to a baseline standard Java assessment given in the term before introducing POGIL, grades on the assessment improved between 12% and 19% during the first three terms of POGIL usage (with no other changes to the course and using the same instructor). Students seemed to form stronger bonds with others in the class, and attendance improved dramatically – students still rarely miss classes with POGIL activities. Therefore, POGIL continues to be the primary methodology used in the department's CS1 classrooms. There are 3 classroom sections of CS1 per year, with about 18 students per section. They complete 20 Java POGIL activities (developed by the IntroCS POGIL project[8]) in teams of 3 (so they easily can share a computer screen) over the course of one semester.

2 POGIL in Computer Organization and Architecture

Following the success of POGIL in CS1, the CS department became interested in expanding its usage to other CS courses, especially Computer Organization & Architecture (CO&A). However, unlike CS1, there were fewer POGIL activities available for CO&A topics, and none for the MIPS language/datapath taught at Regis.

2.1 Initial POGIL use in CO&A

The CS department first tried using two existing POGIL activities (on sequential logic and register files)[2] and two activity drafts in spring 2021, to see how the students responded. The nine students worked in three teams of three, and were enthusiastic about the activities. And four of the students specifically mentioned that more POGIL group activities would be helpful, in response to the open-ended course feedback survey question, *"What else could have been included in the course that would have helped you more with your learning?"*

Given the students' positive response, the CS department wanted to add more POGIL activities to CO&A. But creating new POGIL activities is very labor-intensive and time-consuming. In fact, one of the greatest barriers cited by CS instructors for not adopting POGIL is the lack of vetted POGIL activities for the classes they teach[7].

2.2 The CS POGIL Activity Writing Program

In summer 2022, the NSF funded a CS POGIL activity writing program [6]. Computer Science faculty across the country developed 58 new, high-quality

CS POGIL activities (see Resources section for more information on the activities developed). Authors were given training on POGIL activity writing criteria, which included writing clear learning objectives, designing good models, incorporating explore-discover-apply learning cycles, and including process skills and self-assessment in activities. Each new activity was reviewed and revised as many times as necessary to insure it met all of the above criteria. After a final content review by an expert in the field, the new activity was accepted for publication by the POGIL Activity Clearinghouse (PAC)[14] and for classroom testing.

During the writing program, this author developed nine new activities for Computer Organization &Architecture courses that use the popular Patterson and Hennessy MIPS textbook[13]. Space limitations of this paper prevent complete coverage, but a sampling of some of the CO&A activities will be covered to give the reader an idea of how POGIL activities were incorporated.

3 Activity Example 1: IEEE Format for Binary Floating-Point Numbers

This activity covers data representation of floating-point numbers. It was chosen as an example for this paper, because it has a broader application than just CO&A and can be used in many courses. Several of the models will be shown, along with sample questions.

Model 1 introduces terminology (e.g., normalized form, significand) using decimal floating-point numbers. Then Model 2 (shown in Figure 1) gives an example of converting a decimal floating-point value to a binary floating point value, and shows normalizing binary values and breaking them into parts. Student questions following Model 2 lead to converting decimal floating point values to binary normalized form. For example, some explore and discover questions for the conversion example in Model 2 are:

- *How is the value for 2^{-3} mathematically related to the value for 2^{-2} ?*
- *The table only gives decimal values out to 2^{-4}. As a team, determine the decimal values of 2^{-5}.*

And some explore and discover questions for the normalization examples in Model 2 are:

- *For the values 11001.11 and -0.001101, how many **places** and what **direction** did the radix point move to get from the original binary floating point number to the significand in the normalized form?*
- *How does the **exponent**'s sign and value relate to the movement of the radix point?*

The fractional part of binary **floating point** numbers can be represented using ***negative*** powers of 2 to represent the values to the ***right*** of the radix point (where a **radix** point in binary is similar to a **decimal** point in decimal). In the table below, $\mathbf{2^{-1}}$ is equivalent to $\mathbf{1/2^1}$ and $\mathbf{2^{-2}}$ is equivalent to $\mathbf{1/2^2}$, etc.

2^3	2^2	2^1	2^0	.	2^{-1}	2^{-2}	2^{-3}	2^{-4}
8	4	2	1	.	.5	.25	.125	.0625

Example:
Decimal number `2.625` would be represented in binary as `10.101`
The ones in the binary representation would line up under each column in the table as follows:

		1	0	.	1	0	1	

Convert the binary value back to decimal, by adding the decimal values for each 1 bit:

`2 + 0.5 + 0.125 = 2.625`

Similar to decimal numbers, binary numbers can also be written in **binary normalized form**, using powers of 2 for the exponents (instead of powers of 10).

Decimal Floating Point Number	Equivalent Binary Floating Point Number	Binary Normalized Form	Parts		
			Sign	Significand	Exponent
25.75	11001.11	1.100111×2^4	+	1.100111	4
1.5	1.1	1.1×2^0	+	1.1	0
-0.203125	-0.001101	-1.101×2^{-3}	–	1.101	-3

Figure 1: Example 1 Model 2: Binary Floating Point

Finally, some example application questions for Model 2 are:

- *Given decimal value -0.125, what would its binary floating point equivalent be?*
- *Give the normalized version of your previous answer.*

Model 3 (shown in Figure 2) covers the IEEE format and storage of exponents as biased binary values. Student questions following Model 3 lead students to the discovery the decimal value of the bias used and determining the stored binary exponent. Some example explore and discover questions for Model 3 are:

- *Recalling ranges for binary values, what's the largest positive signed binary value that can be stored using 8-bit two's complement?*
- *What is decimal value of the binary value above?*
- *The value you found in the previous questions (binary and decimal versions) is known as the **bias**.*
  - *What is the difference between 132 and the bias?*
  - *What is the difference between 123 and the bias?*
  - *Use the table to determine what value these differences represent.*

The IEEE single precision standard uses 32 bits to represent a binary floating point value.
The format divides a floating point number into the three parts we have examined:
the **sign**, the **exponent**, and the **mantissa** (aka, the **significand**).

Sign	Exponent	Mantissa/Significand
0	00000000	00000000000000000000000
1 bit	8 bits	23 bits

Storing Biased Exponents

The **exponent** bits represent the signed power of 2 from the **normalized binary form**,
but they not stored in two's complement representation.
Instead they are stored as an **unsigned binary** value *relative* to a **bias**.

Examples:

Binary Normalized Form	Decimal Exponent of 2	Binary Biased Exponent (Stored Exponent Bits)	Decimal Equivalent of Binary Biased Exponent
-1.101 x 2^{-4}	-4	01111011	123
1.11 x 2^{-3}	-3	01111100	124
1.0 x 2^{0}	0	01111111	127
-1.1011 x 2^{2}	2	10000001	129
1.100111 x 2^{5}	5	10000100	132

Figure 2: Example 1 Model 3 IEEE Format & Biased Exponent

- *Have your team come up with a formula that could be used to determine the value of the biased exponent (decimal version).*

An example application question for Model 3 requires the team to show how the exponent for 1.00012 x 2^7 (in binary normalized form) would be stored in the 8-bit exponent field of the IEEE format.

Model 4 (shown in Figure 3) covers storage of the sign and significand with the exponent. Student questions following Model 4 lead students to understand what bits are stored (and not stored) and to perform an entire decimal to IEEE floating point conversion. Some example explore and discover questions for Model 4 are:

- *What bits from the significand in the binary normalized form are stored in the significand field?*
- *What is missing from the significand field that was part of the binary normalized form?*
- *As a team, come up with a possible explanation about why these items do not need to be stored.*

An example application question for Model 4 requires the team to show how decimal 15.25 would be stored in IEEE single precision binary format.

Storing the Mantissa/Significand
As seen above, the significand value is stored in the last 23 bits of the IEEE representation.
However, in order to save bits, only ***part*** of the **normalized significand** is stored.

Examples (*spaces added to significand field for readability*):

Binary Normalized Form	Sign bit field	Exponent field (8 bits of biased exponent)	Significand field (23 bits stored in IEEE format)
-1.101×2^{-4}	1	01111011	10100000 00000000 0000000
1.11×2^{-3}	0	01111100	11000000 00000000 0000000
1.0×2^{0}	0	01111111	00000000 00000000 0000000
-1.1011×2^{2}	1	10000001	10110000 00000000 0000000
1.100111×2^{5}	0	10000100	10011100 00000000 0000000

Figure 3: Example 1 Model 4 Storing the significand

This activity ends with a homework exercise that creates an algorithm for converting from decimal floating-point numbers to their IEEE representations, using what was learned in the activity.

4 Activity Example 2: MIPS Machine Language

Translation of MIPS assembly to MIPS machine language was split into two activities, so that each could be finished within one class period. The first activity introduced the three MIPS machine language formats and translating R-type and I-type immediate math instructions. In the second activity, Model 1 covers translating data transfer instructions (LW/SW), Model 2 covers translating branch instructions (BEQ/BNE), and Model 3 covers translating the jump instruction (J).

Model 2 (shown in Figure 4) will be discussed as an example. Some initial explore/discover questions for Model 2 are:

- *Examine the BEQ instruction code segment in the table in Model 2.*
  - *What is the byte address of the BEQ instruction?*
  - *Recall that a new Program Counter (PC) value (byte address of the next sequential instruction) is calculated during fetch. After the fetch for this BEQ instruction, what will the new PC value be?*
  - *What is the byte difference when the new PC value is subtracted from the address of the NEXT label?*
  - *What would this difference be in words (instead of bytes)?*
  - *What is the decimal value of the bits that store the offset value?*
- *An offset must be relative to something that provides a starting point. What is the offset for branch instructions relative to?*

Model 2: I-type instructions (branch)

Recall that MIPS instructions are all 32 bits in length, so instruction byte addresses increase by 4 bytes per instruction. And I-type instructions all have unique opcodes. The opcode for BEQ is 4, and the opcode for BNE is 5. Here are two assembly code segments, located at the given byte addresses on the left:

```
620         BEQ $8, $9, NEXT
624         next-sequential-instruction
628         another-instruction
632  NEXT:  label-instruction
```

```
880  LOOP:  label-instruction
884         another-instruction
888         another-instruction
892         another-instruction
896         BNE $10, $20, LOOP
900         next-sequential-instruction
```

Here are the machine code translations for the branch instructions from the above assembly code segments.

MIPS Assembly Code				*MIPS I-type Machine Code*			
instruc	RS,	RT,	LABEL	Opcode	RS	RT	offset (2's complement) in words
BEQ	$8,	$9,	NEXT	000100	01000	01001	00000000 00000010
BNE	$10,	$20,	LOOP	000101	01010	10100	11111111 11111011

Figure 4: Example 2 Model 2 MIPS branch instructions

- *As a team, come up with a description of the bit value stored in the offset field of the machine code, in relation to the assembly language code.*

Similar questions are asked about the BNE instruction in the model. Then the final application questions for Model 2 require the student to translate several branch instructions from assembly language to machine language.

5 Activity Example 3: Data Hazards and Forwarding

This activity introduces data hazards and leads the students to understand how forwarding could solve them. Model 2 is shown in Figure 5.

Model 2: Pipeline Diagram with intermediate Pipeline Registers

We will now examine how the data hazards might be handled. Model 2 shows the same instructions in the pipeline as Model 1. But Model 2 also contains the **pipeline registers** that store intermediate values between stages, shown between the stages that they connect.
Stages being performed during each clock cycle are **highlighted** in grey.

Instructions	1		2		3		4		5		6		7
SUB $8, $16, $10	IF	IF/ID	ID	ID/EX	EX	EX/MEM	MEM		WB				
AND $12, $8, $9			IF	IF/ID	ID	ID/EX	EX		MEM	MEM/WB	WB		
OR $13, $10, $8					IF	IF/ID	ID		EX	EX/MEM	MEM	MEM/WB	WB

Figure 5: Example 3 Model 2

Some initial explore/discover questions for Model 2 are:

- *In the Model, fill in the names of the missing pipeline registers used to store the values generated during clock cycle 4.*
- *Examine the execution of the SUB instruction in the model.*
  - *In which stage does the SUB instruction use the ALU to compute the new value for register $8?*
  - *During which cycle does the computation happen?*
  - *At the end of this cycle, where does the SUB instruction store the computed ALU result (for use in later cycles)?*
- *Examine the execution of the AND instruction in the model.*
  - *In which stage does the AND instruction use register $8's value to compute a new value for register $12?*
  - *During which cycle does the computation happen?*
  - *During this cycle, which stage is the SUB instruction in?*
  - *During this cycle, from which pipeline register does the SUB instruction retrieve its ALU result (calculated in SUB's EX stage)?*
  - *Draw an arrow on Model 2, from the pipeline register containing the SUB instruction ALU result for this cycle, to the AND instruction stage that uses register $8's value to compute something.*

Some application questions for Model 2 are:

- *The arrows your team drew demonstrate the concept of **forwarding**. As a team, come up with a definition of **forwarding**.*
- *Assuming a MIPS pipelined CPU implemented **forwarding** of register values as shown, would any of the data hazards in this code still be a problem? Why or why not?*

6 Classroom Testing of the New CO&A POGIL Activities

All nine of the new CO&A POGIL activities developed by the author during the CS POGIL Activity Writing Program were classroom tested for the first time during the spring 2023 semester, along with the two used previously that were developed by other authors, and three draft activities by this author, bringing the total number of POGIL activities used in the course to 14. The class met twice a week for 15 weeks, and each class session was 75 minutes long. There were eight junior- and senior-level CS students enrolled (seven male, one female) in the course, so students were divided into two teams of four.

Before each POGIL activity, the instructor gave a short (under 5 minutes) introduction to the topic. Students were given their own copies of the activity, but only the recorder's copy with the team's consensus answers was collected by the instructor (for grading, students were simply given participation points when they completed an activity cooperatively on a team). Each team gathered around a table, and engaged in discussions to solve each question on the activity. The instructor circulated around the room, answered questions and monitored/redirected the teams, as needed. When teams finished the questions for each model, the instructor led a read out of answers to key questions, with the teams providing the answers. Any differences between the teams' answers were discussed, to insure the learning objectives were acheived. After completing the entire activity, teams filled out a team activity report, reflecting on their learning and use of process skills. Following the activities, students individually completed homework assignments that required application of the concepts learned. Some activities required the entire class period, while others only required part of the session.

The students were much more engaged in the material, as demonstrated by the active discussion and interactions within the teams. And after the first few POGIL activities, whenever a class session did not include a POGIL activity, students voiced disappointment. No specific research was done during this first usage of these new activities. But within the end-of-course evaluations, students were asked to rate the effectiveness of all the POGIL activities, using the question:

- *How important were the POGIL activities in helping you successfully learn the material?*

Answers given for this question are shown in Table 1.

Table 1: Student evaluations - Importance of POGIL activities

very unimportant	unimportant	neutral	important	very important
0	0	2	2	4

Additionally, in the general comments section, several students specifically mentioned the POGIL activities:

- "The POGIL activities provided in class made the biggest difference in understanding the material."
- "Towards the end of the semester, some more POGIL activities relating to caches and I/O would have been beneficial, since the activities really helped my understanding of concepts at the beginning of the semester."

7 Discussion, Conclusions, and Future Work

In the spring 2023 CO&A course, most of the students considered the activities to be important or very important contributions in helping them learn the material. However, since the POGIL activities were not strictly evaluated by any official research questions, the findings are very informal for this pass.

An additional benefit of using POGIL (beyond previous points) became clear during classroom testing. Monitoring the team activities made it very obvious to the faculty which parts of the topics were understood completely and which parts needed more coverage. This is often very hard to discern during a lecture.

During the classroom testing of the nine new activities, several of the activities took longer than the author estimated and had to be continued in the next class period. These activities will be revised, either by shortening them or splitting them into smaller activities, to better fit with classroom time constraints.

It is the author's hope that this preview of some CO&A POGIL activities within this paper piqued the interest of other CO&A faculty to try POGIL. The author will continue to develop more POGIL activities for CO&A. Future work could examine POGIL's effect on student mastery of CO&A material.

8 POGIL Resources

8.1 The POGIL Project (`https://pogil.org`)

The POGIL project website provides more information on POGIL and its usage and effectiveness. The POGIL project also presents workshops (`https://pogil.org/events`) to help faculty learn about POGIL theory and practice, including how to facilitate, evaluate, and develop POGIL activities.

8.2 CS Activities from the CS POGIL Activity Writing Program (`https://bit.ly/2022cspogil`)

This webpage lists all 58 of the new Computer Science POGIL activities developed within the 2022 CS POGIL Activity Writing Program (described in this paper) for a large variety of CS topics (beyond CO&A). Access to the full activities, along with answer keys, can be requested at the site.

CO&A activities developed by the author that are included in the list are:

- IEEE Single Precision Format for Binary Floating Point Numbers
- Booth's Algorithm

- MIPS machine language – Part 1: R-type and I-type immediate
- MIPS machine language – Part 2: I-type data transfer and branch, jump
- Building a MIPS Single Cycle Datapath - Part 1: R-type Instructions
- Building a MIPS Single Cycle Datapath - Part 2: I-type Instructions
- Pipeline Diagrams and Pipeline Timing
- Data Hazards and the concept of a Forwarding solution
- Data Hazard Detection and Handling – Part 1: The Forwarding Unit

NOTE: Contact the author for access to editable versions of any these activities.

8.3 Other CS POGIL Activities (`https://cspogil.org`)

This website hosts the many activities developed by the CS-POGIL project and the IntroCS-POGIL Project.

9 Acknowledgements

Thank you to National Science Foundation grant DUE-162676 for funding the CS POGIL Writing project and to the project mentors and peer reviewers for their constructive feedback on improving the activities.

References

[1] Michelene T. H. Chi and Ruth Wylie. "The ICAP Framework: Linking Cognitive Engagement to Active Learning Outcomes". In: *Educational Psychologist* 49.4 (2014), pp. 219–243. DOI: 10.1080/00461520.2014.965823.

[2] *CS-POGIL Project*. https://cspogil.org.

[3] John J. Farrell, Richard S. Moog, and James N. Spencer. "A Guided-Inquiry General Chemistry course". In: *Journal of Chemical Education* 76.4 (1999), pp. 570–574. DOI: 10.1021/ed076p570.

[4] David M. Hanson. *Instructor's Guide to Process-Oriented Guided-Inquiry Learning*. Hampton, NH: Pacific Crest, 2006.

[5] Helen H. Hu and Tricia D. Shepherd. "Teaching CS 1 with POGIL activities and roles". In: *Proceedings of the 45th ACM Technical Symposium on Computer Science Education*. SIGCSE '14. Atlanta, GA: ACM, Mar. 2014, pp. 127–132. DOI: 10.1145/2538862.2538954.

[6] Helen H. Hu, Tricia D. Shepherd, and Clif Kussmaul. "The CS POGIL Activity Writing Program". In: *Proceedings of the 54th ACM Technical Symposium on Computer Science Education*. Vol. 2. SIGCSE 2023. Toronto, ON, Canada: ACM, Mar. 2023, p. 1408. DOI: 10.1145/3545947.3576352.

[7] Helen H. Hu et al. "Results from a Survey of Faculty Adoption of Process Oriented Guided Inquiry Learning (POGIL) in Computer Science". In: *Proceedings of the 2016 ACM Conference on Innovation and Technology in Computer Science Education*. ITiCSE '16. Arequipa, Peru: ACM, July 2016, pp. 186–191. DOI: 10.1145/2899415.2899471.

[8] *IntroCS-POGIL project*. http://introcspogil.org.

[9] Chris Mayfield et al. "POGIL in CS1: Evidence for Student Learning and Belonging". In: *Proceedings of the 53rd ACM Technical Symposium on Computer Science Education*. Vol. 1. SIGCSE 2022. Providence RI: ACM, Feb. 2022, pp. 439–445. DOI: 10.1145/3478431.3499296.

[10] Jeffrey J. McConnell. "Active and Cooperative Learning: Tips and Tricks (part I)". In: *ACM SIGCSE Bulletin* 37.2 (2005), pp. 27–30. DOI: 10.1145/1083431.1083457.

[11] Richard S. Moog, James N. Spencer, and Andrei R. Straumanis. "Process-oriented guided inquiry learning: POGIL and the POGIL Project". In: *Metropolitan Universities Journal* 17.4 (2006), pp. 41–52. URL: https://journals.iupui.edu/index.php/muj/article/view/20287.

[12] National Research Council. *How People Learn: Brain, Mind, Experience, and School, Expanded Edition*. Washington, DC: National Academy Press, 2000.

[13] David A. Patterson and John L. Hennessy. *Computer Organization and Design: The Hardware/Software Interface*. 6th ed. Waltham, MA: Elsevier, 2021.

[14] *POGIL Activity Clearinghouse*. https://pac.pogil.org.

[15] Andrei Straumanis and Emily A. Simons. "A Multi-Institutional Assessment of the Use of POGIL in Organic Chemistry". In: *Process Oriented Guided Inquiry Learning (POGIL)*. American Chemical Society, 2008. Chap. 19, pp. 226–239. DOI: 10.1021/bk-2008-0994.ch019.

[16] Lindsey Walker and Abdi-Rizak M. Warfa. "Process Oriented Guided Inquiry Learning (POGIL) Marginally Effects Student Achievement Measures butSubstantially Increases the Odds of Passing a Course". In: *PLoS ONE* 12.10 (2017), pp. 1–17. DOI: 10.1371/journal.pone.0186203.

[17] *What is POGIL?* https://pogil.org/what-is-pogil.

Is the Amount of Computer Game Play Since High School Associated With Mental Health Outcomes in Adulthood?*

Max Marc
Management Information Systems
Black Hills State University
Spearfish, SD 57783
max.marc@bhsu.edu

Abstract

For computer educators, there is increasing concern about the dark side of information technology. Useful innovations, such as social media and artificial intelligence, often have a dark side to them. Computer games also have this dual nature. Useful as pedagogical tools, and providing hours of joy to millions of people, computer games may nonetheless be associated with adverse mental health outcomes when abused. This study reports on findings from the longest running longitudinal survey related to computer game usage, covering a single cohort over a 14 year span from adolescence to adulthood. Computer game play during adolescence and early-adulthood was not found to be associated with adverse mental health outcomes in adulthood, but computer game play during adulthood (average cohort age of 30 years) was found to be significantly associated with adverse mental health outcomes.

1 Introduction

For computer educators, computer games represent a double-edge sword. One the one hand, computer games have some positive associations for pedagogy[1, 7]. They help lower entry barriers for people to learn to use computers and are useful for introducing students to concepts in computer programming and user interface design. They are found to offer cognitive benefits and are credited with attracting young people to consider a career in the computer and information systems field.

On the other hand, computer games also have some negative associations for pedagogy. They have been found to be addictive and distracting, and to have associations with adverse mental health outcomes[3]. Indeed, the concern has spawned terminology such as “gaming disorder”[11] and “pathological gaming”[12]. With the additional rising specter of pathological use of social media, there is a need for more longitudinal research studies, based on long-term high-quality data, to study the long-term mental health consequences of using double-edged technologies such as computer games.

2 Background

Extensive prior research has examined associations between computer game play and mental health outcomes, among individuals[4], examining these associations from diverse perspectives. However, the findings have been disparate, inconsistent, and inconclusive. On whether computer game play is associated with mental health outcomes, the findings range from no association, to associations with both positive and negative mental health outcomes. In some cases, computer gaming was found to lead to outcomes such as higher self-confidence, self-efficacy, connection with others, and prosocial behavior. In other cases, computer gaming is found to lead to addiction, disassociation, loss of connection with others, and antisocial behavior. The research work needed to integrate these disparate, inconsistent findings is still ongoing.

Two theoretical perspectives, namely uses-and-gratifications theory[10] and sociotechnical theory[5], can be useful in informing the complex issue of why the findings have been inconsistent and inconclusive[9]. The uses-and-gratifications perspective suggests that individuals are active seekers and users of solutions for their emotional needs (e.g. entertainment needs, relationship needs, sensation needs). Computer games are such a solution. Depending on their disparate needs, users will have disparate outcomes. The sociotechnical perspective suggests that computer games are not neutral. They embody the values and intents of those who have the power to define the terms of the design and distribution of computer games. Moreover, this power dynamic is not static, and is continually negotiated between suppliers and the consumers. Depending on who has the upper hand in this dynamic in each situation, there will be disparate outcomes from consuming computer games.

A potential shortcoming of the existing body of research is that it consists mostly of cross-sectional studies of data (i.e., where data on computer game use and mental health were collected during a single concurrent episode). While useful in many ways, cross-sectional studies can be limited in how well they can inform long-term or time-delayed associations. For a complex, potentially biologically based factor such as mental health, the passage of time, in and of itself, should be considered potentially influential. However, possibly due to the complexity and difficulty of doing long-term longitudinal studies, there are relatively few studies that have incorporated long-term, longitudinal data. Another shortcoming is that most of this research has focused on mental health outcomes among the youth; studies relating to adult mental health are few[8]. There need to be more studies that focus on adult mental health outcomes.

While there are a small number of longitudinal studies that have examined associations between computer game play and adult mental health outcomes, they have tended to focus on early-adulthood[2] and haven't covered very long lengths of time. To integrate and clarify past inconsistencies, there is a clear and present need for more longitudinal studies that focus on long-term computer game play and the association with adult mental health outcomes. The next section describes how this study uses longitudinal data collected over a very long-term (14 years), for the same cohort, from adolescence to adulthood (average adult cohort age of 30 years).

3 Research Method

3.1 Data

This study uses survey data from "The National Longitudinal Study of Adolescent to Adult Health" (abbreviated as "Add Health")[6]. The "Add Health" survey was conducted in several "Waves" and is currently still ongoing. The first Wave used a nationally representative sample of high schoolers in the USA, with an average cohort age of 16 years. Every few years, the same cohort was surveyed again, constituting a new Wave. This study uses data from the first four Waves, because they contain questions pertinent to the study. At the time of the fourth Wave, respondents were adults with an average cohort age of 30 years. The public-use dataset has an unweighted sample size of approximately 6,500 actual respondents in the first Wave, with a small amount of attrition in subsequent waves. An anonymized unique identifier is available to track each respondent across multiple Waves. For analysis purposes, this sample size will be adjusted later using a weighting variable.

3.2 Survey Questions

"Add Health" Waves one to four each contained a question pertaining to how many hours per week the respondent played computer games (questions h1da10, h2da10, h3da10, h4da10). Wave four contained ten questions pertaining to the mental health of the (then adult) respondents. The mental health questions concerned frequency of feeling "bothered by things that usually don't bother you", "depressed", "too tired", "happy", "sad", "others disliked you", "you were just as good as other people". Additional questions referenced trouble staying focused on what one was trying to do, and one's enjoyment of life (questions h4mh18 to h4mh27). The mental health questions were coded on the following scale: 0 = "never or rarely", 1 = "sometimes", 2 = "a lot of the time", 3 = "most of the time or all of the time". For all questions, additional possible responses were "Refused to answer" and "Don't know".

3.3 Quantitative Model

Panel data was constructed using survey respondents who 1) responded ito the question pertaining to how many hours they played computer games in all four Waves, and 2) responded to all questions pertaining to mental health in Wave 4. A weighting variable, available in the dataset, was incorporated to readjust the sample size. The weighting variable reduced the weightage of respondent categories that were over-represented and increased the weightage of respondent categories that were under-represented, within the actual survey sample. The weight-adjusted sample is designed to be representative of the population of the USA. The next section contains descriptive statistics of the weight-adjusted sample.

This study proposes to test the association between hours per week of computer games played in each of four survey Waves, and mental health of respondents in the fourth Wave. A single criterion variable for mental health is calculated as the average of the responses for all the ten mental health questions (after reverse-coding two questions that had opposite-coded scales). Multiple regression was used to conduct the analysis. Multiple regression is appropriate here because all the variables are numeric and there is a single criterion variable.

In Wave 1, the respondents had an average cohort age of 16 years. In Wave 2, the same respondents had an average cohort age of 18 years. In Wave 3, the average cohort was 23 years, and in Wave 4 it was 30 years. Having recorded how much respondents played computer games in each of these Waves, it is possible to build a longitudinal model of how very long-term engagement with computer games may be associated with mental health outcomes in Wave 4, when respondents were adults with an average cohort age of 30 years. It is also possible to measure whether the association from playing computer games in certain age milestones (i.e. Waves) was more or less significant than in other age milestones. A picture thus emerges of how computer game play, over a very long term, in different age categories, may be associated with mental health outcomes in adulthood. Figure 1 offers a visual model of the association being tested.

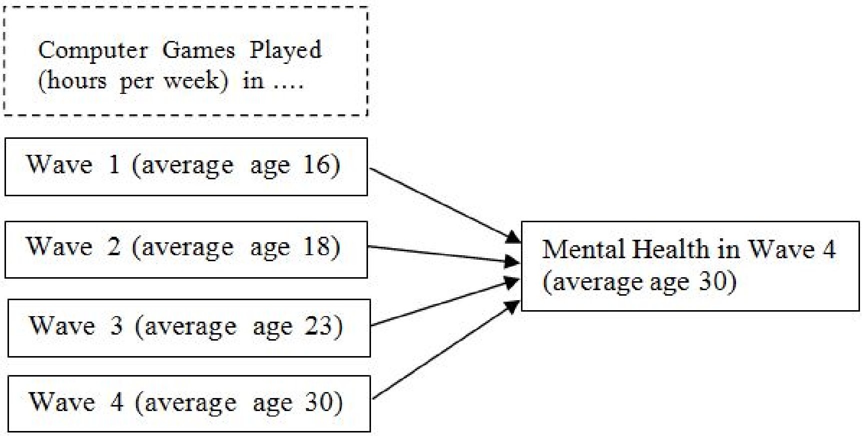


Figure 1: The Quantitative Model

The regression equation is:

MHW4 = b0 + b1CGPW1 + b2CGPW2 + b3CGPW3 + b4CGPW4

Where:

MHW4 = Mental health in Wave 4

CGPWx = Computer Game Play (hours per week) in Wave x

4 Results

4.1 Descriptive Statistics

Table 1: Distribution by Sex

Sex	Proportion
Male	46.4%
Female	53.6%

Table 2: Distribution by Race

Race/Ethnicity	**Proportion**
White (non-hispanic)	76.9%
Black	14.7%
Asian/Pacific Islander	4.1%
Native American	2.9%
Other	6.4%

Table 3: Self-Identified Hispanic Origin

Yes, of Hispanic Origin	11.2%

The weight-adjusted sample size of the panel data is 2,498. The demographic distributions of the weight-adjusted sample data (see Tables 1, 2, and 3) are similar to national distributions (except for self-identified Hispanic Origin in Table 3, which is much lower in this sample than in the general USA population, estimated at 18.7%[13]). This indicates that the panel data are broadly representative of the wider population in the USA.

Table 4: Hours Per Week Computer Games Played

	Wave 1	**Wave 2**	**Wave 3**	**Wave 4**
Percentage of respondents who played zero hours	43%	31.7%	25%	54.1%
Average hours played per week by those who played	5.21	4.42	7.39	7.51

Table 5: Mental Health Distribution in Wave 4

Mental health score range	<= 0.3	0.31 to 0.5	0.51 to 0.8	>=0.81
Proportion of respondents in that range	25%	25%	25%	25%

Table 4 indicates hours per week playing computer games. And Table 5 indicates that 25% of respondents in Wave 4 had a mental health score of less than 0.3, 25% had a mental health score of between 0.31 to 0.50, and so on.

As explained in Section 3.3, the mental health score is calculated as the average of the scores in the ten questions in Wave 4 that were related to mental health. To understand what low or high scores mean in this context, please refer to Section 3.2.

4.2 Multiple Regression

In Table 6, CGP represents Computer Game Play in hours per week, for each Wave of the survey. Model 1 represents the results of the multiple regression using the full weight-adjusted sample, i.e., the original intended analysis. The effective sample size after removing missing values ("Refused to answer", "Don't know") was 2,395. After receiving results from this analysis, it was decided to attempt another analysis (Model 2) after removing all respondents who indicated that they didn't play computer games during at least one of the four Waves. Thus, Model 2 is an analysis of respondents who indicated that they played computer games in each Wave. These are long-term, consistent computer game players. The effective sample size for Model 2, after removing missing values ("Refused to answer", "Don't know") was 412. The results from multiple regression are similar for both Models.

Table 6: Multiple Regression Results

	Model 1		**Model 2**	
	Standardized Beta Coefficient	p-value	Standardized Beta Coefficient	p-value
CGP Wave 1	-.013	.540	-.044	.402
CGP Wave 2	-.016	.473	+.008	.877
CGP Wave 3	-.011	.604	+.023	.656
CGP Wave 4	+.080	.000***	+.180	.001***
	Model Sig.	.006	Model Sig.	.008
	R-square	.006	R-square	.033

In both Models in Table 6, only computer game play in Wave 4 was found to have a significant association (p-value less than 0.01) with mental health outcomes in adulthood. Here, mental health was coded in such a way that a positive Beta Coefficient would indicate worse mental health outcomes from increased computer game play. Thus, Model 1 may be interpreted as follows: A one hour increase in computer game play per week is associated with an 8% worse outcome in mental health (on a mental health scale of 0 to 3, where 3 is the worst mental health outcome). For Model 1 the model significance (0.006) is high, indicating that this regression model offers a better explanation than no model. Nevertheless, the R-square is low, implying that mental health outcomes in adulthood are associated with many factors, and computer game play is only one of them.

5 Discussion and Conclusions

Data were collected for the same cohort at different points in their life from adolescence to adulthood (at average cohort ages of 16 years, 18 years, 23 years, and 30 years). The results indicate that the amount of computer game play at 16 years, 18 years, or 23 years (average cohort age) is not significantly associated with mental health outcomes in adulthood (average cohort age of 30 years). However, computer game play at 30 years (average cohort age of 30 years) is significantly associated with negative mental health outcomes in adulthood (average cohort age of 30 years). In adulthood (average cohort age of 30 years), an increase in one hour of computer game play per week is associated with 8% worse mental health outcomes (in the mental health scale used here). Model 2 in Table 6,indicates that among adults (average cohort age of 30 years) who have consistently played computer games since High School, a one hour increase in computer game play per week is associated with 18% worse mental health outcomes using the same mental health scale.

In order to make advances in integrating prior disparate, inconsistent and inconclusive findings, this area of research (association between computer game play and mental health outcomes) needs long-term longitudinal studies, and a focus on mental health in adulthood. The research study presented here uses data collected over a period of 14 years, from the same cohort. Starting in adolescence with an average cohort age of 16 years and ending in adulthood with an average cohort age of 30 years, this is the longest term of any longitudinal cohort study related to computer game play and mental health made public thus far. In this way, this study makes a major contribution to the existing body of research.

This study is an association study that makes no claims about causality. Two theoretical perspectives, uses-and-gratifications and sociotechnical, were described earlier. They are useful in speculating about causality and its direction here. Uses-and-gratifications theory can be employed to speculate that adults in the average 30 year cohort age, who are dealing with negative mental health, are seeking out computer game play to receive emotional relief. And that the arrow of causality is from mental health to computer game play. Sociotechnical theory could be employed to speculate that computer games are typically designed for younger people. Hence adult consumers don't have a favorable balance of power in the design of computer games. And thus computer game play by adults leads to adverse mental health outcomes for them. These are just a couple of examples of how theories can inform speculation about causes and effects. A limitation of this study is that, in the data sample, the proportion that self-identified as being of Hispanic Origin is lower than in the current general population of the USA. Although it may be worth clarifying that this is likely because when the survey began in Wave 1, the proportion that self-identified as being of Hispanic Origin in the general population in the USA was lower than it is now.

Future research should incorporate variables such as gender, socio-economic status and education into our model to investigate whether the association found in this study varies for different categories of these variables. Future research should also strive to look beyond the USA, at international data, to see the extent to which the findings from this study can be universalized.

Technologies have become an integral part of our lives, shaping the way we interact, learn, and entertain ourselves. From nuclear power to artificial intelligence and social media, these advancements have proven to be double-edged swords, capable of bringing both blessings and curses to our society. Computer games, in particular, epitomize the dual nature of technology. On one hand, they can serve as a refuge for individuals seeking connection and community. However, the same technology that can foster connection and community can also be isolating. Some individuals may find themselves engulfed by the virtual world, losing touch with the real-life relationships and experiences that are essential for human well-being. Ultimately, it is our responsibility as individuals and as a society to navigate the potential for both good and harm, seeking ways to maximize the benefits while minimizing the negative consequences. More work needs to be done to enable prosocial aspects of computer games and to enable self-regulation and moderation in those who play them.

References

[1] Brandon K. Ashinoff. "The potential of video games as a pedagogical tool". In: *Frontiers in Psychology* 5 (2014). DOI: 10.3389/fpsyg.2014.01109.

[2] Sarah M. Coyne et al. "Pathological video game symptoms from adolescence to emerging adulthood: A 6-year longitudinal study of trajectories, predictors, and outcomes." In: *Developmental psychology* (2020). https://api.semanticscholar.org/CorpusID:218475490.

[3] Christopher J. Ferguson. "Do Angry Birds Make for Angry Children? A Meta-Analysis of Video Game Influences on Children's and Adolescents' Aggression, Mental Health, Prosocial Behavior, and Academic Performance". In: *Perspectives on Psychological Science* 10.5 (2015), pp. 646–666. DOI: 10.1177/1745691615592234.

[4] Christopher J. Ferguson, Mark Coulson, and Jane Barnett. "A meta-analysis of pathological gaming prevalence and comorbidity with mental health, academic and social problems". In: *Journal of Psychiatric Research* 45.12 (2011), pp. 1573–1578. DOI: 10.1016/j.jpsychires.2011.09.005.

[5] Bradley S. Greenberg et al. "Orientations to Video Games Among Gender and Age Groups". In: *Simulation & Gaming* 41.2 (2010), pp. 238–259. DOI: 10.1177/1046878108319930.

[6] Kathleen Mullan Harris et al. "Cohort Profile: The National Longitudinal Study of Adolescent to Adult Health (Add Health)". In: *International Journal of Epidemiology* 48.5 (June 2019), 1415–1415k. DOI: 10.1093/ije/dyz115.

[7] Abhijit Jain. "Extra-Curricular Computing Engagement, Academic Achievement, and Income Attainment: From Youth to Adulthood". In: *Journal of Computing Sciences in Colleges* 2 (29 Dec. 2013), pp. 105–12.

[8] Nor Shuhada Mansor, Chin Moi Chow, and Mark Halaki. "Cognitive effects of video games in older adults and their moderators: a systematic review with meta-analysis and meta-regression". In: *Aging & Mental Health* 24.6 (2020), pp. 841–856. DOI: 10.1080/13607863.2019.1574710.

[9] Max Marc. "Not Playing the Game: Negative Opinions about Online Dating and Video Gaming among Non-Participants". In: *Journal For Virtual Worlds Research* 10 (Sept. 2017). DOI: 10.4101/jvwr.v10i2.7273.

[10] Julio Angel Ortiz. "Re-Gaming the Digital Divide: Broadband, MMOGs and U.S. Latinos". In: 2010. URL: https://api.semanticscholar.org/CorpusID:59376366.

[11] John B. Saunders et al. "Gaming disorder: Its delineation as an important condition for diagnosis, management, and prevention". In: *Journal of Behavioral Addictions* 6.3 (2017), pp. 271–279. DOI: 10.1556/2006.6.2017.039.

[12] Timothy Sim et al. "A Conceptual Review of Research on the Pathological Use of Computers, Video Games, and the Internet". In: *International Journal of Mental Health and Addiction* 10.5 (2012), pp. 748–769. DOI: 10.1007/s11469-011-9369-7.

[13] Census Bureau United States. *Table 4. Hispanic or Latino Origin by Race: 2010 and 2020*. https://www.census.gov/data/tables/2020/dec/2020-redistricting-supplementary-tables.html.

Developing Identity-Focused Program-Level Learning Outcomes for Liberal Arts Computing Programs*

Conference Tutorial

Jakob Barnard[1], Grant Braught[2], Janet Davis[3], Amanda Holland-Minkley[4], David Reed[5], Karl Schmitt[6], Andrea Tartaro[7], James Teresco[8]
[1]University of Jamestown, Jamestown, ND 58405
Jakob.Barnard@uj.edu
[2]Dickinson College, Carlisle, PA 17013
braught@dickinson.edu
[3]Whitman College, Walla Walla, WA 99362
davisj@whitman.edu
[4]Washington & Jefferson College, Washington, PA 15317
ahollandminkley@washjeff.edu
[5]Creighton University, Omaha, NE 68178
DaveReed@creighton.edu
[6]Trinity Christian College, Palos Heights, IL 60463
Karl.Schmitt@trnty.edu
[7]Furman University, Greenville, SC 29690
andrea.tartaro@furman.edu
[8]Siena College, Loudonville, NY 12211
jteresco@siena.edu

The SIGCSE Committee on Computing Education in Liberal Arts Colleges (SIGCSE-LAC Committee) has found that liberal arts and small colleges approach design of their computing curricula in unique ways that are driven by institutional mission or departmental identity. This impacts how faculty at these colleges adopt curricular guidelines such as the current ACM/IEEE-CS CS2013[2]. The committee is developing guidance, informed by its sessions at recent CCSC and SIGCSE conferences, to help with the design and/or revision of CS curricula in liberal arts contexts[1]. This will ultimately be included in the committee's article in the Curricular Practices Volume that will be released as a companion to the new ACM/IEEE-CS/AAAI Computer Science Curricula guidelines, CS2023 (`https://csed.acm.org`). Curricular guidelines like CS2013 or CS2023 inform curriculum design, but are balanced with the vision for a program, departmental strengths, locale, student populations and unique academic experiences. The desire to craft distinctive curricula, combined with the size of prior curricular recommendations, requires an assessment of trade-offs between achieving full coverage of curricular recommendations and a school's other priorities. SIGCSE-LAC's guidance will encourage faculty to reflect on their programs and the role of CS2023, beginning with their institutional and departmental priorities, opportunities and constraints.

The specific goal of this session is to help participants develop program-level learning outcomes that align with the unique features of their programs. Following an overview and brief discussion of the newest CS2023 draft, participants will begin working through a preliminary version of the committee's reflective assessment process. This process is framed by a series of scaffolding questions that begin from institutional and departmental missions, identities, contexts, priorities, initiatives, opportunities, and constraints. From there, participants will be led to identify design principles for guiding their curricular choices, including the CS2023 recommendations. Examples gathered from the committee's previous CCSC and SIGCSE sessions will be available to help to articulate identity and program design principles, which will then be used for the identification of identity-focused program-level learning outcomes. Participants will leave the session with a better understanding of how CS2023 can impact their programs and a jumpstart on the entire reflective assessment process. Feedback on the process and this session are welcome and will be used to refine the committee's guidance prior to its publication in the CS2023 Curricular Practices volume.

Presenter Biography

Janet Davis is Microsoft Chair and Professor of Computer Science at Whitman College, where she serves as the department's founding chair. She co-organized SIGCSE pre-symposium events in 2020 and 2021 on behalf of the SIGCSE-LAC Committee.

Biographies of Other Authors

Jakob Barnard is Chair and Assistant Professor of Computer Science & Technology at the University of Jamestown. He is a member of the SIGCSE-LAC Committee and his research involves how curricula has been integrated into Liberal Arts Technology programs. **Grant Braught** is a Professor of Computer Science at Dickinson College. He is a facilitating member of the SIGCSE-LAC Committee, has organized committee events focused on curricula and has published widely on issues related to CS education, particularly within the liberal arts. **Amanda Holland-Minkley** is a Professor of Computing and Information Studies at Washington & Jefferson College. Her research explores novel applications of problem-based pedagogies to CS education at the course and curricular level. She is a facilitating member of the SIGCSE-LAC Committee. **David Reed** is a Professor of Computer Science and Chair of the Department of Computer Science, Design & Journalism at Creighton University. He has published widely in CS education, including the text *A Balanced Introduction to Computer Science*, and served on the CS2013 Computer Science Curricula Task Force. **Karl Schmitt** is Chair and Associate Professor of Computing and Data Analytics at Trinity Christian College. He has served on the ACM Data Science Task Force and various Computing, Technology, Mathematics Education related committees for the MAA, ASA and SIAM. His interests explore data science education, and interdisciplinary education between computing, mathematics, data, and other fields. **Andrea Tartaro** is an Associate Professor of Computer Science at Furman University. Her computer science education research focuses on the intersections and reciprocal contributions of computer science and the liberal arts, with a focus on broadening participation. She is a member of the SIGCSE-LAC Committee, and has published and presented in venues including the CCSC and the SIGCSE Technical Symposium. **Jim Teresco** is a Professor of Computer Science at Siena College. He has been involved in CCSC Northeastern for almost 20 years and currently serves as board chair, and has been involved with the SIGCSE-LAC Committee for 4 years. His research involves map-based algorithm visualization.

References

[1] Amanda Holland-Minkley, Jakob Barnard, Valerie Barr, Grant Braught, Janet Davis, David Reed, Karl Schmitt, Andrea Tartaro, and James D. Teresco. Computer science curriculum guidelines: A new liberal arts perspective. In *Proceedings of the 54th ACM Technical Symposium on Computer Science Education V. 1*, SIGCSE 2023, page 617–623, New York, NY, USA, 2023. ACM.

[2] Association for Computing Machinery (ACM) & IEEE Computer Society Joint Task Force on Computing Curricula. Computer science curricula 2013: Curriculum guidelines for undergraduate degree programs in computer science, 2013. `https://www.acm.org/binaries/content/assets/education/cs2013_web_final.pdf`.

Getting Started on Jetstream2*

Conference Tutorial

Zachary Graber and Daniel Havert
Research Technologies
Indiana University
Bloomington, IN

As research and education advance, so does their need for advanced computational resources. While some universities are fortunate to be able to provide these resources in abundance, many do not have free availability to such cyberinfrastructure for their research, much less for their instruction. Through Advanced Cyberinfrastructure Coordination Ecosystem: Services & Support (ACCESS), advanced computing resources such as Jetstream2 are shared with educators for free. This sharing of resources provides access to educators who normally would not have access to such platforms.

Jetstream2[1] is an NSF-funded, user-friendly cloud computing environment for researchers and educators running on OpenStack and featuring Exosphere as the primary user interface. Jetstream2 is built on the successes of Jetstream1, continuing the main features of that system and extending to a broader range of hardware and services, including GPUs, large memory nodes, virtual clustering, and other features. It is designed to provide both infrastructure for gateways and other "always on" services, as well as to give researchers and educators access to interactive computing and data analysis resources on demand. One of the goals of providing such a resource without cost is to give colleges and universities access to these resources not only for research but also for instruction, thereby democratizing cloud computing for educators.

Tutorial Audience and Details

This tutorial targets an audience of educators and researchers. Attendees will get an overview of Jetstream2, the ACCESS ecosystem, and how to get on Jetstream2, with a walk through of how to access and launch VMs on Jetstream2 via the Exosphere interface. It will provide various examples and use cases of Jetstream2 for instruction, along with other helpful tips and tricks.

Tutorial Session Requirements

- A computer with internet access.
- An ACCESS account. Can be created for free at: `https://identity.access-ci.org/new-user`
- After you create an ACCESS account, fill in the google form at: `https://forms.gle/dNwn7sj9CBfLyGev5` to let us know your ACCESS username, so we can add you to a special training allocation and you can follow along with the tutorial.

Acknowledgements

This material is based upon work supported by the National Science Foundation under Grant 2005506. 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.

Biography

Zachary Graber is part of the Research Cloud Services team in the Research Technologies division of Indiana University (IU) that supports Jetstream2. He received his bachelor's degree in Computer Science from IU's Luddy School of Informatics, Computing, and Engineering.

Daniel Havert is part of the Research Cloud Services team in the Research Technologies division of Indiana University that supports Jetstream2. He received his bachelor's degree in Physics from Embry-Riddle Aeronautical University and is currently completing a PhD in Physics at Indiana University. His interests include cloud computing, artificial intelligence, and educational outreach.

References

[1] David Y. Hancock et al. "Jetstream2: Accelerating Cloud Computing via Jetstream". In: *Practice and Experience in Advanced Research Computing*. PEARC '21. Boston, MA: Association for Computing Machinery, 2021. DOI: 10.1145/3437359.3465565.

How to Install and Use a Security Onion NIDS VM in a Defensive Cybersecurity Course*

Conference Tutorial

Mohamed Lotfy
Utah Valley University
Orem, UT 84058
MohamedL@uvu.edu

To prepare both IT and cybersecurity graduates and to meet industry needs, cybersecurity courses must introduce current offensive and defensive tools and practices to secure computing resources, systems, services, data, and network services. These current offensive and defensive cybersecurity tools should be introduced and applied in hands-on activities, thus allowing students to gain the needed knowledge of current cybersecurity best practices[2, 1]. The hands-on approach allows students to develop defensive cybersecurity competency, enabling them to build layered defenses that harden systems to advanced persistent threats. Teaching defensive cybersecurity practices requires an environment where attacking vulnerable network intrusion detection systems (NIDS), host intrusion detection systems (HIDS), and honeypots hosts are used to perform the different defensive actions to the phases of the cyber kill chain on vulnerable hosts. Using an encapsulated virtual environment, where the different attacks on vulnerable hosts can be conducted, reduces the risk to institutional networks and systems.

Tutorial Description

In this tutorial the presenter will provide a hands-on working example of how to install a Security Onion (SO) NIDS/HIDS VM[5] on a testing environment, using either VMware Workstation Pro[6] or Oracle VM VirtualBox[4] on laptops or PCs. The virtual environment will include a SO VM, an offensive security Kali Linux VM, and a Linux VM. The presenter will show how to write rules to detect ICMP and TCP packets, different file transfers, which cause alerts on the SO NIDS dashboards.

Tutorial program

In the tutorial the presenter will illustrate and explain the following:

1. How to install a SO NIDS/HIDS VM in VMware and/or VirtualBox.
2. Describe the format and structure of Snort rules.
3. How to write Snort rules to detect ICMP and TCP traffic on the subnet.
4. How to write Snort rules to detect different file formats.
5. How to craft special packets using the hping3 command.
6. How to see the alerts on the SO dashboards.

Expected outcomes

Attendees will exit the tutorial with a working VMware or VirtualBox environment and learn how to use the SO NIDS system to detect packets and different file transfers on the subnet.

Target audience

Any faculty who would like to incorporate a VMware or VirtualBox virtual environment and use SO NIDS in a defensive cybersecurity course.

Prerequisites

Attendees should be familiar with Linux, networking, and some programming knowledge (Java, C++, Python, etc.). It is highly recommended that attendees bring their own laptops with VMware[6] or VirtualBox[4] and a Kali Linux VM[3] installed.

References

[1] Pooneh Nikkhah Bahrami, Ali Dehghantanha, Tooska Dargahi, Reza M Parizi, Kim-Kwang Raymond Choo, and Hamid HS Javadi. Cyber kill chain-based taxonomy of advanced persistent threat actors: Analogy of tactics, techniques, and procedures. *Journal of information processing systems*, 15(4):865–889, 2019.

[2] Eric M Hutchins, Michael J Cloppert, and Rohan M Amin. Intelligence-driven computer network defense informed by analysis of adversary campaigns and intrusion kill chains. *Leading Issues in Information Warfare & Security Research*, 1(1):80, 2011.

[3] OffSec Services Limited. KALI pre-built virtual machines, 2023. URL: `https://www.kali.org/get-kali/#kali-virtual-machines`.

[4] ORACLE. ORACLE VM VirtualBox, 2023. URL: `https://www.virtualbox.org/`.

[5] Security Onion Solutions, LLC. Security Onion, 2023. URL: `https://securityonionsolutions.com/`.

[6] VMware. VMware Workstation Pro, 2023. URL: `https://www.vmware.com`.

Teaching Global and Ethical Perspectives in Information Technology*

Cynthia Krebs, Jan Bentley and DeDe Smith
Utah Valley University
Orem, UT 84058
{cynthia.krebs, jan.bentley, smithdo}@uvu.edu

Information technology and computer science practitioners make ethical decisions every day. To aid students as they make ethical decisions, the Information Systems and Technology (IS&T) Department at Utah Valley University offers a "Global, Ethical, & Professional Perspectives in Information Technology" course, a required 4000-level course for IS&T students. In this course students identify and express their own values, explore the values of others in a global community, analyze case studies and apply ethical decision-making standards, and learn to ethically resolve differences. This course offers a global perspective to students as they examine current ethical issues within information systems and technology fields.

Tutorial Description

In this tutorial, the presenters will discuss the course's focus on ethics and ethical relationships, examine the Portulans Institute's Global Networked Readiness Index, and explore challenges that today's students face in a global environment. The presenters will cover strategies used for teaching ethics with a global awareness focus. Presenters will cover methods for teaching students to use critical thinking skills to examine ethical and legal issues related to freedom of expression, social networking, global etiquette, cybercrime, intellectual property, and software development. Other methods to be discussed include case studies, guest speakers, media coverage, readings, role-play, pair and share, papers, and worksheets.

Tutorial program

In the tutorial the presenters will cover methods used to teach the following:

1. An overview of ethics
2. Ethical problem solving
3. Global Network Readiness issues
4. Privacy issues
5. Freedom of expression issues
6. Other ethics issues

Expected outcomes

Attendees will exit the tutorial with a framework for teaching ethics with a global perspective to information technology and computer science students.

Target audience

Any faculty who would like to introduce a more formalized ethics program in a higher education information technology or computer science course.

Prerequisites

No prerequisites necessary.

References

[1] A decade on, Edward Snowden remains in Russia, though U.S. laws have changed, 2023. Visited June 4, 2023. URL: https://www.npr.org/2023/06/04/1176747650/a-decade-on-edward-snowden-remains-in-russia-though-u-s-laws-have-changed.

[2] A framework for ethical decision making, 2021. URL: https://www.scu.edu/ethics/ethics-resources/a-framework-for-ethical-decision-making/.

[3] Global Network Readiness Index (NRI), 2023. Visited July 27, 2023. URL: https://portulansinstitute.org/.

[4] Sydnee Gonzalez. Navajo president says 'Oppenheimer' film erases how nuclear testing impacted his people, 2023. Visited July 26, 2023. URL: https://www.ksl.com/article/50694964/navajo-president-says-oppenheimer-film-erases-how-nuclear-testing-impacted-his-people.

[5] Carmen Nesbitt. Utah voucher boosters are promoting a 'misleading' website, state education officials say, 2023. Visited July 26, 2023. URL: https://www.sltrib.com/news/education/2023/07/26/misleading-website-invites-utahns/.

[6] George Reynolds. *Ethics in information technology*. Cengage learning, 2019.

[7] Gai Sher and Ariela Benchlouchl. Unmasking AI bias: a collaborative effort, 2023. Visited July 21, 2023. URL: https://www.reuters.com/legal/legalindustry/unmasking-ai-bias-collaborative-effort-2023-07-21/.

[8] Heather Widdows. *Global ethics: An introduction*. Routledge, 2014.

Incorporating Computing for the Social Good Into the Classroom*

Conference Workshop

Johanna Blumenthal and Richard Blumenthal
Department of Computer and Cyber Sciences
Regis University
Denver, CO 80221
{jblumenthal, rblument}@regis.edu

This workshop will provide participants with hands-on learning experiences to familiarize them with the Computing for Social Good in Education (CSG-Ed) community and the methods CSG-Ed uses to create socially relevant classroom computing activities. CSG-Ed is an umbrella term meant to incorporate any educational activity, from small to large, that endeavors to convey and reinforce computing's social relevance and potential for positive societal impact[1, 2]. Incorporating CSG-Ed into classroom activities across a computing curriculum addresses a range of desirable goals including:

- Support the ethical and societal learning outcomes specified in the ACM curricular recommendations (e.g., Computer Science, Cybersecurity, Data Science, Information Systems, and Software Engineering), as well as in ABET accreditation and the Seoul Accord objectives.
- Encourage computing students to use their computing education towards the benefit of society (e.g., climate change, world hunger, etc.)
- Increase enrollments by focusing on students who want to make socially relevant contributions to our communities and our world. For example, expanding diversity and inclusion, since research suggests: student orientation towards social activism has been a consistent negative predictor of interest in computing over the past 40+ years, women place greater emphasis on social activism than men, and choice of major is influenced by one's values rather than one's interests[3].

The workshop is designed to demonstrate the ease in which educators can incorporate various CSG activities into their pedagogy. This will be reinforced by having participants use CSG-Ed methods to create student-focused exercises during the workshop, which can be taken back to their home universities. The workshop also focuses on growing the CSG-Ed community, by connecting participants with each other and existing members of the community.

Workshop Agenda

The agenda for the workshop is:

- Introductions: facilitators and participants
- CSG-Ed Overview: a brief history and current activities
- CSG-Ed Methods: introduce the CSG-Ed taxonomy and design patterns
- Activities: participants will use the CSG-Ed methods to create socially relevant computing activities for use in their own classrooms. Participants will also present these CSG activities to the group.
- Wrap up and discussion of next steps.

The workshop will also provide ample opportunities for participants to discuss existing and proposed CSG activities, to gain additional insight from the facilitators and other workshop participants. Workshop participants will only need pencil and paper.

Authors Information

The authors are active in the CSG-Ed community and have led several CSG-Ed focused tutorial sessions for educators in computing and related disciplines at the ACM Technical Symposium on Computer Science Education conferences (SIGCSE) and Data for Good for Education workshops.

References

[1] Michael Goldweber, John Bar, and Tony Clear. *ACM Inroads*, 4(1):58–79, 2012.

[2] Michael Goldweber, Lisa Kaczmarczyk, and Richard Blumenthal. Computing for the social good in education. *ACM Inroads*, 10(4):24–29, 2019.

[3] Linda J. Sax, Katheleen J. Lehman, Jerry A. Jacobs, M. Allison Kanny, Gloria Lim, Laura Monje-Paulson, and Hilary B. Zimmerman. Anatomy of an enduring gender gap: The evolution of women's participation in computer science. *The Journal of Higher Education*, 88(2):258–293, 2017.

Platform-Free Mobile Application: Chatbot That Uses ChatGPT*

Poster Abstract

Marcos Pinto
NYC College of Technology
Brooklyn, NY 11201
mpinto@citytech.cuny.edu

This poster presents a mobile application that allows any person to communicate with a chatbot via a mobile device using natural language, and get replies from it on a variety of topics such as business ideas, freelancing, blogging, email marketing, essay creation, coding, ebooks, etc. The use of ChatGPT as the backend chatbot is justified by the increasing popularity of this chatbot, which has been currently hailed as the most important chatbot that makes use of artificial intelligence to harvest and curate data from several data sources.

Teaching an Undergraduate Computer Graphics Elective Course: Lessons Learned*

Poster Abstract

George Thomas
University of Wisconsin Oshkosh
Oshkosh, WI 54956
thomasg@uwosh.edu

The challenge of teaching introductory undergraduate Computer Graphics involves both covering the theoretical and practical components of an unusually large diversity of topics in a limited amount of time, and implementing concepts in a multi-layered programming infrastructure that can also exploit the advances in modern hardware. Our experience teaching such a course in a purely lecture-based format led us to introduce a lab-based component into a subsequent version of this course. Our original version of the course consisted of three hours of lecture, with short review problems assigned every lecture, accompanied by 5-6 programming projects. This poster will present our modified version of the course, which introduced carefully structured weekly labs that complemented the lecture material for that week, but eliminated one hour of lecture per week. The programming projects were replaced by two large projects, with the second project being a team-based game development project, where teams were free to design a game of their choosing within certain broad parameters. High-level quantitative student feedback indicated a 4% increase in students who felt that the course activities aided their learning. In addition, while there was some concern about the density of topic coverage in the now reduced two-hours of lecture per week, student comments about the labs were overwhelmingly positive, with many students remarking on how the labs helped them better understand course concepts. In future versions of this course, we plan to restructure delivery of the course content while retaining and improving the labs as an integral part of the course.

Reviewers — 2023 CCSC Rocky Mountain Conference