The Journal of Computing Sciences in Colleges

Papers of the 27th Annual CCSC Central Plains Conference

April 9th-10th, 2021
University of Missouri Kansas City
Kansas City, MO

Baochuan Lu, Editor
Southwest Baptist University

Bin Peng, Regional Editor
Park University

Volume 36, Number 6 **April 2021**

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 **5**

CCSC National Partners **7**

Welcome to the 2021 CCSC Central Plains Conference **8**

Regional Committees — 2021 CCSC Central Plains Region **9**

Reviewers — 2021 CCSC Central Plains Conference **11**

The Next Generation of Technology Leaders — Keynote **12**
Dan Zimmerman, MSTS

Discovering Enterprise Architecture Developing a Course With Enterprise Application Software Tools **14**
Maria Weber, Kyle Chapman, John Buerck, Saint Louis University

Teach Like a Git: Streamlining Assignment Administration and Enforcing Good Habits With Professional Tools and Software Development Practices **27**
Nathan W. Eloe, Northwest Missouri State University

Open-Source Scholarship at Teaching-Oriented Institutions **37**
Joseph Kendall-Morwick, Missouri Western State University

Fourth Hour: A CS1 Review Session Led by Teaching Assistants Using Peer Instruction **45**
Megan Gilbert, Dee A.B. Weikle, Chris Mayfield, Chris Johnson, James Madison University

Infusion of Machine Learning Concepts and Skills: A Workshop Review **55**
Sambit Bhattacharya, Bogdan Czejdo, Valentin Milanov, Fayetteville State University

Hey Alexa! A Fun Introduction to AWS Lambda, Serverless Functions, and JavaScript with Conversational AI — Conference Workshop/Tutorial **66**
Denise M Case, Northwest Missouri State University

Short Modules for Introducing Heterogeneous Computing — Conference Workshop/Tutorial **67**
David P. Bunde, Knox College; Apan Qasem, Texas State University; Philip Schielke, Concordia University Texas

How Can It Be Fixed in Terms of Typing? — Nifty Assignment **69**
Cong-Cong Xing, Nicholls State University; Jun Huang, Northwest Polytechnic University

Creating a Server as a Nifty Assignment — Nifty Assignment **71**
Samuel Hsieh, Sean Wolfe, Ball State University

The Role of Technology in Digital Forensics Investigations — Nifty Assignment **74**
Rad Alrifai, Northeastern State University

Engaging Graduate Students During the Pandemic — Panel Discussion **78**
Charles Badami, Ajay Bandi, Denise Case, Aziz Fellah, Northwest Missouri State University; Mahmoud Yousef, University of Central Missouri

The Impact of COVID-19 on Industry and Education — Panel Discussion **80**
Joni Adkins, Diana Linville, Northwest Missouri State University; Bilal Clarence, Capital One Financial; Shanon Elliott, Waddell and Reed; Sean Paddock, Federal Reserve Bank of Kansas City

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:

Karina Assiter, President (2022), (802)387-7112, karinaassiter@landmark.edu.

Chris Healy, Vice President (2022), chris.healy@furman.edu, Computer Science Department, 3300 Poinsett Highway Greenville, SC 29613.

Baochuan Lu, Publications Chair (2021), (417)328-1676, blu@sbuniv.edu, Southwest Baptist University - Department of Computer and Information Sciences, 1600 University Ave., Bolivar, MO 65613.

Brian Hare, Treasurer (2020), (816)235-2362, hareb@umkc.edu, University of Missouri-Kansas City, School of Computing & Engineering, 450E Flarsheim Hall, 5110 Rockhill Rd., Kansas City MO 64110.

Cathy Bareiss, Membership Secretary (2022), cathy.bareiss@betheluniversity.edu, Department of Mathematical Engineering Sciences, 1001 Bethel Circle, Mishawaka, IN 46545.

Judy Mullins, Central Plains Representative (2023), Associate Treasurer, (816)390-4386, mullinsj@umkc.edu, UMKC, Retired.

Michael Flinn, Eastern Representative (2023), mflinn@frostburg.edu, Department of Computer Science Information Technologies, Frostburg State University, 101 Braddock Road, Frostburg, MD 21532.

David R. Naugler, Midsouth Representative(2022), (317) 456-2125, dnaugler@semo.edu, 5293 Green Hills Drive, Brownsburg IN 46112.

Grace Mirsky, Midwest Representative(2023), gmirsky@ben.edu, Mathematical and Computational Sciences, 5700 College Rd. Lisle, IL 60532.

Lawrence D'Antonio, Northeastern Representative (2022), (201)684-7714, ldant@ramapo.edu, Computer Science Department, Ramapo College of New Jersey, Mahwah, NJ 07430.

Shereen Khoja, Northwestern Representative(2021), shereen@pacificu.edu, Computer Science, 2043 College Way, Forest Grove, OR 97116.

Mohamed Lotfy, Rocky Mountain Representative (2022), Information Systems & Technology Department, College of Engineering & Technology, Utah Valley University, Orem, UT 84058.

Tina Johnson, South Central Representative (2021), (940)397-6201, tina.johnson@mwsu.edu, Dept. of Computer Science, Midwestern State University, 3410 Taft Boulevard, Wichita Falls, TX 76308.

Kevin Treu, Southeastern Representative (2021), (864)294-3220, kevin.treu@furman.edu, Furman University, Dept of Computer Science, Greenville, SC 29613.

Bryan Dixon, Southwestern Representative (2023), (530)898-4864,

bcdixon@csuchico.edu, Computer Science Department, California State University, Chico, Chico, CA 95929-0410.

Serving the CCSC: These members are serving in positions as indicated:

Bin Peng, Associate Editor, (816) 584-6884, bin.peng@park.edu, Park University - Department of Computer Science and Information Systems, 8700 NW River Park Drive, Parkville, MO 64152.

Shereen Khoja, Comptroller, (503)352-2008, shereen@pacificu.edu, MSC 2615, Pacific University, Forest Grove, OR 97116.

Elizabeth Adams, National Partners Chair, adamses@jmu.edu, James Madison University, 11520 Lockhart Place, Silver Spring, MD 20902.

Megan Thomas, Membership System Administrator, (209)667-3584, mthomas@cs.csustan.edu, Dept. of Computer Science, CSU Stanislaus, One University Circle, Turlock, CA 95382.

Deborah Hwang, Webmaster, (812)488-2193, hwang@evansville.edu, Electrical Engr. & Computer Science, University of Evansville, 1800 Lincoln Ave., Evansville, IN 47722.

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.

Platinum Partner

Turingscraft
Google for Education
GitHub
NSF – National Science Foundation

Silver Partners

zyBooks

Bronze Partners

National Center for Women and Information Technology
Teradata
Mercury Learning and Information
Mercy College

Welcome to the 2021 CCSC Central Plains Conference

The last year has posed a series of challenges, as educators, as computing professionals, and as citizens. The COVID-19 pandemic has forced us to adapt quickly, moving instruction online to a degree considered improbable a few years ago. The news continues to report amazing breakthroughs in artificial intelligence, and frightening tales of built-in biases, flaws, and abuses of such technology. Preparing our students to become responsible professionals in such an environment presents unique challenges.

The CCSC Central Plains Conference hopes to contribute toward preparing ourselves to help prepare our students. We have panels and presentations on a variety of topics; of 10 papers submitted, we accepted the best 5, an acceptance rate of 50%. I had hoped to use the conference to present UMKC's new engineering lab building, the Robert W. Plaster Free Enterprise & Research Center, including the high-performance computing lab. But as the pandemic has forced the conference online just as much as our classes, an online presentation will have to do.

I want to thank the members of our regional steering committee, who have done the majority of the actual work for the conference; the Central Plains region has always enjoyed a high level of participation from faculty across many schools in the region, and it has been a pleasure to work with them in preparing this years conference.

I hope you find the conference as rewarding as we have found preparing it.

Brian Hare
University of Missouri-Kansas City
Conference Chair

2021 CCSC Central Plains Conference Steering Committee

Conference Chair
Brian Hare ..... University of Missouri Kansas City
Conference Publicity
Michael P. Rogers ..... University of Wisconsin Oshkosh
Bin Peng ..... Park Universty
Keynote Speakers
Brian Hare ..... University of Missouri Kansas City
Scott Sigman ..... Drury University
Pre-Conference Workshop
Judy Mullins ..... Retired
Michael P. Rogers ..... University of Wisconsin Oshkosh
Wen Hsin ..... Park University
Papers, Panels, Tutorials, Workshops
Scott Bell ..... Northwest Missouri State University
Ron McCleary ..... Retired
Nifty Assignments
Mahmoud Yousef ..... University of Central Missouri
Michael P. Rogers ..... University of Wisconsin Oshkosh
Lightning Talks
Diana Linville ..... Northwest Missouri State University
Kendall Bingham ..... University of Missouri Kansas City
K-12 Outreach
Mahmoud Yousef ..... University of Central Missouri
Belinda Copus ..... University of Central Missouri
K-12 Nifty Assignments & Lightning Talks
Belinda Copus ..... University of Central Missouri
Student Paper Session
Scott Sigman ..... Drury University
Student Poster Competition
Joseph Kendall-Morwick ..... Missouri Western University
Student Programming Contest
Charles Riedesel ..... University of Nebraska-Lincoln
Dayu Wang ..... St. Charles Community College
Two-Year College Outreach
Rex McKanry ..... St. Charles Community College
Belinda Copus ..... University of Central Missouri
Career Fair

Charles Riedesel ..... University of Nebraska-Lincoln
Rex McKanry ..... St. Charles Community College

Zoom Tech

Rex McKanry ..... St. Charles Community College
Jennifer McKanry ..... University of Missouri St Louis
Michael P. Rogers ..... University of Wisconsin Oshkosh
Kendall Bingham ..... University of Missouri Kansas City

Regional Board — 2021 CCSC Central Plains Region

Regional Rep & Board Chair

Judy Mullins ..... Retired

Registrar & Membership Chair

Ron McCleary ..... Retired

Current Conference Chair

Brian Hare ..... University of Missouri Kansas City

Next Conference Chair

Scott Sigman ..... Drury University

Past Conference Chair

Dayu Wang, Deepika Jagmohan ..... St. Charles Community College

Secretary

Diana Linville ..... Northwest Missouri State University

Regional Treasurer

Denise Case ..... Northwest Missouri State University

Regional Editor

Bin Peng ..... Park University

Webmaster

Michael P. Rogers ..... University of Wisconsin Oshkosh

Reviewers — 2021 CCSC Central Plains Conference

Rad Alrifai ..... Northeastern State University, Tahlequah, OK
Beth Arrowsmith ..... University of Missouri - St. Louis, Saint Peters, MO
Ajay Bandi ..... Northwest Missouri State University, Maryville, MO
Sambit Bhattacharya ..... Fayetteville State University, Fayetteville, NC
John Buerck ..... Saint Louis University, St. Louis, MO
Denise Case ..... Northwest Missouri State University, Maryville, MO
Chia-Chu Chiang ..... University of Arkansas at Little Rock, Little Rock, AR
George Dimitoglou ..... Hood College, Frederick, MD
Ernest Ferguson ..... Northwest Missouri State University, Maryville, MO
David Furcy ..... University of Wisconsin Oshkosh, Oshkosh, WI
David Heise ..... Lincoln University, Jefferson City, MO
Suvineetha Herath ..... Carl Sandburg College, Galesburg, IL
Wen Hsin ..... Park University, Parkville, MO
Joseph Kendall-Morwick ... Missouri Western State University, Lawrence, KS
Srinivasarao Krishnaprasad ... Jacksonville State University, Jacksonville, AL
Baochuan Lu ..... Southwest Baptist University, Bolivar, MO
Jose Metrolho ..... Polytechnic Institute of Castelo Branco, Castelo Branco, Portugal
Muath Obaidat ..... LaGuardia Community College, Long Island City, NY
Michael Oudshoorn ..... High Point University, High Point, NC
Hassan Pournaghshband ..... Kennesaw State University, Kennesaw, GA
Zhengrui Qin ..... Northwest Missouri State University, Maryville, MO
Michael Rogers ..... University of Wisconsin Oshkosh, Oshkosh, WI
Jamil Saquer ..... Missouri State University, Springfield, MO
Cecil Schmidt ..... Washburn University, Topeka, KS
William Siever ..... Washington University, St. Louis, MO
Scott Sigman ..... Drury University, Springfield, MO
Timothy Urness ..... Drake University, Des Moines, IA
Ken Vollmar ..... Missouri State University, Springfield, MO
Henry Walker ..... Grinnell College, Grinnell, IA
Baoqiang Yan ..... Missouri Western State University, Saint Joseph, MO

The Next Generation of Technology Leaders*

Keynote

Dan Zimmerman, MSTS

Abstract

Every month there is another technology-based business with a multi-billion-dollar valuation. 60% of the NASDAQ 100 are Technology Services, Electronic Technology, or Health Technology businesses. Even businesses that aren't classified as technology business are relying on technology to deliver a higher percentage of their value proposition. This reality has led to the now often repeated phrase, "every company is a tech company". So, who should be leading all of these tech companies? MBAs or CS grads? The next generation of technology leaders won't be just "running IT". Technology leaders are driving value creation. In the near future creative, business savvy technologists will be the number one or number two person in virtually every company. How can Universities and Colleges help grow the next generation of technology leaders and cultivate a new stereotype of computer scientist. The next generation of technology leaders are also the next generation of business leaders.

Bio

Dan Zimmerman is the Chief Product and Technology Officer at MSTS, responsible for product strategy, innovation, and architecture for the company's software product-lines. Dan has over 25 years of experience leading software engineering teams and product management. He's led the launches of multiple SaaS applications, three successful agile transformations and won two CIO

100 awards. Dan excels at creating alignment within organizations to design cultures of transparency and optimization.

Prior to MSTS, Dan held senior roles at Nordstrom Bank, TSYS, and Western Union. Dans been a speaker at multiple conferences including Finovate and the KC IT Symposium and has been published in IDG Connect, Inside Big Data, and The Startup.

Discovering Enterprise Architecture Developing a Course With Enterprise Application Software Tools*

Maria Weber, Kyle Chapman, John Buerck
School of Professional Studies
Saint Louis University
Saint Louis, MO 63108
{maria.l.weber, kyle.chapman, john.buerck}@slu.edu

Abstract

Information Technology (IT) and Digital Transformation (Dx) are shifting how companies function and embrace emerging technologies. Companies are reimagining how work can be done by adopting new technologies to existing or new infrastructure. Organizations need skilled IT professionals who can drive this transformation, leading to the establishment of a robust enterprise. Industry trends indicate a need to shift academic curricula in Information System (IS) programs. Enterprise Architecture (EA) is a course in a graduate IS program at Saint Louis University. The EA curriculum focuses on the alignment between IT and business. The course covers designing, planning, and implementing organizational changes to corporate architecture through standardization of principles, models, and procedures. The use of EA open-source tools in this course is innovative and ideal for building students' marketable skills and improving their university experience. This paper provides an overview of designing and delivering a gateway course in Enterprise Architecture and Infrastructure Systems using enterprise-class and open-source software tools as part of the Master of Information Systems.

1 Introduction

Information technology (IT) is one of the most dynamic and developing businesses in the world. Digital transformation (Dx) and the convergence of digital technologies such as virtualization, IoT, big data, and cloud computing have redefined how we do business today. Employers require Information Systems (IS) professionals to execute the Dx toward efforts that advance their business. The US Bureau of Labor Statistics predicts a growth of 11 percent in IT jobs in the next ten years, with an expectation of around 1.3 million IT job openings in the US by 2026 [6]. Additionally, Microsoft CELA Data Science predicts that by 2025, many technology-oriented jobs will be created, with 13 million in the US and 149 million worldwide [4].

Job growth is promising, and higher-than-average starting salaries are expected for Computer Science (CS) and IS professionals; therefore, these majors have become more popular among college students [3]. Most of the fastest-growing IS/CS occupations require a bachelor's degree and experience in the related field. Students in these fields are often exposed to developing technologies, such as artificial intelligence, cloud computing, and coding, to help them gain more marketable skills during their college experience. Higher education institutions are trying to tailor their programs to meet industry needs, in-demand job skills, and utilize enterprise tools to train students with the skills needed to land a job. Universities offering IT degrees are becoming more innovative in making graduate students more marketable for job seekers due to increased competition for quality employees [5].

A new online graduate program in Information Systems started in Fall 2020 at Saint Louis University, which provides flexible education for adult learners pursuing an advanced degree. The Master of Information Systems program has been designed to combine skills from business and technology. Graduates of the IS program will be prepared for advanced leadership roles to drive business innovation and value while improving operations through current and future information technologies. These prospective enterprise leaders will apply their analytical skills to determine organizational needs, implementation, management, and improvement of systematic processes into Agile information system applications.

This paper presents an outline of the design and delivery of an entryway course in Enterprise Architecture and Infrastructure Systems using enterprise-class and open-source software tools as part of the Master of Information Systems. This course is an eight-week introduction to operating models, EA concepts, frameworks, and modeling language. The course is offered to non-traditional students in a virtual environment. The course design is based on the principles and methods used in non-traditional education to fit students' needs as students look for topics that have current relevance in the workplace

or personal life. Students are encouraged to utilize the concepts learned during class in virtual labs based on various cutting-edge enterprise application software trends in the IT industry. Key to the course design was to use enterprise software application tools in an Agile fashion to give students marketable skills that could be used immediately upon the conclusion of this course. Each enterprise tools highlighted in this course was mapped to a learning objective that will achieve the course goal. The main features for all the tools chosen were high industry ranking, ease of installation, accessibility, and affordability. Teaching the latest technologies will allow students to succeed in any IT job.This course methodology can be applied to other IT/CIS/IS courses.

2 Design of the Course

Strong, et al. [5], in previous work, emphasized the importance of using enterprise systems (ES) in teaching due to the high demand for experienced students. This paper narrates five different higher education institutions' implementation experiences, the journeys, challenges of integrating, and establishment of an ES curriculum that faculty embrace. Strong, et al. [5] based their research on vendor-specific ES such as ERP, SAP, which had a high cost and took a long time to deploy. In a most recent publication, Gamble [3] takes an analogous approach using enterprise integration and architecture tools in developing an EA course in higher education. Gamble uses open-source enterprise tools; though, some of these tools are no longer [3].

Enterprise Architecture is a set of processes and practices that shape how an enterprise's structure and behavior align with its overall business strategies [6]. Embracing Dx in an academic institution increases agility and innovation. A new pilot study takes a transformative teaching approach in higher education using various open-source enterprise software application tools in high demand in the IT industry [4]. Each tool is integrated into the EA curriculum every week using an agile approach in an eight-week session. Students' interaction and experience with these tools provide in-demand skills highly wanted by IT companies. This method also develops a holistic strategy and agility to provide marketable skills that can be added to a student's resume immediately after concluding this course.

The course was developed using backward design. The design strategy involved identifying the desired goals and planning the learning experience based on each goal. First, the overall course goal explored translating business vision and strategy into IS applications that support the business function. Enterprise Architecture acts as an essential role in the alignment between business and IT. Therefore, the course's scope includes people, processes, data, technologies, and relationships to the environment.

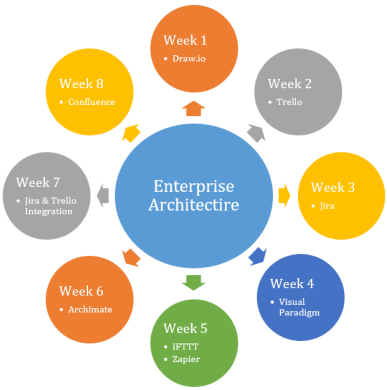

Figure 1: Enterprise Architecture Course Outline

Second, the instructor identified the relevant learning outcomes while developing the core competencies the students will gain in each module. As shown in Figure 2, a set of building blocks is needed to develop knowledge of EA in 8 weeks.

Figure 2: Enterprise Architecture Learning Experience

Third, this course was designed by mapping each week's learning outcomes with open-source enterprise tools that fit the module's needs and carefully choosing them based on industry ranking. Each enterprise software tool translated into a hands-on experience to reinforce class lectures [3]. Additionally, lab assessments enhanced students' conceptual understanding of the tools and theory relationships, level reasoning skills, and the development of workable experience in the lab. The course had a project-based summative evaluation to measure the progress through the class. Students received prompt feedback, reflecting on the objectives they accomplished, the tools they utilized, and how they performed.

Fourth, enterprise software tools were accompanied by traditional teaching materials such as books, case studies, and case scenarios. Top-ranked EA books were required, such as *Enterprise Architecture as Strategy* and *Enterprise Architecture at Work*. Current case studies were chosen from the IT research databases including IEEE and ACM. Students hone their evidence-

based decision-making in root cause analysis and role-playing scenarios. Every week, students developed their critical thinking by reflecting on real-world scenarios from Fortune 500 companies.

Finally, in the virtual classroom, an engaging online environment was created as a critical component to engage students to participate actively. Teaching with the best practices in technology-enabled software helped create an interactive multimedia experience with interactive presentations and lectures using visual aids.

3 Tool Identification

Enterprise architects use process modeling, analysis, and workflow automation in their day-to-day duties to offer different views for the stakeholders within an organization. Therefore, understanding modeling is an essential task, but what tools should be used to learn modeling? Using industry trends and real-world enterprise software tools are the most effective and evidence-based practices that allow students to gain expertise for in-demand skills and real-world scenarios [2]. Enterprise software application tools enable a business to reduce IT costs and minimize manual data input. These enterprise application software tools feature particular benefits, such as teamwork, quick response to the marketplace, increased work quality, and greater employee collaboration and efficiency. Similar, to the benefits in the IT industry, in academia, choosing enterprise systems is vital to students' hands-on experience to gain marketable skills [4]. These were key aspects that helped narrow down the myriad of options available selecting the enterprise tools:

- Easy access
- Affordability
- Open-Source
- Easy installation
- IT industry higher ranking
- In-Demand - Marketable Skills

The tools were chosen using Gartner Industry ratings. Gartner is the world's leading IT research and advisory company. IT industry leaders base many of their IT software or hardware decisions upon Gartner reports such as the Gartner Magic Quadrant and Gartner Peer Insight. The Gartner Magic Quadrant is a two-dimensional matrix that presents market trends formed from vendor evaluation. Gartner Peer Insights is an online platform portal for peer-driven ratings and IT solutions review [5].

Applying the same tactic to academia, the course uses the enterprise application software tools for instruction and critical-thinking learning listed in Table 1.

Table 1: Enterprise Software Applications Gartner's Ranking

Enterprise Software Application	**Tool**	**Gartner Peer Insights**	**Gartner Magic Quadrant**
Enterprise Agile Planning (EAP) Tools	• Jira	• 4.4/5	Leader
Enterprise Business Process Analysis	• Draw.io • Visual Paradigm	• 4.4/5	-
Integration Application Software as a Service	• IFTTT • Zapier	• 4.7/5 • 4.8/5	Leader
Enterprise Collaborative Work	• Trello	• 4.3/5	-
Enterprise Team Workspace	• Confluence	• 4.4/5	-

3.1 Enterprise Agile Planning (EAP) Tools

Enterprise Agile planning (EAP) tools help organizations use Agile practices to achieve business goals, collaborate, and perform enterprise-class Agile development. Academia's faculty can use Agile to create more personalized learning, teach students cross-functional roles, accelerate innovation, engagement, technology, and data shifts.

Among all the EAP tools, Jira (`https://www.atlassian.com/software/jira`) was chosen due to wide use in the IT industry. Jira is used to plan, track, document software changes, and release projects from beginning to end. Additionally, Jira allows integration with other platforms and several developer tools such as GitHub (`http://github.com`).

Jira offers a free version for its cloud-based product that does not require installation. Jira instances allow instructors to create traditional or next-gen projects using two types of templates: Kanban or Scrum. In a traditional project approach, a Kanban template is used to monitor teamwork from a backlog to Agile team members in a continuous flow during a sprint period. On the other hand, a next-gen project uses a Scrum template to organize cycles, manage progress, plan upcoming work. In an EA course, Jira was used to track projects, roadmaps, documentation, reports, and dashboards. Jira also allows rule automation, an excellent way to gain coding experience.

3.2 Enterprise Business Process Analysis (EBPA)

Gartner defines EBPA as "the discipline of business and process modeling aimed at transforming and improving business performance." [1] EPBA allows companies and leaders to make strategic and evidence-based operational decisions by examining cross- viewpoint, models, diagrams, mind-maps, and cross-function analysis.

Enterprise business process analysis tools are used in the course to set

a baseline for the basic concepts of EA, business, and the operating model. EBPA helps brainstorming and mind-mapping to identify critical components of EA: Core business processes, shared data, and customer information. Students learn about core diagrams or high-level view based on their company's operating model with that information.

Three similar cloud-based tools were available for use: Draw.io (`https://draw.io`), Lucid Chart (`https://www.lucidchart.com`), and Visual Paradigm (`https://www.visual-paradigm.com`). These tools are cloud-based, no installation needed, no cost. The idea was for the students to explore the different templates and modeling samples offered by these tools, which is especially important for students new to the world of modeling.

3.3 Integration Application Software as a Service

Integration Application Software as Service tools uses workflow automation to decrease repetitive processes, save time and money, increase efficiency, and reduce errors. Tools such as IFTTT (`https://www.ifttt.com`) and Zapier (`https://www.zapier.com`) allow specific integrations to automate processes or tasks, saving time and effort. IFTTT calls these integrations "applets," small applications with a specific purpose.

Applets can be created by combining different applications and adding triggers from within the applications. Applets have two types of actions: Do and IF, depending on the desired outcome. Actions run to gather data and push it to a new app based on the rules provided, allowing connectivity between devices and applications.

Integration Application Software in higher education can inspire students to seek out IS careers, learn more about automation, IoT, coding, and other CS/IS technologies. Integration Applications and EA go hand-in-hand because their purpose is to standardize and automate processes. Students with no coding experience are empowered to create, test, and deploy an applet within minutes. Zappier and IFTTT provide cloud-based, installation-free, and no-cost for a certain number of integrations.

3.4 Enterprise Collaborative Work Management Tool

Enterprise Collaborative Work Management tool is a collaboration space for individuals and teams to share project content securely and proactively. Its purpose is to increase productivity and collaboration. These tools' functionalities are content management for unstructured data, file sharing, content repositories, dashboard, and task management.

Fortune 500 companies promote knowledge-sharing using enterprise workspaces to store all their business knowledge and collaborate in training doc-

uments, technical documentation, and projects Trello (`http://trello.com`) is an enterprise collaborative work management tool that uses a Kanban board as a space for teams and projects.

Trello's board is composed of four elements: menu, boards, lists, and cards. Trello lists and cards are used to define tasks and keep teams organized to reach their goals. Trello can be integrated into an EA course to track students' weekly work or projects. Trello helps students learn how agile teams work in projects. Trello is a cloud-based collaboration tool with a free version that allows integration and automation.

3.5 Enterprise Team Knowledge-Based Workspace

Enterprise Team Knowledge-Based Workspace is an online space that allow teamwork, collaboration, and documentation. These digital spaces are like an intranet that serves as a document repository or wiki. Large organizations and Fortune 500 companies promote knowledge-sharing. They use enterprise knowledge-based workspace to store all their business knowledge and collaborate in training documents, technical documentation, projects, and more.

Confluence (`https://www.atlassian.com/software/confluence`) is a knowledge based digital workspace composed of pages and spaces. Spaces are how the content is classified. Spaces and organized by teams, projects, or individuals. Pages are hosted in the spaces created and organized in hierarchical order with a parent-child relationship.

Confluence in higher education can be used to create and compile the final project, an executive presentation, and a SWOT (strengths, weakness, opportunities, and threats) analysis that integrates the concepts, tools, and conclusions learned. Confluence is cloud-based, needs no installation, and provides free access for small teams.

3.6 Enterprise Architecture Methodologies

Enterprise Architecture Framework, language, and modeling tools were based on standards defined by the Open Group. This worldwide consortium supports open, vendor-neutral technology standards (Table 2).

Table 2: Enterprise Architecture

Enterprise Software Application	**Tool**	**Open Group**
Enterprise Architecture Framework	TOGAF	World's leading companies choice
Enterprise Architecture Modeling Language	ArchiMate	Supported by multiple vendors
Enterprise Architecture Modeling Tools	Archi	Open-Source Modelling tool

3.6.1 Enterprise Architecture Framework (EAF)

An Enterprise Architecture framework (EAF) establishes the guidelines, blueprint, and best practices to create and operate an EA and achieve the enterprise goals with aligned IT, business strategies, and objectives. Different frameworks and methodologies classify the architecture description into domains, layers, or views. These frameworks produce several artifacts that help top management make long-term decisions around new design requirements, sustainability, operations, and support.

TOGAF (`https://www.opengroup.org/togaf`) is the EAF used by the world's leading organizations; therefore, it is the choice for our EA course. The TOGAF Architectural Development Method (ADM) provides the steps needed to develop and manage an EA architecture's lifecycle. The TOGAF and ADM can be taught in higher education to cover planning, lifecycle, projects, capabilities, strategy, business, and EA concepts.

3.6.2 Enterprise Architecture Modeling Language

Modeling languages use diagrams, models, and annotations to document, visualize, and describe software systems and business processes. An EA modeling language is a language that represents the different views, layers, and aspects of EA. ArchiMate is the EA modeling language standardized by the Open Group that provides concepts and models to better fit EA needs.

ArchiMate (`https://www.opengroup.org/archimate-home`) is used for enterprise modeling across different layers. ArchiMate focuses on modeling viewpoints of the business, application, technology, and architectural layers. ArchiMate is composed of different notations and concepts that complement TOGAF.

Process modeling such as UML or BPMN could be used in academia to introduce ArchiMate. ArchiMate offers students and teachers a common language to discuss organizational structures, systems, and IT infrastructure. ArchiMate helps students to ensure consistency across a company's processes.

3.6.3 Enterprise Architecture Modeling Tools

Enterprise Architecture Modeling Tools are created by different vendors to support EA Modeling Language ArchiMate. The tool used in this course was an open-source modeling tool called Archi (`https://www.archimatetool.com/resources`). Installing Archi is easy and self-intuitive. The Open Group Library has ArchiMate case studies that can be imported into Archi as an XML file.

Archi allows enterprise architects with Canvas to create visual representations or views of organizations' current and target architectures. Archi view-

points can help stakeholders make better decisions to benefit the company. Archi can be used to teach modeling, decision-making, cross-functions, ArchiMate, and EA.

4 Mapping of Learning Outcomes and Enterprise Software Tools

This course's framework was implemented by mapping each of the module's learning outcomes with enterprise software tools to achieve the course goals of preparing students for industry needs. This framework can be modified to suit a similar course's needs. The course's design included consideration of the population of non-traditional students with and without an IT background.

The enterprise software tools chosen were used weekly with a baseline project, which involves a weekly company assignment, a real-world case scenario for discussion, critical thinking, analysis, research, and a hands-on activity [5]. Students performed labs with enterprise tools and created deliverables of the models, automated processes, or developed views to upload in the learning management system. Also, students reflected on the scenario or task given and wrote a reflection about the tool, the concepts learned, and the possible effects of the tool on their company. This is a foundational course that uses technology adoption, integrated into the curriculum, as shown in Figure 3.

5 Students Experience & Lessons learned

Some of the challenges presented were for students with no technical background or IT experience. This challenge requires preparing additional material with step-by-step instructions and complimentary demo videos of the tools used. Each tool is part of a hands-on activity developed to reinforce concepts learned and supplement the learning outcomes chosen for the module. Each hand-on activity has a deliverable, either a XML or PDF file, a written reflection. In the end, an assessment is presented to evaluate what students have learned from different tools.

The utilization of open-source software is a common practice in the IT world, but it has advantages and disadvantages. The advantages include no-cost, easy to install, and community-support. The disadvantages are no vendor support for existent issues. One recommendation for teaching with EA tools is to keep a backup tool for each module as availability may change. Students developed their critical thinking by reflecting on real-world problems in case scenarios, simulations, and role-playing exercises. Students use evidence-based decision-making in root cause analysis, resolving issues, and supporting their findings. Students utilize the concepts learned in class in virtual labs with

Week	Course Objective	Tool	Projects	Course Use
1	•Discuss the fundamental concepts and applications of EA.	• Lucid Chart • Draw.io • Visual Paradigm	• Mind mapping • Core diagram	• Explain the foundation for execution and EA • Describe operating models.
2	•Discuss the purpose of EA and its impact on business and IT.	• Trello	• Kanban board to organize sprint (weekly goals)	• Describe EA benefits • Identify the stages of business maturity
3	•Discuss the purpose of EA in outsourcing and the profitability of the operating models.	• Jira (Issue Tracking) • Roadmap	• Information Technology Infrastructure Library (ITIL)	• Support an organization in gaining optimal value by aligning IT services with business strategy.
4	•Identify and differentiate among several EAFs.	• Visual Paradigm • Lucid Chart	• BPMN process diagram • UML activity	• Draw company processes using the BPMN and UML
5	•Describe TOGAF framework definition, processes, tools, and extensions	• Trello • IFTTT or Zapier	• Automation of Trello cards with IFTTTor Zapier.	• Learn the TOGAF Architecture Development Methodology (ADM)
6	•Describe standard EAF and their applications.	• ArchiMate	• "Layered View" in ArchiMate	• Create a diagram of TOGAF Architectures Business, Application &Technology.
7	•Discuss the purpose of Collaborative, Agile EA	• Agile Jira automation	• Agile EA	• Automation using Agile tracking tool Jira
8	•Describe the Dx and EA.	• Confluence integration	• Digital Platform • Digital Roadmap	• Final Project • Executive Presentation

Figure 3: Mapping of the Course

the latest tools that mimic an enterprise and present real-world experiences' challenges. As an instructor, the experience was pleasant. The students were creative and resourceful. Despite the non-technical experience of students' in the first cohort, they excelled in modeling tools and business process analysis. Students were able to identify the essential parts and explain the foundation for execution and its effect on EA and operating models.

6 Future Plans and Conclusions

There are a plethora of technologies changing our lives. Companies are eager to embrace the Dx to innovate and transform to adapt to the current industry shift. Enterprise Architecture helps organizations to align business and IT objectives, goals, and infrastructure. Enterprise Architecture tools assist companies in planning the roadmaps needed for Dx. Higher education is also evolving; graduate students have new expectations, dreams, and hopes. Academia needs to shift its curriculum in educating students to be prepared for the workforce based on knowledge, competence, and mastery of skills. The importance of using enterprise tools in academia centers on the demand of students to gain marketable skills. Enterprise tools as a hands-on activity give students real-world experience and prepare them for organizational change and Dx. These tools offer the ability to collaborate, test, simulate to help students create and implement models to better business and IT processes, development, and architecture.

This course framework can be adapted to any technology course. Enterprise software application tools will continue changing, becoming obsolete, or emerging; it is advised to research these tools' availability. The assignments and integration between enterprise tools could be implemented using an application programming interface (API) reliant on the student-population coding background. Future work may include developing software to integrate EA tools, cloud computing, and Artificial Intelligence.

References

[1] Market guide for enterprise business process analysis, 2015. `https://www.gartner.com/doc/2998122`.

[2] Jae J Choi and Thomas L Ngo-Ye. Selecting enterprise applications for curriculum: Insights from a teaching initiative. *Issues in Information Systems*, 20(1), 2019.

[3] MT Gamble. Teaching enterprise integration and architecture-tools, patterns, and model problems. In *AMCIS*, 2011.

[4] Brad Smith. Microsoft launches initiative to help 25 million people worldwide acquire the digital skills needed in a COVID-19 economy, 2020. https://news.microsoft.com/skills/#skills-and-jobs-analysis.

[5] Diane Strong, Jane Fedorowicz, James Sager, Glenn Stewart, and Edward E Watson. Teaching with enterprise systems. *Communications of the Association for Information Systems*, 17(1):33, 2006.

[6] Elka Torpey. Occupations that have it all: Many openings, fast growth, and high wages, 2020. https://www.bls.gov/careeroutlook/2020/data-on-display/occupations-that-have-it-all.htm.

Teach Like a Git: Streamlining Assignment Administration and Enforcing Good Habits With Professional Tools and Software Development Practices*

Nathan W. Eloe
School of Computer Science and Information Systems
Northwest Missouri State University
Maryville, MO 64468
nathane@nwmissouri.edu

Abstract

Computer Science is a rapidly evolving field; the tools and methodologies that educators must introduce to novice programmers are both changing and growing. Teachers are often tasked with doing more with less, which leads to a difficult issue: how does one teach more content without more time or (often) more resources? This paper explores the integration of professional tools and practices at the pedagogical level in a Data Structures course and the effects it has on course and assignment administrative overhead (both from the instructor and the teaching assistants). Additionally, assignment expectations (with associated grade implications) that reinforce best practices and guide students toward solutions are presented.

1 Introduction

Computer Science education is constantly evolving, and in some ways becoming more complex. Educators teach more than programming; students must

learn to solve novel problems using a set of tools that grow increasingly powerful (and proportionally complex) over time. Students need to be familiar with tools such as Integrated Development Environments (IDEs), Debuggers, and Version Control. Ideally novice programmers would graduate from a program able to collaborate with others using professional tools and software development practices. To this end teachers must give students the ability to practice those skills in the context of a course that is not necessarily devoted to teaching these tools or the best practices while using them. While these tools might not be primary learning outcomes in a given course, giving students the opportunity to form good habits at an early stage in their education allows for more thorough mastery of these tools and methodologies later in the curriculum. This paper introduces some tools and practices students should be introduced to, and the steps taken to integrate them at the pedagogical level in a Freshman/Sophomore level Data Structures course while encouraging the formation of habitual best practices, and gives examples of how to expand the introduction of best practices in further coursework.

1.1 Version Control

Readers interested in the history of version control systems should refer to [13]. Version Control Systems (VCS) are powerful tools in a developer's arsenal that enable clean collaboration, regression testing and analysis, and organization of one's problem solving process. The VCS of choice for integrating into the assignment workflow is Git [5]. There are several reasons for choosing this particular VCS; this is a continuation of previous work to reduce the learning curve involved with such a powerful tool [7, 4, 6], and there is significant support, infrastructure, and resources available to educational institutions from companies like GitHub (through their GitHub Campus Program [8] and GitHub Classroom [10]).

1.1.1 Best Practices

Some common advice given to those using Git is to "commit early, commit often." While many variations of this mantra exist, the idea is that you commit small sets of related changes that accomplish a task. Git (and other VCS) works best when you interact with them frequently, having granular commits that give you the flexibility to determine where and how functionality may have changed or broken. Other best practices revolve around the content of commit messages, or branching, but the focus of this paper and the assignment structure is specifically around the idea of interacting frequently with your VCS.

1.2 Software Development Workflows

It is in no way feasible to introduce student to every Software Development Process; these are well-defined workflows by which developers design and build software. While no one workflow is universally used, understanding that standard processes exist and how to interact with them is beneficial to students. This paper will focus on Test Driven Development [3] or TDD, where unit tests are developed before the code is.

1.3 Continuous Integration

Continuous Integration [12](CI) has over time come to mean not just interacting with a centralized repository, but a configuration of automated testing whenever code is pushed to a main branch. Standalone CI servers such as Jenkins [1] can be configured and maintained, but GitHub has recently introduced the ability to automatically perform configured workflows through GitHub Actions [9]. This particular set of tools has the benefit of being well integrated with GitHub Classroom and its autograding system.

1.4 But Why Data Structures?

Data Structures is an interesting class; much of the content is common across learning institutions (common Abstract Data Types including lists and trees, complexity, and introduction to certain common algorithms). These ADTs lend themselves well to teaching students to code to an API; an `ArrayList` (or `vector`, or `dynamic array` depending on your programming language) will support the same operations from semester to semester. This alone lends itself to having students work with pre-written unit tests; the postconditions of each of the operations in these data structures is well defined. Additionally, many of the assignments are unlikely to change drastically over time; at some point in the class students will be asked to implement a Linked List. The biggest changes to many of the assignments comes from upgrades to tooling (build systems, language changes, etc) and tweaks to make the assignments more accessible to students (removing ambiguity, for example). By working on assignments distributed with unit tests, students are essentially interacting with Test Driven Development on a weekly basis without even knowing it; while they do not write tests until the end of the semester, they are still taking time to interact with a professional development workflow. This provides a level of transparency for students (they know when their code works, because the tests pass) and allows them to see regressions in real time assuming they run the tests as they develop their code.

Finally, at this point students have already been introduced to their primary problem solving tool: a programming language. While this author believes that

introducing some of these tools (specifically version control) could be possible in an introductory programming class, it would need to be done carefully to avoid overloading students who may have little to no experience with computers (much less programming). This is particularly a topic of interest that will be investigated in the future with faculty teaching such courses. Already some headway has been made in this area, with this toolset being used in the last project in the course preceding the Data Structures Course.

While mastery of the Git VCS is in no way a primary learning objective or outcome of the Data Structures course, students are still forming habits that will carry forward both in their education and career. Professional developers don't (or shouldn't, at least) email zip files of code or deploy production code by submitting an archive of code to a dropbox. These are the very habits students are forming when submitting code through an LMS, or in some cases observed by the author, in Word document submissions. While there is no line item on the Data Structures curriculum that says students will be familiar with Git or CI/CD, the students can be exposed to these tools in a way that makes it more familiar when they encounter them later in their education in classes where such tools and methodologies would be expected to be first-class learning outcomes.

2 Structuring An Assignment

Most programming assignments and labs in the Data Structures course follow the same structure and delivery mechanism. GitHub Classroom is used to distribute an assignment skeleton containing classes with method stubs (the course is currently taught in Java). Additionally, unit tests are distributed in the project. The assignment description and expectations are distributed as the `README.md` file with the repository. Students are expected to write code that passes the unit tests (essentially following the Test Driven Development paradigm) and commit and push to their own repository on GitHub. A common rubric for most labs is shown in Table 1

Table 1: Rubric for Most Labs in the Data Structures Course

Criteria	Total Points
Source file comment headers	1
Coding style (clean and consistent)	1
At least one unique commit per milestone	3
Correctness (tests)	5

Documentation is provided to students for using GitHub Desktop as a simple GUI git client to interact with GitHub; it exposes primarily the simple func-

tionality that is needed for the majority of development work (`add, commit, push, pull`). It also integrates cleanly with GitHub (allowing students to open and clone their repositories directly from the repository page itself).

When the student pushes their commits to GitHub, GitHub Actions is configured to automatically run the tests and report the result through the GitHub Classroom interface; this streamlines the process of grading for correctness for the TAs and Instructor. In this way, students are interacting with TDD, Unit Tests, Git, and Continuous Integration in almost every programming assignment they do in the class.

The students' results are displayed directly on their GitHub repository; both they and any course staff can see at a glance not only if the code is correct, but with a few clicks can see the results of the tests. Figure 1 shows the repository view when the student fails any of the tests associated with the project; more details can be obtained by clicking on the red X and selecting the details link. When students pass all of the configured tests, they see a view similar to Figure 2. This enables course staff to quickly offer advice and help without first needing to clone the students' assignment and open it in an IDE to run it.

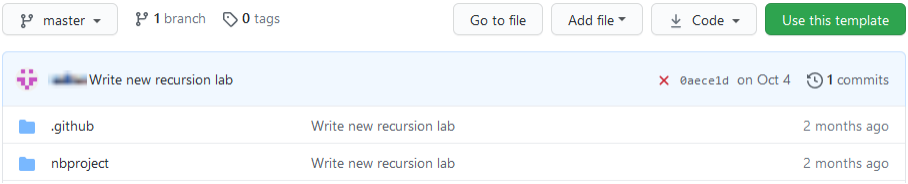

Figure 1: Repository View With Failing Tests

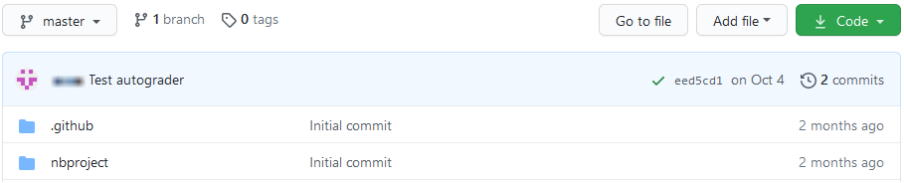

Figure 2: Repository View With All Tests Passing

2.1 Milestone Commits

The rubric mentions milestones, which is an addition to the assignments that is designed to both encourage more frequent interaction with git and give stu-

dents an idea of where to start their assignments and progress through in a logical way. The assignments are broken into milestones; these are specific points at the assignment where students are required to commit their work before proceeding with the next part of the assignment. Table 2 shows a set of milestones for a lab with only a few tests.

Table 2: Example Milestones for a Lab With Two Unit Tests

Description	**Tests passed**
Add required source headers to the non-test `.java` files	n/a
Implement the `heapifyUp()`, `add(Integer i)` methods	`testAdd`
Implement the `heapifyDown()`, `remove()` methods	`testRemove`

These milestones provide multiple benefits to the students. They require small, more frequent commits than a singular monolithic commit at the end of the assignment. More importantly they guide students through areas of the lab where there might be some dependencies. For example, in the lab where they implement a Linked List, there is really no test that can pass without the implementation of the `get(int index)` method or the `add(int index, Integer value)` method; by guiding them through those portions of the assignment first, the students are not put in a situation where they cannot pass the unit tests or verify their code is correct because of some missing prerequisite functionality.

Additionally, these milestones offer some level of protection against academic integrity violations in a class where many of the assignments can't change from semester to semester. While the projects can be rotated through in a way that helps mitigate academic dishonesty, lab assignments where students are implementing a specific data structure cannot really change from semester to semester. By enforcing the idea of milestone commits, even if the student has a fully working version of someone's code, they must work through the assignment step by step, understanding what is and isn't part of a particular milestone. In practice it has also helped identify academic dishonesty; a student who has many commits with code that does not solve the problem suddenly having a completely rewritten commit that passes all of the tests raises a red flag. In this case, it's relatively easy to compare the code to previous semesters or other students' work in the class to see if there is evidence of wrongdoing. While not a perfect solution to a problem that is not easily solvable, it does give more visibility into a student's work and progress.

The idea of milestone commits are the cornerstone of this course structure. When students use the milestones to guide their development of solutions, they are practicing the often repeated "commit early, commit often" wisdom of working with version control. Additionally, they are forced to stop and

decompose the problem into smaller more easily digestible portions. Problem decomposition is a vital skill in a field that revolves around problem solving using computers as a tool. While students at times are uncomfortable with the idea of the milestone commits, when the reasoning is explained they begin to see how important it can be, especially when there are larger projects.

3 Course Administration: TAs and Instructors

This approach offers several benefits to TAs and Instructors, all of which result in course staff being able to spend more time interacting with and helping students instead of creating, distributing, and grading assignments.

3.1 Assignment Distribution

Because many of these assignments don't change drastically from semester to semester, a repository can be maintained for each one. Distributing the assignment becomes the simple task of creating the assignment in GitHub Classroom and distributing the acceptance link to students on the course website. In situations where assignments are not reusable (or at least immediately), this may add additional overhead, though in the author's experience interacting with Git and GitHub Classroom is at least as fast (if not faster) than distributing the assignments through some other medium. This is in part due to the author's familiarity and experience with Git; not all instructors may be as comfortable with this tooling, and may find interacting with the LMS faster.

3.2 Grading

With the common rubric shown in Table 1, grading is quite fast. The correctness score can be determined from a glance at GitHub Classroom's interface, though even without that helpful UI it would be very quick to view the results of the CI server and see how many tests passed and failed (which was the approach taken before GitHub Actions and GitHub Classroom's autograding feature were released). The number of commits is quickly visible from the repository's main page, and the source code can directly be viewed in GitHub's interface. This means that grading programming projects doesn't require the grader (either TA or instructor) to download or clone every assignment, open it in an IDE, and run the code manually. When allocating hours to TAs, it is possible to drastically reduce the number of dedicated grading hours and allow them more help session hours, which increases the available time for direct interaction with students.

Using a centralized automated grading system has an additional benefit: it provides a single point of "truth" for what the target system is. Over the course

of a semester students' systems may become configured differently. Perhaps one student updated Java, or is running a different version of the IDE with an older version of Gradle. In an Operating Systems Class where a similar structure is used for the assignments, some students are using MacOS, while others decide to use different distributions of Linux. In cases like this the "it worked on my computer" argument can be avoided because the result of the testing is transparently available to the student soon after their code is pushed to the repository. In some cases in upper level classes this has led to some interesting lessons about how different systems and different versions of standard libraries can behave differently when certain behavior is undefined (which is a valuable lesson to learn).

3.3 Assisting Students

Using these tools, it is possible to configure them so only the TAs, Instructor, and the student have access to the student's repository as the repositories are not set up to be public. This means that when asking for help, a student can push their code directly to GitHub, and send a communication to course staff. Course staff can respond with links to specific lines of code or leave comments on the commit itself. This is a much more professional approach to requesting assistance (instead of emailing screenshots or zips of code). Again, this streamlines the process, something that has been greatly appreciated during the current pandemic, but is beneficial even in situations where remote/asynchronous communication is not the accepted norm.

3.4 Room for Growth

This approach has students use the basic elements of tools and workflows that they might see upon entering the professional world. It also leaves room for additional practice and topics to be added to their software development toolkits without drastically changing the core of the workflow. With little extra work on the part of the educator, students could be taught what makes a good commit message, and evaluated on how well they can follow some basic rules for well structured commit messages [2, 5]. Concepts such as branching in git, or a fully featured industry accepted workflow like GitHub Flow [11] can be added to the requirements. The core of the development model remains unchanged, only adding the introduction to branching and potentially pull requests.

After working with this system enough, students could be instructed to write their own unit tests, or configure GitHub Actions (or other CI solution of choice) to run their tests in situations where everyone may not be using the same tooling. Again, the additions are scaffolded onto the original workflow, and do not drastically change the assignment submission process.

4 Conclusion

This paper introduces a structured approach to assignments that immerses students in professional workflows and tools, but does so in a way that builds on previous work to flatten the learning curve of these powerful tools. By keeping a consistent approach to the assignments and reducing administrative overhead involved in distributing, collecting, and assessing assignments, course staff is able to spend more time interacting with and helping students directly, and the limited supply of TA hours can be allocated more efficiently.

This approach has other benefits; in later classes more best practices in interacting with version control or professional workflows can be added in through assignments and projects. The author uses this approach in both an Algorithms course and an upper level Operating Systems class and requires students to write descriptive concise commit messages. The practice they've gained from the milestone approach helps them in dividing their solution into manageable pieces and documenting them through the commit messages.

Additionally, this approach is not limited to the tools described in this paper. Before GitHub Classroom simplified the distribution of assignments, a custom tool using GitLab and a simplified Git Client [6] was used. Until the introduction of GitHub Actions, Travis CI was the Continuous Integration platform of choice (as late as the Spring semester of 2020). This approach works in any class where there is a consistent set of tooling; for example it is also used in the Operating Systems class, where assignments use the C programming language, and a different testing framework is used. It could be modified by requiring the students to write their own testing code and configure GitHub actions to run them automatically on every push, giving them some experience in configuring their own automated workflow and CI system. By allowing pedagogy to evolve with the available resources and tooling, students can gain skills and knowledge useful in the professional world. Using these tools in the assignment submission and evaluation process itself allows educators to expose students to these concepts while still keeping important core course content. Doing so while reducing course administration overhead allows teachers and teaching assistants to spend more time interacting with students rather than the course's Learning Management System.

References

[1] Jenkins. `https://www.jenkins.io`, 2020. Accessed: 2020-12-03.

[2] Chris Beams. How to write a git commit message. `https://chris.beams.io/posts/git-commit/`, 2014. Accessed: 2020-12-04.

[3] Kent Beck. *Test-driven development: by example.* Addison-Wesley Professional, 2003.

[4] Denise Case, Nathan Eloe, and Jennifer Leopold. Scaffolding version control into the computer science curriculum. In *Proceedings of the 22nd International Conference on Distributed Multimedia Systems (DMS 2016) and Workshop on Distance Education Technology(DET 2016)*, pages 175–183, Salerno, Italy, 10 2016.

[5] Scott Chacon. *Pro Git.* Apress, Berkely, CA, USA, 2nd edition, 2014.

[6] Nathan Eloe. Gitsubmit and VeCVL: Integrating Version Control in Introductory Computer Science Education. In *Proceedings of the 23rd International Conference on Distributed Multimedia Systems, Visual Languages, and Sentient Systems (DMSVLSS 2017)*, pages 1–7, Pittsburgih, PA, USA, 7 2017.

[7] Nathan Eloe, Denise Case, and Jennifer Leopold. VeCVL: A Visual Language for Version Control. In *Proceedings of the 22nd International Conference on Distributed Multimedia Systems (DMS 2016) and Workshop on Visual Languages and Computing (VLC 2016)*, pages 105–111, Salerno, Italy, 10 2016.

[8] GitHub, Inc. Engaged students are the result of using real-world tools - GitHub Education. `https://education.github.com/`, 2020. Accessed: 2020-12-03.

[9] GitHub, Inc. GitHub Actions Documentation - GitHub Docs. `https://docs.github.com/en/free-pro-team@latest/actions`, 2020. Accessed: 2020-12-03.

[10] GitHub, Inc. GitHub Classroom. `https://classroom.github.com`, 2020. Accessed: 2020-12-03.

[11] GitHub, Inc. Understanding the GitHub flow - GitHub Guides. `https://guides.github.com/introduction/flow/`, 2020. Accessed: 2020-12-04.

[12] Mathias Meyer. Continuous integration and its tools. *IEEE software*, 31(3):14–16, 2014.

[13] Nayan B Ruparelia. The History of Version Control. *ACM SIGSOFT Software Engineering Notes*, 35(1):5–9, 2010.

Open-Source Scholarship at Teaching-Oriented Institutions*

Joseph Kendall-Morwick
Department of Computer Science, Math, and Physics
Missouri Western State University
St. Joseph, MO 64507
josephkendallmorwick@missouriwestern.edu

Abstract

The shared values between free and open source software (FOSS) and those of the academic community have led to increasing adoption of FOSS projects by researchers and instructors in computing fields. Many papers have been written evaluating the pedagogical value of individual FOSS projects and FOSS as a whole. More recently the scholarly value of FOSS has also been explored. This paper bridges the conversations on the scholarly and pedagogical merits of FOSS development and identifies the value of FOSS to helping computing departments in small and regional colleges and universities better support the mission of their institution.

1 Introduction

Practical "real world" exercises in software development have long been an important part of immersing undergraduates in the inherent complexities of such projects that are difficult to relate in a reading or lecture format. Free and Open Source Software (FOSS) projects, in particular, have been gaining traction as a means for delivering such learning experiences, in part due to FOSS's overlapping principles with academic freedom [8]. As the success of free and open-source software (FOSS) continues to grow, academics in computing fields are taking an increasing interest in evaluating the merit of such work in a scholarly context [10]. This takes on a greater importance for faculty at small

and regional colleges and universities (hereafter referred to as "small colleges") for whom, as Reder suggests, "research is not only a source of inspiration for faculty members' teaching" but "has also become their teaching" [9]. FOSS projects are both an important scholarly activity and pedagogical tool for faculty of such institutions as such projects enable faculty to involve students in longer term projects that can extend beyond the constraints of a single course.

The significance of these activities at small colleges needs to be considered in the context of their institutional missions. In the book "Scholarship Reconsidered: Priorities of the Professoriate", Boyer states "we need a climate in which colleges and universities are less imitative, taking pride in their uniqueness. It's time to end the suffocating practice in which colleges and universities measure themselves far too frequently by external status rather than by values determined by their own distinctive mission" [3]. This perspective carries important implications for small colleges, in part due to their tendency towards a distinctive local culture [9], but particularly when scholarship is the second or third priority to teaching and service to the community. Boyer further states that education will be improved "in large measure, by the way scholarship is defined and, ultimately, rewarded". A nuanced perspective on the nature and importance of scholarship at small colleges, and in particular one that recognizes FOSS development, will help computing departments at these institutions better meet their mission on all three fronts simultaneously.

2 FOSS as a Scholarship Priority at Small Colleges

2.1 Definitions and Goals of Scholarship

Definitions of scholarly activity can vary from discipline to discipline, but a common thread in such definitions is peer review and public availability. Originality is also emphasized, and in the sciences especially, reproducibility is quite important. Artifacts of scholarly activities can also vary, but again there is a common recognition for published journal articles and books that meet the above criteria.

Boyer identifies multiple categories of scholarship: discovery, integration, application, and teaching, and identifies "discovery" as the primary category for traditional research over much of the latter half of the last century [3]. This category makes the traditional peer reviewed article an ideal means of documenting and communicating such advances. However, FOSS presents an opportunity and even a motivation to move further in to the other areas of scholarship [10].

2.2 Justification for FOSS as Scholarship

FOSS development and academic publications have their differences, but they also share a lot in common, especially in the ways that academic publications meet the standards for scholarly work. Like a paper, software is and should be recognized as a transfer of useful knowledge. "Free and Open-Source" implies a standard for public availability beyond what many academic publishers provide. The reproducibility of source-code is practically absolute, well exceeding the capacity of a paper to describe an experiment to a reproducible extent. While originality is not a cornerstone of FOSS development, proper attribution of sources is, particularly through the use of open-source software licenses and the ability to track the development of a project through version control.

There are also challenges to viewing FOSS as scholarship. For example, software may be seen as less permanent, given shifting needs of users, definitions of programming languages, and availability and protocols for interfaces. This may affect how the impact of a FOSS contribution is considered.

Another area where it's difficult to draw a clear analogy between publications and FOSS development is peer-review. Although processes for such review exist, they can be difficult to formalize. It isn't always clear what a peer is in this context, and contributions (in terms of acceptance in to a project) are often judged by the developers maintaining the project who cannot be viewed as independent. It also isn't always clear what the significance or impact of a FOSS contribution is. Quantifying contributions by "lines-of-code" is challenged by the fact that each line of code can have wildly varying significance to the project, and it's also not clear what impact a programmer had on a project if their code is later modified or overwritten.

Beyond credit within a project, it's also unclear how to judge a project as a whole and award credit proportionally. Software can be given a DOI and cited directly, but it remains unclear if this improves attribution of credit to FOSS developers [16]. Adoption (by users) or incorporation (by other developers) could be considered somewhat equivalent to citation, but measuring this is difficult. Metrics involving the number of downloads or social media endorsements, such as the "star" option on github, could be considered, but aren't especially meaningful and can be manipulated.

To be fair, these challenges in evaluating the impact of artifacts aren't unique to FOSS. Peer-reviewed publication is also vulnerable to abuses and can similarly result in work of low importance, as outlined by Denning [6]. These artifacts, like software, can also have a limited life time and see their influence diminish over time. The judgment of tenure and promotion committees is still relied on heavily in these cases, and perhaps should also be the answer to judging the merits of FOSS contributions.

Existing literature provides viable pathways to implementing usable stan-

dards for evaluation of FOSS as scholarship. Ambati and Kishore explore a process of peer review involving FOSS savvy scholars [1]. Hafer and KirkPatrick present best practices for both researchers and committees in documenting and evaluating FOSS contributions [10]. However, these recommendations are mostly applicable to judging projects as a whole rather than individual contributions.

Separately judging the value of individual contributions to the larger project is also important, and existing software development infrastructure facilitates such review. For example, github "pull requests" are a useful means of documenting a single contribution large enough to satisfy some coherent requirement of the overall project. Howison and Herbsleb argue that rewarding these smaller and more dynamic contributions would encourage greater collaboration [12]. Although some such contributions might be purely judged as service work, others could be considered scholarly with fractional credit from the overall project [10]. Katz outlines a process for "transitive credit" particularly useful for library code integrated in to multiple larger projects [14]. Encouraging greater collaboration would benefit faculty at small colleges by providing incentive for integrating undergraduate projects and forming partnerships with larger institutions and organizations.

2.3 Impact of FOSS on Communities

The needs of communities served by small colleges are unique and not always easily solved with a one-size-fits all software solution. This makes FOSS particularly attractive as the source code is available and can be modified to meet specific needs. However, this requires resources: money, hardware, and talent. Non-profit organizations such as Code for America help connect those willing to provide their time and talent, pro bono, to municipalities that can benefit from this work. Many academic institutions already partner with such organizations or bring their communities the same kind of benefits directly.

Academic institutions are also poised to offer more than development efforts for software projects. In the broader allied computing disciplines beyond Computer Science and Software Engineering, programs in Information Technology, Information Systems, and Cybersecurity can similarly involve students and faculty in work to deploy, monitor, and maintain software applications [15]. As computing resources become less expensive and more readily available for the purposes of custom deployments, these disciplines also have an important role in FOSS projects for local / civic benefit.

3 Teaching with FOSS

Software development topics such as requirements analysis, software testing, and many others, are best approached within an applied context, such as a significant software project. FOSS development provides opportunities for projects with a real impact while maintaining suitability for a non-profit organization (in this case, academic institutions) [18]. FOSS can also play a role in recruitment and retention [4].

3.1 Real Projects for Real Clients

The pedagogical value of the applied nature of software projects is enhanced when work is performed for real clients whose needs are not being simulated for the purpose of an exercise [4]. These courses often take shape as a senior-level capstone course where students are expected to leverage the skills they have gained from prior courses in a more professional environment. The "real" clients, in these cases, can (and often are) the same benefactors of FOSS work mentioned in section 2.3 and are best situated within the same FOSS projects involving faculty, allowing faculty to further leverage their research experience in the classroom. Such project courses have become popular and continue to gain significant attention within the CS education research community [2].

3.2 Pedagogical Advantages of FOSS at Small Colleges

Many small colleges bring their communities the kind of benefits discussed in section 2.3 through RPRC courses. Missouri Western State University is one such example, having sponsored projects with government clients in Buchanan County. Cooperation between such institutions through FOSS can multiply the benefit, such as the humanitarian LibreFoodPantry collaboration between Worcester State University, Western New-England University, and Nassau Community College [17]. When state and local governments are balancing budgets and weighing investments in regional higher education vs government IT infrastructure, such partnerships are a way to achieve both simultaneously.

FOSS also provides a means for students to get involved with larger projects at a much earlier stage than a RPRC course. Publicly available FOSS code can be modified in small ways, targeting specific learning outcomes, where students can observe the impact of their work on the greater system. Introductory students can be directed to make small changes to a large software project, for example modifying the logic controlling the behavior of an NPC in a game, and gain an appreciation for what they're learning that can't be provided by isolated and impractical "toy" assignments.

Beyond government clients and other non-profits, the academic institution itself often can be, and should be, a client for FOSS work. Much of the same rationale applies for the institution to be its own client, but pedagogical advantages have been noted to this specific kind of FOSS work as students involved both as clients and contributors to a project can have a greater appreciation and understanding of the work [5, 11].

The focus of a FOSS project sponsored by a small college need not strictly be local, even if the institution's resources aren't sufficient for tackling large-scale global projects (such as the Sakai LMS [7]). Many FOSS projects provide opportunities for projects within projects that may be more amenable to long-term management by one or two faculty members, such as the IPAL plugin for Moodle [13].

Beyond pedagogical goals, FOSS helps academic institutions to help their students reach greater success earlier in their careers after graduation. Traditionally, software developers seeking employment would burnish their credentials through their employment history and recommendations from supervisors and co-workers. However, employers are increasingly judging applicants' potential through analysis of their contributions to open source projects. This raises the importance of student-faculty FOSS collaborations both as applied learning activities and as opportunities for portfolio and reputation building for the student.

4 Conclusions and Future Work

FOSS development itself is worthy of being considered scholarship, but producing metrics to measure the impact of that work remains a challenge [16]. The scholarly value of FOSS development is already recognized by the fact that many CS publications are centered around the activity of software development and principally serve to legitimize the FOSS contribution in the eyes of tenure and promotion committees [10]. Overall, more meta-level research is needed to drive the academic community closer to a consensus on standards, especially in regard to smaller contributions, but existing procedures are mature enough for FOSS development to play a much greater role in tenure and promotion decisions and consequently also undergraduate curriculum right now. Because this reporting has been primarily from the perspective of large research institutions, small colleges should collaborate, perhaps through workshops, in tailoring consideration of FOSS scholarship to their own distinct missions.

For small colleges, the integration of scholarship with higher priority goals of teaching and service can improve all three. With more formal recognition of FOSS contributions, faculty at such colleges could spend more time on expanding and exploiting their FOSS work for teaching and service purposes rather

than pursuing traditional publications to justify their FOSS work. Collection and curation of adopted best practices for consideration of FOSS contributions is warranted, but more so reporting is needed on the outcomes for students and community stakeholders leading from work inspired by updated definitions of scholarship. This will help put best practices in to context and assist faculty at small colleges to better meet their institutional missions through FOSS development.

References

[1] Vamshi Ambati and S P Kishore. How can academic software research and open source software development help each other? *Institution of Engineering and Technology Conference Proceedings*, pages 5–8(3), January 2004.

[2] Jürgen Börstler and Thomas B. Hilburn. Team projects in computing education. *ACM Transactions on Computing Education*, 16(2), March 2016.

[3] Ernest L. Boyer. *Scholarship reconsidered : priorities of the professoriate.* Carnegie Foundation for the Advancement of Teaching Princeton, N.J, 1990.

[4] Vincent A. Cicirello. Experiences with a real projects for real clients course on software engineering at a liberal arts institution. *Journal of Computing Sciences in Colleges*, 28(6):50–56, June 2013.

[5] Vincent A. Cicirello. Student developed computer science educational tools as software engineering course projects. *Journal of Computing Sciences in Colleges*, 32(3):55–61, January 2017.

[6] Peter J. Denning. A new social contract for research. *Communications of the ACM*, 40(2):132–134, February 1997.

[7] James Farmer and Ian Dolphin. Sakai: elearning and more. *EUNIS 2005-Leadership and Strategy in a Cyber-Infrastructure World*, 2005.

[8] The Free Software Foundation. What is free software? `https://www.gnu.org/philosophy/free-sw.html`. Accessed 2020-11-08.

[9] K.H. Gillespie, L.R. Hilsen, and E.C. Wadsworth. *A Guide to Faculty Development: Practical Advice, Examples, and Resources.* JB - Anker. Wiley, 2002.

[10] Lou Hafer and Arthur E. Kirkpatrick. Assessing open source software as a scholarly contribution. *Communications of the ACM*, 52(12):126–129, December 2009.

[11] Shalin Hai-Jew. Building open-source resources for online learning in a higher education environment. *Open-Source Technologies for Maximizing the Creation, Deployment, and Use of Digital Resources and Information*, pages 115–135, 01 2012.

[12] James Howison and James Herbsleb. Incentives and integration in scientific software production. *Proceedings of the ACM Conference on Computer Supported Cooperative Work*, pages 459–470, 02 2013.

[13] William Junkin. Ipal: In-class polling for all learners. `https://www.compadre.org/iPAL/webdocs/About.cfm`. Accessed 2020-11-08.

[14] D.S. Katz. Transitive credit as a means to address social and technological concerns stemming from citation and attribution of digital products. *Journal of Open Research Software*, 2(1), 2014.

[15] David Klappholz, Vicki Almstrum, Ken Modesit, Cherr Owen, Allen Johnson, and Steven Condly. A framework for success in real projects for real clients courses. *Software Engineering: Effective Teaching and Learning Approaches and Practices*, pages 157–190, 01 2008.

[16] Matthew S. Mayernik, David L. Hart, Keith E. Maull, and Nicholas M. Weber. Assessing and tracing the outcomes and impact of research infrastructures. *Journal of the Association for Information Science and Technology*, 68(6):1341–1359, 2017.

[17] Karl R. Wurst, Stoney Jackson, Heidi J. C. Ellis, Darci Burdge, and Lori Postner. Want your students to participate in open source? join us in librefoodpantry! *Journal of Computing Sciences in Colleges*, 35(8):252–253, April 2020.

[18] Kwok-Bun Yue, Zahabia Damania, Raunaq Nilekani, and Krishani Abeysekera. The use of free and open source software in real-world capstone projects. *Journal of Computing Sciences in Colleges*, 26:85–92, 04 2011.

Fourth Hour: A CS1 Review Session Led by Teaching Assistants Using Peer Instruction*

Megan Gilbert, Dee A.B. Weikle, Chris Mayfield, Chris Johnson
Department of Computer Science
James Madison University
Harrisonburg, VA 22807
gilberme@dukes.jmu.edu
{weikleda,mayfiecs,johns8cr}@jmu.edu

Abstract

As enrollment in computer science courses increases, efforts to retain students, especially those in underrepresented groups, have become increasingly important. In fall 2019, our department piloted a new intervention to support students in the event they fall behind in CS1. This intervention, which we called Fourth Hour, was a series of weekly review sessions led by undergraduate teaching assistants. Each session featured peer instruction questions specifically designed to address student misconceptions reported in the literature. To evaluate the impact of Fourth Hour, we invited students from all ten sections of our CS1 course (∼30 students each) to participate in a research study. Students completed pre and post assessments to measure learning gains and took three surveys to measure changes in attitude. Attendees reported that Fourth Hour had a positive impact on their learning and sense of belonging. Our results suggest that Fourth Hour improved outcomes for first-year students who scored below the mean on the first exam. We report lessons learned and offer suggestions for schools looking to implement similar interventions.

1 Introduction

Our department strives to teach well and create a positive learning culture. We use various forms of active learning in the majority of our courses, which are limited to a class size of 30 and taught by faculty. We support a wide range of activities to promote a sense of community including undergraduate TAs that provide help in courses and evening lab hours.

In spite of these efforts, we still lose a substantial portion of our first-year students. CS1 is notorious for students falling behind and having a difficult time getting caught up, in part because of the way it builds from one week to the next. To address this issue, we developed a new program called "Fourth Hour". Our main goal was to increase sense of belonging and learning gains by giving students a second chance to learn material. We piloted the program in Fall 2019, and it has continued during the Spring 2020 and Fall 2020 semesters.

We gathered data to analyze whether Fourth Hour could improve belonging and learning for regular attendees. All 282 students registered in the Fall 2019 course were invited to participate. Students completed a pre and post semester assessment and three attitudinal surveys. Those who never attended and gave consent were used as a control group. The pre-assessment was given during the first week of class, and the post-assessment in week 10. Results from pre and post assessments serve as the measure of material retention. Attitudes were measured three times over the course of the semester during week 2, week 8, and week 14.

Overall, we did not see major differences when comparing all students who attended Fourth Hour with all those who did not. However, we did see a difference in the target group for the intervention: first-year students who scored below the mean on the first exam. Section 4 summarizes these results along with statistical analyses that lead to specific guidance for future studies. In the next two sections, we will present the design and implementation of Fourth Hour itself and the accompanying materials.

2 Background: Choosing a Teaching Pedagogy

When designing Fourth Hour, we considered several pedagogical approaches including Peer Led Team Learning (PLTL), Process Oriented Guided Inquiry Learning (POGIL), Pair Programming, and Peer Instruction (PI). Only active learning methodologies were considered, because there is strong empirical evidence that active learning produces better learning outcomes [1].

PLTL was the initial inspiration for the Fourth Hour because of its success in the literature. Rydr and Hari show how PLTL can be used to help students who have little to no programming experience or are in minority groups

[6]. The class was offered as an additional 1-credit optional course of content characterized as "supplemental learning, not remedial". Groups are led by a peer leader, which could be a TA, to complete projects over a 2-hour workshop time. PLTL benefits include "significant gains in performance, retention, perseverance, and student attitudes and opinions" in STEM areas [2]. However, PLTL requires significant resources: one trained TA per group of eight, one classroom per group, and at least two hours of workshop time each week.

Many of our faculty teach with POGIL, so that was a natural option to consider for Fourth Hour. POGIL has students complete specially designed worksheets in teams of 3-4 where each student has a specific role. Faculty serve as facilitators to answer questions, direct whole class discussions, give real-time feedback, and keep groups on task. Although POGIL has positive material retention results and increases sense-of-belonging [4], facilitators need substantial training to be most effective [2]. It's also more appropriate for a traditional classroom setting in which attendance is more consistent.

Pair Programming is a technique that pairs students to collaborate on a programming assignment or task. One student writes code while the other observes, and the second half of the time the roles switch [9]. Pair-Programming has shown positive overall results in increasing retention and performance [5]. However, the goal with the Fourth Hour was to help students correct misconceptions as individuals, rather than practice programming.

Ultimately, we decided to adopt Peer Instruction for the review sessions. Data shows that even for first-time instructors, the results of using PI are significant in helping students retain material [10]. PI requires fewer resources and less training, which is helpful given that TAs tend to change every semester. It also increases material retention by having students work together to answer specially designed questions. Students are asked a question individually via an online quiz tool. Then, they break into groups to discuss answers and decide on a group answer. When students work individually to answer another similar question, individual scores increase for the second question.

3 Implementation

Each Fourth Hour session was designed to review CS1 concepts taught the previous week. The same session was repeated Monday, Tuesday, and Wednesday evening to accommodate schedules. Slideshows and worksheets were created by a Lead TA with faculty support, and all three sessions used the same materials. Each session included introductions, approximately three PI questions, a review of vocabulary, and short cooperative activities for groups of 2-3. Two TAs worked at each session: one as the presenter and the other as an observer. The Lead TA presented at the first session, and then the observer for

the first session presented at the second, and the second observer presented at the third session, where the Lead TA observed. This provided consistency among the sessions and feedback for those presenting. The presenters ran the slide show with the PI questions, answered student questions, and worked out problems on the board. The observer took attendance, checked the accuracy of the presenter, and supported the presenter with student questions.

3.1 Implementing Peer Instruction

We chose the free online tool Socrative to implement PI and increase engagement. Wash found that students reported strong agreement that using such an online tool increased their class participation, helped them receive feedback on their performance, and aided in creating positive interactions with their peers and/or professor [8]. Socrative is straightforward to use; a classroom code is used to sign in as opposed to creating a personal account, and it is platform friendly, usable on mobile devices and laptops [7]. PI questions were set up in Socarative as anonymous and teacher-paced. Each such question was asked twice using traditional PI pedagogy, after which the presenter would go over both the correct answer and why the other answers were incorrect.

3.2 Creating Peer Instruction Questions

PI questions were specifically designed to address misconceptions in the literature. Kaczmarczyk et al. discovered four themes in CS1 misconceptions [3]:

- T1: Misunderstanding the relationship between language elements and underlying memory usage.
- T2: Misunderstanding the process of while loop operation.
- T3: Lacking a basic understanding of the Object concept.
- T4: Cannot trace code linearly.

The authors developed a further breakdown of T1 (regarding memory usage):

- M1: Semantics to semantics
- M2: All Objects same size
- M3: Instantiated no memory allocation
- M4: Uninstantiated memory allocation
- M5: Off by 1 array construction
- P1: Primitive no default
- P2: Primitives don't have memory

These misconceptions are self-explanatory, with a couple of exceptions. We will rename "M1: Semantics to semantics", which refers to misunderstanding variable declaration by applying real world semantics to the process, as "M1:

Table 1: CS1 Misconceptions Covered in PI Questions

	T1	T2	T3	T4	M1	M2	M3	M4	M5	P1	P2
S1	X										X
S2	X				X						
S3	X		X				X			X	X
S4	X			X	X						X
S5					X						
S6	X			X							X
S7	X		X	X	X		X	X			X
S8	X		X	X	X	X	X		X		X
S9	X		X	X	X	X	X		X		X
S10	X		X	X	X	X	X		X		X
S11	X		X	X	X	X	X				X
S12		X			X						
S13	X		X	X	X	X	X		X		X

Semantics to variable declarations". And we will rename "P1: Primitive no default", which refers to misunderstanding that instance variables of a class have no default value, as "P1: Instance variable no default". Table 1 describes the misconceptions covered in each session of the Fourth Hour.

3.3 Fourth Hour Curriculum Materials

The complete schedule and all instructional materials are publicly available here: `https://github.com/mGilbert15/The-Fourth-Hour`. This repository includes all presentation slides and supplementary materials, such as worksheets and sample code. The PI questions described in the previous section are embedded in the presentation slides.

4 Results

Despite using motivation theory to market and promote Fourth Hour, attendance was only significant the week before exams. Thus, a limiting factor when looking at results of this study is the low number of participants. Furthermore, learning gains and sense of belonging were greater for those students who succeeded in the course without attending Fourth Hour.

Subsequently, we focused on identifying those from the control group who were more like Fourth Hour attendees in characteristics, such as background and early course performance. For sense of belonging, we focused on underrepresented students, which were one of the initial targets of this intervention.

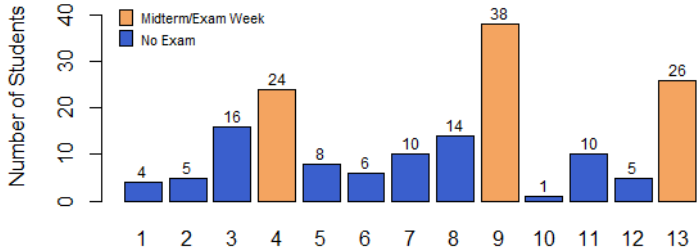

Figure 1: Fourth Hour Attendance Total per Week

4.1 Participant Attendance Trends

282 students registered for our CS1 Fall 2019, of which 262 participated in the study. 71 unique students attended Fourth Hour at least once. 69 out of the 71 students that attended Fourth Hour gave consent to participate.

Fourth Hour attendance was collected every week and analyzed at the end of the semester. Results shown in Figure 1 depict a trend of increased attendance on exam weeks and immediate drop the week after. Note in weeks 4 and 9, the Fourth Hour and exams were in the same week, while week 13 was the Fourth Hour for the final exam in week 14. In the sections that follow, No Frequency refers to students who attended Fourth Hour 0 times, Low Frequency is 1–3 times, and High Frequency is 4+ times. 193 students fall into No Frequency category, 58 in Low Frequency, and 11 in High Frequency.

4.2 Evaluating Content Retention

To evaluate content retention over the course of a semester, we analyzed the pre- and post-Core Assessment (CA) scores. Normalized learning gain was calculated to quantify learning over the course of the semester using:

$$(post - pre)/(100 - pre)$$

Students with a low normalized learning gain score had less learning gain than students with higher scores in general. While there are theoretical situations where students could score extremely high on the pre-CA and even higher on the post-CA that result in misleading high learning gain scores, this was rare in the actual data. Several students had a negative learning gain score, indicating that they scored higher on the pre-CA than the post-CA. Negative scores could be attributed to good guessing on the pre-CA, which was multiple choice as opposed to the post-CA which was short answer. For all of the questions, data was omitted for those that didn't complete either the pre or post-CA.

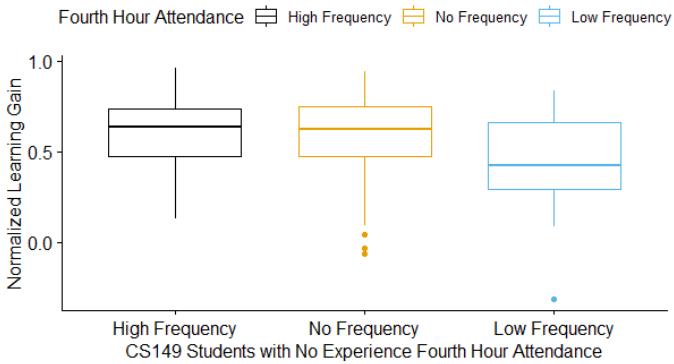

Figure 2: CS149 Students with No Prior Coding Experience

The initial question was to determine if there was a significant difference in learning retention between Fourth Hour attendees and the control group that did not attend. A Shapiro-Wilk test determined the data was not normal, so a Wilcoxon Rank Sum test was used to determine that the difference was statistically significant (W=2761, p=0.0004443). It was not a surprise that overall students who did not attend Fourth Hour had higher material retention than those who did attend. Fourth Hour attendees were likely to be struggling students seeking help, while those that never attended were likely to be making good progress.

Subsequently, we decided to compare smaller populations of students that had more similar characteristics (to see what difference, if any, Fourth Hour made). We compared students who had little to no prior programming experience, as well as those who had low performance on the first midterm. The next two subsections describe the results.

4.2.1 Evaluating Students with no Previous Coding Experience

Was there a significant difference in learning retention between students who had no previous coding experience who attended Fourth Hour? What impact did High Frequency (HF) attendance vs. Low Frequency (LF) attendance have?

The data was determined normal. The HF group had p=0.8981 and LF group had p=0.206 based on a Shapiro-Wilk test with a significance level of 0.05. An unpaired 2-tailed T-Test (0.05 significance-level) yielded no statistically significant difference between the two groups (p=0.2907), although the mean values appear to differ. (HF 0.592, LF 0.449, No Frequency 0.568). See Figure 2. The sample size was insufficient. (LF n=28, HF n=6).

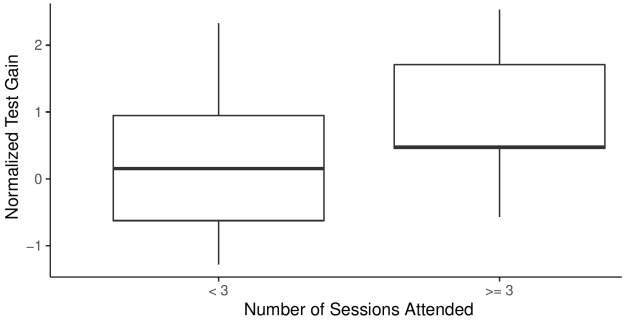

Figure 3: Normalized Gain Between Tests for First-year Students

4.2.2 Evaluating First Year Students with Low First Test Scores

If we consider only first-year students who performed below the mean on the first coding exam (N=31), we find that those who regularly attended Fourth Hour outperformed those who did not, as shown in Figure 3. The Wilcoxon Rank Sum test suggests that these two populations have differing medians, but the results again are not significant (p=0.23).

4.3 Evaluating Sense of Belonging

Students who participated in the study were asked to take three surveys over the course of the semester. The most interesting result comes from five items on the last survey. The items were preceded by "In this computer science class. . . "

1. I feel that I belong to the computer science community.
2. I feel accepted.
3. I feel like an outsider.
4. I try to say as little as possible.
5. I trust my instructors to be committed to helping me learn.

Questions were presented in Likert Scale format. Positive items (1,2,5) were given the numerical values: strongly disagree: 1, disagree: 2, neutral: 3, agree: 4, strongly agree: 5. Negative items (3,4) numerical value scales were reversed. Sense of belonging was calculated using the mean score. Scores closer to 1.0 correspond to lower sense of belonging.

Perhaps not surprising, self-belonging scores for those who attended Fourth Hour were considerably lower than self-belonging scores for those who never attended Fourth Hour. This is at least in part because students seeking outside help usually do so after poor results that impact belonging. The box plot in Figure 4 depicts sense of belonging at the end of the semester using mean scores

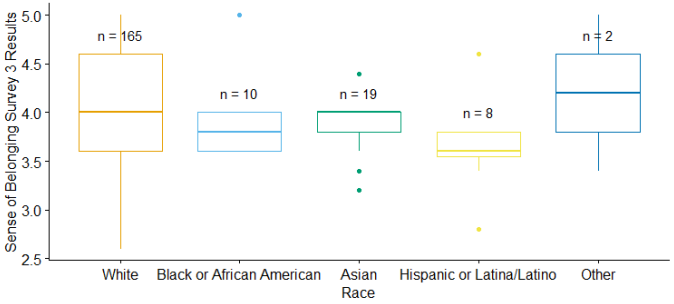

Figure 4: Sense of Belonging in CS149 Students

based on racial demographics. The trend of lower self-belonging for Fourth Hour attendees continued, except for Asian attendees. While these results are not statistically significant, we report them to document our methodology.

5 Conclusions and Future Work

This study developed materials for a series of CS1 review sessions called Fourth Hour. We designed dozens of Peer Instruction questions based on misconceptions in the literature. The primary contributions here are the structure of the intervention along with the materials and PI questions. In addition, our initial analyses show the potential of the intervention and direction of future work. Statistical analysis on assessments and attitudinal surveys suggest that struggling students with lower sense of belonging were the ones who sought help. Interactions with Fourth Hour attendees indicated participants felt better after attending, and attendance improved in the middle of the semester. Therefore, we believe the intervention has potential to improve both learning gains and sense of belonging, especially for underrepresented students.

To fully assess the intervention, future work will consider weekly assessments and/or extra credit to motivate increased attendance, pre and post assessments for each session, different measures of learning gains, and mixed-methods analysis with attending students. TAs as well as attendees should be surveyed, because TAs anecdotally indicated they too benefited from the experience. It is encouraging that a high percentage of underrepresented students and those with less programming experience do seek outside help. Though they had lower sense of belonging than the control group, that sense of belonging was not so low that they declined to participate. Future efforts will emphasize outreach to these students.

References

[1] Louis Deslauriers, Logan S. McCarty, Kelly Miller, Kristina Callaghan, and Greg Kestin. Measuring actual learning versus feeling of learning in response to being actively engaged in the classroom. *Proceedings of the National Academy of Sciences*, 116(39):19251–19257, 2019.

[2] Thomas Eberlein, Jack Kampmeier, Vicky Minderhout, Richard Moog, Terry Platt, Pratibha Varma-Nelson, and Harold White. Pedagogies of engagement in science: A comparison of PBL, POGIL, and PLTL. *Biochemistry and Molecular Biology Education*, 36(4):262–273, 07 2008.

[3] Lisa Kaczmarczyk, Elizabeth Petrick, J. East, and Geoffrey Herman. Identifying student misconceptions of programming. In *SIGCSE'10: Proceedings of the 41st ACM Technical Symposium on Computer Science Education*, pages 107–111, 03 2010.

[4] Richard Moog, F.J. Creegan, D.M. Hanson, J.N. Spencer, and A.R. Straumanis. Process-Oriented Guided Inquiry Learning: POGIL and the POGIL Project. *Metropolitan Universities Journal*, 17(4):41–52, 01 2006.

[5] Leo Porter, Mark Guzdial, Charlie McDowell, and Beth Simon. Success in Introductory Programming: What Works? *Commun. ACM*, 56(8):34–36, August 2013.

[6] B.G. Ryder and P. Hari. Peer-Led Team Learning Computer Science: Module 1: Introduction to Computer Science, 2012. Retrived from http://www.pltlis.org. Originally published in Progressions: The Peer-Led Team Learning Project Newsletter, Volume 9, Number 1, Fall 2007.

[7] Socrative, 2020. "Home", Showbie Inc, https://www.socrative.com/.

[8] Pamela Wash. Taking advantage of mobile devices: Using Socrative in the classroom. *Journal of Teaching and Learning with Technology*, 3(1):99, 06 2014.

[9] Laurie Williams, D. Scott McCrickard, Lucas Layman, and Khaled Hussein. Eleven Guidelines for Implementing Pair Programming in the Classroom. In *Agile 2008 Conference*, pages 445–452. EBSCOhost, 2008. doi:10.1109/Agile.2008.12.

[10] Daniel Zingaro and Leo Porter. Peer Instruction in Computing: The Value of Instructor Intervention. *Comput. Educ.*, 71:87–96, 2 2014.

Infusion of Machine Learning Concepts and Skills: A Workshop Review*

Sambit Bhattacharya, Bogdan Czejdo, Valentin Milanov
Department of Mathematics and Computer Science
Fayetteville State University
Fayetteville, NC 28301
{sbhattac,bczejdo,vmilanov}@uncfsu.edu

Abstract

With the recent successes of research in Artificial Intelligence (AI), specifically Machine Learning (ML), infusion of AI and ML concepts and use of tools can help increase the responsible adoption of AI and ML in different disciplines. We report on the design of a workshop on ML, results from the evaluation of the workshop, and suggests topics and pedagogical approaches that may be adopted for disseminating the core concepts of ML which are among the most prevalent data-driven modeling approaches in AI. The workshop individually targeted a diverse range of participants including college and university faculty members from different disciplines at 2-and-4 year institutions. The materials of the workshop were made publicly available. Future workshops and educational modules building on this work will be able to infuse knowledge of the frontiers of AI and problems & benefits.

1 Introduction

When AI pioneer Geoff Hinton says "Deep learning is going to be able to do everything," [7] many are likely to take notice, but understanding and agreeing on what exactly he means by everything and what it means to be able to do those things can become a very challenging task. Dr. Hinton is in a group of esteemed individuals who have been making similar statements. Even after

these pronouncements there seems to be a significant number of examples where the actions of AI in the real world are of major concern [13].

Disseminating AI knowledge is an important goal and institutions of higher education need to lead in this effort by engaging educators and researchers from multiple disciplines. The overall goal of the workshop development and implementation project reported in this article was to conceptualize, design, and implement a model for individually targeted infusion of Machine Learning (ML) concepts and skills. It describes the experiences of the authors who organized a summer workshop (July 20-24, 2020) on teaching ML techniques at Fayetteville State University (North Carolina). The foundation for launching this effort targeted professors at 2-and-4-year colleges and universities. It had two goals: 1) Increase understanding of the successes of AI and its limitations as described above, 2) Provide an enriching teaching and learning experience using active learning pedagogical techniques.

In addition to existing data trends from national market analyses on current and future jobs in AI and allied fields, the authors collected their own data to better understand and assess the local demand for teaching AI in colleges and universities. This was done by gathering feedback from faculty members from local community colleges and universities using online surveys, personal correspondence, conducting site visits to colleges and universities, doing lectures, and having follow up conversations.

2 The Need for Teaching ML at Colleges and Universities

The authors conducted a series of lectures at various institutions within the University of North Carolina System including: Winston Salem State University, Elizabeth State University, NC A&T State University, and UNC Pembroke. The lectures were attended by both faculty and students from Engineering disciplines and Sciences. The authors also conducted phone conversations with local officials from Cumberland county (where Fayetteville is located) and the adjacent Bladen and Hoke Counties that are primarily rural and underserved counties of the state.

As a follow up to the visits and conversations, an online survey was administered to assess the biggest need area(s). A total of 39 responses were recorded. The most common response (above 70%) was Artificial Intelligence (AI). The respondents indicated that it is a high priority to educate underrepresented students, students from underserved areas such as rural and/or remote communities, and first-generation college-enrolled students, to help prepare them for future education and the STEM workforce and particularly AI within STEM. In line with the directions suggested by the survey respondents, there are conversations happening at international meetings like the AAAI 2020 Workshop

on Equity and Diversity in AI [3]. AI researchers and educators are also researching the discrepancy in the quality of education worldwide, including AI education.

Fayetteville State University is also in the proximity to Fort Bragg, one of the largest military installations in the country. The military personnel and their spouses make up a significant portion of the student population and contractors seek talented graduates. The students trained in computing and developing ML applications are in increasing demand. Job searches on Google and Glassdoor using simple terms such as "data scientist," "AI," "Fort Bragg," and "Fayetteville," yield in 50+ hits.

3 Machine Learning Explainability

Numerous barriers prevent students from acquiring AI skills [3]. Underserved communities typically lack resources—equipment such as computers among other tools. While adequate supply of equipment remains a challenge, another gap is limited physical space for conducting hands-on AI activities such as robotics and distributed sensing for decision making. Other challenges are lack of enough trained instructors qualified to teach AI with qualified members leaving underserved communities in search of better opportunities. Because of these reasons, most underserved communities face critical challenges in educating local students in AI and therefore, fail to accommodate the rapidly changing state of art in AI, especially development of new ML models and their applications.

In addition to the above challenges, there are specific challenges related to the limited explainability of AI models [12]. AI models are very heterogenous requiring a substantial effort to construct an integrated teaching approach. Explainability is a growing area of research in ML [4, 11]. Most of the research is oriented towards explainability as an indicator of limitations of ML models in various disciplines [14, 10, 15]. ML explainability is strongly related to educational "explainability." In order to design the individually targeted AI workshop, the authors have extracted from the literature a taxonomy of explainability components [6, 5]that directly relate to the educational applications and outcomes to help construct educationally-oriented ML "explainability." An important characteristic of the workshop plan was to emphasize diversified ML content and teaching methods that are responsive to the needs of learners from both Computer Science and other disciplines.

Several core elements of each ML process and the taxonomy of the "explainability" of ML processes in a broader sense are presented in the table below. According to this classification, "transparency" is the first component and it addresses the fundamental issue of understanding any ML process. In a basic ML

process, a model is learned from the given input data and with a specific learning algorithm. Then, in the next phase, the output results are computed using the learned model. Transparency of an ML process can therefore be further classified into the transparency of an overall process structure and function, the transparency of individual process components, the transparency of the learning algorithm, and transparency of how the specific solution is obtained by the algorithm. This document refers to these four transparency categories as shown in Table 1 as: process transparency, component transparency, transparency of learning algorithm, and transparency of testing algorithm.

Informally, full transparency is the opposite of the "black-box" restricted visibility, where using an ML model to perform predictions is the main goal, but the question of how the model outputs its predictions is left unanswered. Transparency thus indicates the level of understanding on how the model actually works. Transparency was explained further by referring to simulatability (considering the entire model) decomposability (considering individual components) and algorithmic transparency for the learning and testing algorithm. All transparency categories determine how well each component's function and structure can be explained in understandable terms to a human.

The analysis of the role of hyper-parameters is also important in order to consider all ML process complexities. Hyper-parameters can determine general model structure, components, optimization algorithm, learning rate, and the size of stochastic sampling, and are often chosen in a heuristic fashion [12]. Due to the possible presence of several local minima, the solution is usually not easily reproducible; therefore, these decisions are not transparent. They can be considered as part of the transparency of the process or associated with the learning algorithm. Transparency of ML processes is an important part of student learning knowledge since this knowledge is needed to optimize the models with respect to high accuracy. Recently there is also growing need to provide the explanation of output results with respect to domain knowledge [11].

In addition to "transparency," another core element of each ML process is identified in Table 1. This category is titled "explainability for domain knowledge" and it requires techniques to discover the underlying reasons for the ML to produce specific results. The explanability of a complex ML process can be improved by using simpler models with only less accurate result. Traditionally, decision trees were used for the explanation of results. More recently the model-agnostic module LIME [11] by Ribeiro et al. was proposed. In the latter case, feature importance can be detected by local linear proxy model in the neighborhood of a focused data. The educational content analyzed here, namely ML processes, is diversified in terms of explainability as shown in the Table 1. Such content is significantly different from content in other areas/dis-

ciplines and making the educational process effective requires special effort. Frequently, ML processes cannot be easily explained, and the user has to deal with "black-box" models. Mapping the hard-to-explain content with the proper pedagogical methods is critical. There are multiple pedagogical approaches to teaching various algorithms, procedures and rules developed for the "black-box" or "gray box" approach.

4 POGIL as a Teaching Method To Address ML Teaching Challenges

The Process Oriented Guided Inquiry learning (POGIL) teaching technique [9] supported the authors efforts and help them design hands-on activities in AI education. They aligned explainability (Table 1) into the POGIL teaching method. As a result, they were able to choose the best teaching method instead of applying one solution that may not fit all.

Table 1: Taxonomy of AI Processes Explainability

Model/ Algorithm	**Transparency Of Whole Process**	**Transparency Learning Algorithm**	**Transparency Testing Algorithm**	**Transparency Hyper-Parameter**	**Explainability for domain knowledge**
Linear Regression	High	High	High	High	High
Decision Tree	High	High	High	High	High
SVM	Moderate to High	Moderate	Moderate to High	Low	Low
Generic NN	Moderate to High	Moderate	Moderate to High	High	Low
Generic Convolutional Networks	Low to Moderate	Low to Moderate	Low to Moderate	Low to Moderate	Low to Moderate
Generic NL Deep Learning	Low	Low	Low	Low	Low

As a pedagogical approach, POGIL allows students to socially construct knowledge through iterative cycles that include three steps: exploring a model, inventing a concept, and applying the resulting ideas. A growing body of research indicates that relative to lecture-based approaches, POGIL supports student learning more effectively [8]. It allows for a structured approach that requires participants to work in self-managed or regulated teams to explore content, ask questions, solve problems, conduct analysis, record the proceedings, and cooperate to draw valid conclusions. While there are any number of student-centered active learning classroom techniques, POGIL is unique in that it makes students responsible for their own learning, in collaborative teams, so it helps them develop group process skills while they are gaining content knowledge.

Within the POGIL methodology, the authors used an AI platform called Google Colab which hosts notebook service for AI and Data Science [2]. The main intent behind using the POGIL technique was to enable workshop participants to learn and work in teams, collaborate to understand a concept and solve a structured problem or set of questions rather than being given the content via a lecture by a teacher. In summary, the POGIL approach included: 1) Faculty generated model and related content and 2) Specific problem or defined set of questions for small groups to solve/answer with little guidance from the facilitator.

5 Workshop Organization

In summer 2020, the authors of this paper hosted a week-long workshop titled Strategies to Train and Engage Students In Artificial Intelligence. The workshop brought together educators from across North Carolina who learned to use new tools and strategies for teaching AI that would benefit students from diverse backgrounds [1]. The workshop was evaluated by MN Associates, Inc. (MNA). A total of 19 participants engaged in hands-on sessions offered at the AI workshop. Of these, one of the educators was from a community college and 18 were from various universities. The following were goals of the workshops:

- Develop strategies to provide AI knowledge and skills to college students
- Develop techniques for exposing students to the frontier of AI knowledge–problems & benefits emphasizing Machine Learning
- Developing and applying AI computing frameworks including Cloud computing for ML
- Using Python with AI libraries
- Integrate AI computer coding in teaching methods

In the pre-workshop phase the authors and MNA co-developed an online participation interest form (pre-survey) that was sent to various UNC system colleges, universities, and high schools to help recruit workshop participants. Questions related to demographic questions as well as prior/current knowledge of AI topics such as Regression, Decision Tree, and Artificial Neural Networks were posed. Upon completion, the authors extended a formal e-invite to the participants in preparation of the workshop.

Daily post-survey questions were posed as a series of multiple-choice and open-comments questions related to each day's sessions. In both pre- and post-workshop surveys, the participants were asked to rate on a 5-point Likert scale (Low to Very High) the level of their understanding of three topics- Regression, Decision Tree, and Artificial Neural Network before and after the workshop.

6 The Workshop and Results

During the workshop three model building approaches were introduced: Regression, Decision Tree, and Artificial Neural Networks. The materials from this workshop are available on Github [1].

6.1 Regression

The relatively well-known regression model was introduced first. Understandably, the initial level of the understanding of the topic among the participants was already very high compared to other topics. Still, majority of participants reported that their understanding of regression model improved after the exposure to POGIL lessons. During this phase of the workshop it was evident that there were two relatively different groups of participants in respect to previous exposure to ML concepts. The percentage points change in understanding from pre-to-post workshop is shown in Figure 1.

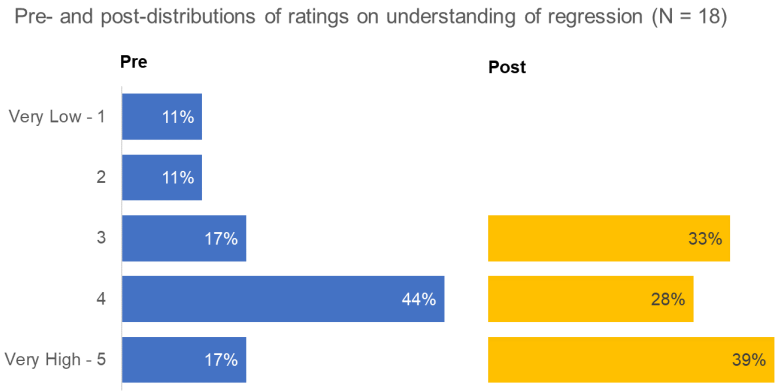

Figure 1: The percentage points change in understanding from pre-to-post workshop for regression model.

6.2 Decision Tree

The less-known (relative to Regression) Decision Tree learning model was introduced next. Surprisingly, the initial level of the understanding of the topic among the participants was already high compared to other topics. Our hypothesis is that the answers were related to using the decision tree and not necessary constructing it in a bottom-up method from labeled data, as done

in ML. Based on the survey, majority of participants also reported that their understanding of Decision Trees model improved after the exposure to POGIL lessons. During this phase of the workshop it was observed that the first group moved from level 1 and 2 to level 3 and the second group moved from level 3 to 4 or 4 to 5. There were no big jumps as in the case of regression but smaller and consistent growth of reported knowledge. The percentage points change in understanding from pre-to-post workshop is shown in Figure 2.

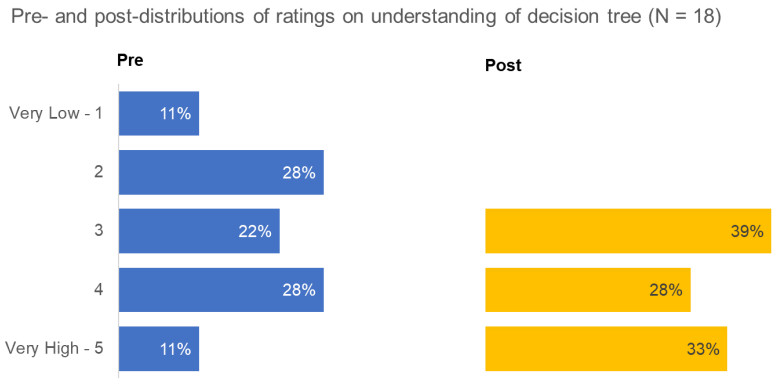

Figure 2: The percentage points change in understanding from pre-to-post workshop for Decision Tree model

6.3 Artificial Neural Network (ANN)

The ANN model was introduced last. The results of workshop here show a considerable impact. First of all, the initial level of the understanding of the topic among the participants was much lower compared to other topics. The majority of participants also reported that their understanding of ANN model improved significantly after the exposure to POGIL lessons. The percentage points change in understanding from pre-to-post workshop is shown in Figure 3.

7 Summary

Based on the assessments conducted by the authors during a summer workshop on teaching AI, results from the evaluation of the workshop, and understanding the burgeoning needs of AI education, this article suggests topics and pedagogical approaches that may be adopted for disarming the core concepts of

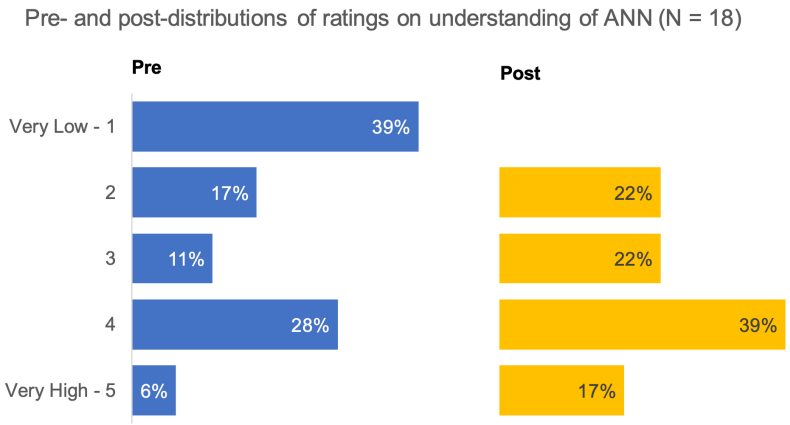

Figure 3: The percentage points change in understanding from pre-to-post workshop for ANN model

ML which are among the most widely used data-driven modeling approaches in AI. These workshops may be individually targeted to a diverse range of participants including both Computer Science and non-Computer Science students, and college and university faculty members from different disciplines at 2-and-4 year institutions. The content of future workshops and educational modules should continue to expose students to the frontier of AI knowledge and problems & benefits.

Acknowledgements

This article is based upon work supported by the National Science Foundation under Grant No. 1818694.

References

[1] AI workshop. `https://github.com/sbhattac/ai-workshop`.

[2] Welcome to colaboratory. `https://colab.research.google.com/`.

[3] AAAI. AAAI-20 call for diversity and inclusion activities. `https://aaai.org/Conferences/AAAI-20/aaai-20-diversity-inclusion/`.

[4] Robert Andrews, Joachim Diederich, and Alan B Tickle. Survey and critique of techniques for extracting rules from trained artificial neural networks. *Knowledge-based systems*, 8(6):373–389, 1995.

[5] Alejandro Barredo Arrieta, Natalia Díaz-Rodríguez, Javier Del Ser, Adrien Bennetot, Siham Tabik, Alberto Barbado, Salvador García, Sergio Gil-López, Daniel Molina, Richard Benjamins, et al. Explainable artificial intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI. *Information Fusion*, 58:82–115, 2020.

[6] Riccardo Guidotti, Anna Monreale, Salvatore Ruggieri, Franco Turini, Fosca Giannotti, and Dino Pedreschi. A survey of methods for explaining black box models. *ACM computing surveys (CSUR)*, 51(5):1–42, 2018.

[7] Karen Hao. AI pioneer Geoff Hinton:"deep learning is going to be able to do everything", 2020. `https://www.technologyreview.com/2020/11/03/1011616/ai-godfather-geoffrey-hinton-deep-learning-will-do-everything/`.

[8] Helen H Hu, Clifton Kussmaul, Brian Knaeble, Chris Mayfield, and Aman Yadav. 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*, pages 186–191, 2016.

[9] Clifton Kussmaul, Helen H Hu, and Matthew Lang. Guiding students to discover CS concepts and develop process skills using POGIL. In *Proceedings of the 45th ACM technical symposium on Computer science education*, pages 745–745, 2014.

[10] Jesus Mena. *Machine learning forensics for law enforcement, security, and intelligence*. CRC Press, 2011.

[11] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. " why should I trust you?" explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1135–1144, 2016.

[12] Ribana Roscher, Bastian Bohn, Marco F Duarte, and Jochen Garcke. Explainable machine learning for scientific insights and discoveries. *IEEE Access*, 8:42200–42216, 2020.

[13] Cynthia Rudin. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence*, 1(5):206–215, 2019.

[14] Cynthia Rudin and Berk Ustun. Optimized scoring systems: Toward trust in machine learning for healthcare and criminal justice. *Interfaces*, 48(5):449–466, 2018.

[15] Jacky Visser, John Lawrence, and Chris Reed. Reason-checking fake news. *Communications of the ACM*, 63(11):38–40, 2020.

Hey Alexa! A Fun Introduction to AWS Lambda, Serverless Functions, and JavaScript with Conversational AI*

Conference Workshop

Denise M Case
Northwest Missouri State University
Maryville, MO 64468
dcase@nwmissouri.edu

Learn how to design and create engaging cutting-edge apps! We'll introduce principles of designing apps that apply freely available conversational AI tools (e.g., Alexa, Google Home, and Siri) [2]. We'll explain the powerful and easy 'serverless' functions that power them - no programming experience required! We'll introduce cloud computing and show the benefits of serverless functions [1]. Along the way, participants will open an Amazon Web Services account and get started as an Alexa developer for free!

We'll show how to use AWS Lambda to run code without worrying about servers, complex file transfers, or setup. We'll show you how to easily upload and edit simple web-enabled functions. You'll learn how Alexa can be used to illustrate basic programming concepts like programming flow and enumerated types. Participants get access to the project code for the reference app, checklists for working through the steps to publish, and if they like, can leave with a newly submitted Alexa skill customized for their organization or institution.

References

[1] Simon Eismann, Joel Scheuner, Erwin van Eyk, Maximilian Schwinger, Johannes Grohmann, Nikolas Herbst, Cristina Abad, and Alexandru Iosup. Serverless applications: Why, when, and how? *IEEE Software*, 2020.

[2] Ashwin Ram, Rohit Prasad, Chandra Khatri, Anu Venkatesh, Raefer Gabriel, Qing Liu, Jeff Nunn, Behnam Hedayatnia, Ming Cheng, Ashish Nagar, et al. Conversational AI: The science behind the Alexa prize. 2018.

Short Modules for Introducing Heterogeneous Computing*

Conference Tutorial

David P. Bunde[1], Apan Qasem[2], Philip Schielke[3]
[1]Knox College
Galesburg, IL 61401
dbunde@knox.edu
[2]Department of Computer Science
Texas State University
San Marcos, TX 78666
apan@txstate.edu
[3]Concordia University Texas
Austin, TX 78726
Philip.Schielke@concordia.edu

Abstract

CS faculty have spent the last several years adding parallel computing to their curricula since essentially all processors sold today have multiple cores. A typical target system is a multicore processor with identical cores. This is the configuration for most current desktop and laptop systems, but the technology continues to evolve and systems are incorporating heterogeneity. Many phone processors include cores of different sizes so the phone can vary its power and performance profile over time. Other processors incorporate low-power modes or instructions for specialized computations. Meanwhile, high-end systems make heavy use of accelerators such as graphics cards. We are at a stage where heterogeneous computing concepts should pervade the curriculum rather than being limited to upper-level courses.

This tutorial motivates heterogeneous parallel programming and then presents modules that introduce aspects of it such as energy/performance tradeoffs,

SIMD programming, the benefit of memory locality, processor instruction set design tradeoffs, and CPU task mapping. Each module uses only a few days of class time and includes assignments and/or lab exercises which are available online (`https://github.com/TeachingUndergradsCHC/modules/`). Here are the modules:

1. The first module shows the challenges and benefits of task mapping on a heterogeneous system. The module includes a lab to provide students with hands-on experience running parallel workloads in heterogeneous environments. It is aimed at CS 2, but also fits in Systems and Parallel Programming courses.

2. The second module looks at heterogeneity on ARM processors, particularly Thumb mode, a low-power mode with restricted instructions. The module is based on the Raspberry Pi, a low-cost system aimed at hobbyists. It highlights performance/power tradeoffs and is aimed at Computer Organization.

3. The third module shows how memory locality can improve performance on a program that uses CUDA to run on a graphics processing unit (GPU). This module demonstrates heterogeneity resulting from both CUDA's SIMD model of computing and the different memory types on a GPU. It highlights memory locality and is aimed at systems-oriented courses.

Acknowledgements

This tutorial presents work supported by NSF grants OAC-1829644 & OAC-1829554.

How Can It Be Fixed in Terms of Typing?*

Nifty Assignment

Cong-Cong Xing[1], *Jun Huang*[2]
[1]*Department of Mathematics-Computer Science*
Nicholls State University
Thibodaux, LA 70310
cong-cong.xing@nicholls.edu
[2]*School of Cyber Security*
Northwest Polytechnic University
Taichang, China 215411
huangj@ieee.org

Motivation It is well known that C is a somewhat weakly typed (not strongly typed) language. However, it is not always clear how to let students (1) realize that this "weak typing feature" of C could potentially cause issues and (2) consequently understand the importance of the "strong typing" advocate in programming languages. This nifty assignment presents an exercise that aims to do just that.

Overview Traditionally, C does not have an *authentic* boolean type notation and the associated *true* and *false* constants[1], although it offers the if-else statement where a boolean-typed expression would be needed, and the comparison expressions where a boolean-typed result would be produced. In C, "zero values" (e.g., 0) are treated as *false*, and any "non-zero values" (e.g., 1) are treated as *true*. While this mechanism works fine in most situations, it does cause some strange problems in some cases. For example, consider the expression `3<7<5`. Not only is `3<7<5` legal in C, but it evaluates to 1 (true) in C as well. Now, the result 1 (true) produced by `3<7<5` may seem to be puzzling, strange, or "counter-mathematical" to students, since `3<7<5` is mathematically false. What transpires here is that C treats `3<7<5` as `(3<7)<5`. As `3<7` evaluate to 1 (true), `(3<7)<5` is reduced to `1<5`, which subsequently evaluates to 1

[1]In C99, words *true* and *false* can be recognized as *bool*-typed values with the inclusion of `<stdbool.h>`; but they are just synonyms of 1 and 0. No fundamental changes are made.

(true). So, naturally, we would be wondering: *What can conceivably be done in C (in terms of typing) so that this "counter-mathematical" expression will be statically blocked by the C compiler (especially noticing that expressions such as* `a<b<c` *will not compile in Java)?*

Classroom Observations and Possible Gains

This question/problem was discussed in a junior-level course on Unix and C Programming, and students in that class were later asked to propose their solutions to this problem in writing. Surprisingly, most students failed to propose introducing an *authentic boolean type (something similar to that in Java)* and the associated typing rules into C so that `3<7<5` will be rejected by the type-checker of the compiler. In particular, they failed to see that a newly introduced authentic boolean type with values (*true* and *false*) is able to tag `3<7` as a boolean-typed (sub)expression, and that the expression `3<7<5` (parsed as `(3<7)<5`) will be subsequently blocked by the compiler for typing errors since a boolean-typed expression or value is not comparable with an int-typed expression or value.

Rather, most of the students proposed to let the C compiler disallow expressions of the form of `a<b<c` and force them to be rewritten as `a<b && b<c`. Although this solution appears to be able to "solve" the `3<7<5` problem since `3<7 && 7<5` would yield 0 (false) now, it is unfortunately not implementable as long as the authentic boolean constants *true* and *false* are delegated as integers 1 and 0 in C. To see this, we begin with the assumption that expressions such as `a<b` where `a` and `b` are of type int, are allowed in C. By the way that a type system works, any int-typed expressions can be put into the positions of `a` and `b` in `a<b` to construct new expressions. As such, noticing that the expression `3<7` is of type int (as it evaluates to 1), it will be allowed to be put into the position of `a` in `a<b`, resulting in the expression `3<7<b`. Thus, expressions of the form of `a<b<c` are an inevitable consequence of the combination of allowing expressions of the form of `a<b`, and delegating boolean values *true* and *false* as integers 1 and 0. The fact that most students missed this point suggests that this problem might not be as simple as it looks, and it might trigger something which requires a deeper understanding on how type-checking works.

Potential gains from this assignment may include:

- A realization and appreciation of the importance of strong typing (vs. weak typing).
- An improved understanding on how the type-checking mechanism works.
- Inspired interests in advanced courses such as compiler construction or theory of programming languages.

We hope that this assignment can be found useful by colleagues.

Creating a Server as a Nifty Assignment*

Nifty Assignment

Samuel Hsieh and Sean Wolfe
Computer Science Department
Ball State University
Muncie, IN 47306
{shsieh, stwolfe}@bsu.edu

A server is a computer program that provides a service to concurrently running clients. Nowadays using a server, such as a Web server, an email server, etc., is a frequent, real-life activity for most Internet users. Creating a server should be a learning experience in an undergraduate program in computer science. Since a server and its clients usually run concurrently, this topic naturally belongs to a course that has a concurrency component, such as operating systems. An assignment to create a server familiarizes students with a conceptual model of a server and provides students with an opportunity to instantiate the model.

An often-used conceptual model is that of a receptionist and a service-providing staff. To service a client's request, a server needs to perform two tasks: first to connect with the requesting client and then to actually provide the service to the client. The first task is like the job of a receptionist who takes incoming phone calls from clients. Once the receptionist is in touch with a client, the call is transferred to an appropriate employee, who provides the actual service to the client. Hence, within a sever, there is a thread that works like a receptionist. Once this receptionist thread connects with a client, it creates a new thread, to be referred to as a client handler, that actually services the client's request. In other words, the process of servicing a request is a two-part dialog. In the first part, the receptionist and the requesting client establish a connection, and in the second part, a client handler delivers and the requesting client receives the requested service.

Normally multiple client requests are made concurrently and thus multiple client handler threads run concurrently, servicing the clients' requests simultaneously. So a basic program structure of a server consists of a receptionist thread, which, for example, may conveniently be the main method of the server

program, and a client handler class, of which many client handler threads can be created. Since a server and its clients usually run on separate computers, communication between the server and the clients takes place via message passing.

As for the service to be provided by the server to be implemented in a server-writing assignment, a simple but still interesting example is a server that provides the following service: within 10 guesses the server correctly determines an integer between 1 and 1000 that a client has in mind. The server should be able to serve many clients concurrently. Implementing the receptionist thread and the client handler class in Java is a reasonable programming assignment in an upper-division semester-long 3-credit course in operating systems that requires several programming assignments.

A simple way to test such a server is to telnet to the server. For example, the following is a dialog with such a server via telnet. The client has the number 345 in mind. The name `Sam` and the responses `H`, `L` and `C` are entered by the client and all other text in the dialog is from the server.

```
What's your name?
Sam
Hi Sam. In your mind, choose an integer in 1..1000.
I will show your chosen number within 10 guesses.
500 ? Enter C if correct, H if too high, or L if too low
H
250 ? Enter C if correct, H if too high, or L if too low
L
375 ? Enter C if correct, H if too high, or L if too low
H
312 ? Enter C if correct, H if too high, or L if too low
L
343 ? Enter C if correct, H if too high, or L if too low
L
359 ? Enter C if correct, H if too high, or L if too low
H
351 ? Enter C if correct, H if too high, or L if too low
H
347 ? Enter C if correct, H if too high, or L if too low
H
345 ? Enter C if correct, H if too high, or L if too low
C
Thank you, Sam
```

Obviously, the server uses the bi-section method (i.e., the interval-halving method) to find the client's chosen integer. Since $log_2 1000 < 10$, the server is

able to do so within 10 guesses. When the server detects inconsistency in a client's responses, the server informs the client with a message, such as `Hey, you cheated`.

We note that such a server may create and destroy a large number of threads. This overhead can be controlled by using a thread pool to reuse threads and thereby to limit the number of threads that the server creates. This enhancement is also an appropriate programming exercise.

The Role of Technology in Digital Forensics Investigations*

Nifty Assignment

Rad Alrifai
Mathematics and Computer Science
Northeastern State University
Tahlequah, OK 74464
alrifai@nsuok.edu

Abstract

Technological artifacts can be collected as evidence, used by investigators, or misused by suspects. The presented assignment is aligned with one of several topics covered in introduction to digital forensics, a required course for students either majoring or minoring in cybersecurity. The assignment consists of three different components: a technology component, a legal case study, and a research component. The assignment combines computing topics related to digital forensics with criminal justice topics that are related to actual legal cases with at least one piece of digital forensic evidence. The students will also research a topic related to the role of technology in a digital forensics investigation to expand their knowledge beyond the scope of the course.

1 Introduction

Digital forensics represents a relatively new area in forensic science that focuses on discovery, investigation, and collection of evidence found on digital devices. The course is part of a cybersecurity undergraduate program offered as an interdisciplinary degree consisting of courses from computer science, criminal justice, and information systems. The presented assignment is one of several different assignments that share a similar structure and are assigned to the same cohort of students who are enrolled in the introduction to digital forensics course at a regional university. As illustrated in figure 1, each assignment is designed to contribute to what students should know, be able to do, and

consider by completing three different tasks: a research component, a hands-on lab, and a legal case study.

2 The Assignment

Each assignment consists of three different components: a legal case study, a technology component, and a research component.

2.1 The Legal Case Study Component

The first part of the assignment consists of a legal case study intended to engage students and allow them to experience the legal aspects of a digital forensic case. In this component, students are assigned an actual legal case that is publicly available on the internet. This component of the assignment reads as follows: There is an argument that says, many cases are lost in court due to the mishandling of digital evidence: read the cases on page 2 of "Digital Evidence and the U.S. Criminal Justice System." [1] Then select one of the cases to answer the following questions:

- Write one to two paragraphs summarizing the main facts about this case.
- Briefly describe the primary issue in this case as it relates to digital evidence.
- Identify the primary legal issues.
- What was the court ruling on the case?
- Briefly comment on any constitutional amendments or explicitly referenced laws that are related to this case.
- Briefly, summarize your reaction and state if you were surprised by the court ruling on this case, why/why not?

2.2 The Hands-on Technology Component

The hands-on technology section of the assignment is to allow students to experiment with hardware and software components similar to those used in a digital forensics investigation. In this assignment, students were asked to install and investigate the use of virtual boxes in a digital forensic investigation. Students are requested to follow the steps below to complete the assignment:

- Install VirtualBox. Verify that you have completed the installation by capturing a screen of the VirtualBox running on your machine. Include the captured image in your homework submission.
- Describe the benefit of using a virtual box or similar software in a lab used for examining digital forensic evidence.

- Describe challenges related to investigating a computer for evidence when the computer has a virtual box installed on it.

2.3 The Research Component

The research section of the assignment is intended to encourage students to learn independently by researching new topics to expand their knowledge about a challenge that they may face but are not currently familiar with. An example of this component in the assignment is related to the use of technology in encountering crime. Students are asked to learn about how emerging technologies such as social media, cryptocurrencies and blockchain applications, cloud computing, the Internet of things, or big data analytics can be the source of both opportunities and challenges in a digital investigation. The students are asked to read the online article "Emerging Technology Trends and Their Impact on Criminal Justice" [2] and write a one-page summary to describe the use, limitations, and legal and technical issues related to the use of an emerging technology in a digital forensics case.

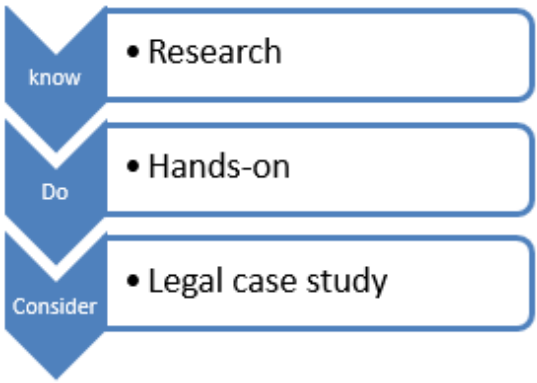

Figure 1: Assignment Goals and Corresponding Modules.

References

[1] Sean E Goodison, Robert Carl Davis, and Brian A Jackson. Digital evidence and the us criminal justice system. identifying technology and other needs to more effectively acquire and utilize digital evidence. 2015. `https://www.rand.org/pubs/research_reports/RR890.html`.

[2] John S Hollywood, Dulani Woods, Andrew Lauland, Brian A Jackson, and Richard S Silberglitt. *Emerging Technology Trends and Their Impact on Criminal*

Justice. RAND, 2018. https://www.rand.org/pubs/research_briefs/RB9996.html.

Engaging Graduate Students During the Pandemic*

Panel Discussion

Charles Badami[1], *Ajay Bandi*[1], *Denise Case*[1]
Aziz Fellah[1], *Mahmoud Yousef*[2]
[1]*Northwest Missouri State University, Maryville, MO 64468*
{cbadami, ajay, dcase, afellah}@nwmissouri.edu
[2]*University of Central Missouri, Warrensburg, MO 64093*
yousef@ucmo.edu

This panel discussion will look at what our graduate students need to stay engaged and productive during the pandemic. Many graduate students are nearly a half a world away from home while earning degrees, working, and studying. This discussion looks at various teaching practices and approaches that can help our students practice their skills in this challenging environment. We will look at how to incentivize the high-performing students, how to engage the many attending, but possibly quiet students, and look at what measures can be taken to help engage students who might have temporarily "checked out". This discussion will ask about techniques tried and results obtained, and help us improve our courses while mitigation measures continue. We'll be specific—synchronous/asynchronous, video-on/video-off, rewards-or-not for engagement/attendance/participation, and how to assess performance while minimizing cheating and/or circumventing the learning process. Practice ideas may come from recent research [1, 2] and personal experience. Faculty benefit from learning how students responded to different approaches in teaching.

References

[1] Elizabeth F Barkley and Claire H Major. *Student engagement techniques: A handbook for college faculty.* John Wiley & Sons, 2020.

[2] Ashok Kumar Veerasamy, Daryl D'Souza, Rolf Lind, Erkki Kaila, Mikko-Jussi Laakso, and Tapio Salakoski. The impact of lecture attendance on exams for

novice programming students. *International Journal of Modern Education and Computer Science*, 8(5):1, 2016.

The Impact of COVID-19 on Industry and Education*

Panel Discussion

Joni Adkins[1], Diana Linville [1], Bilal Clarence[2], Shanon Elliott[3], Sean Paddock[4]
[1]School of Computer Science and Information Systems
Northwest Missouri State University
Maryville, MO 64468
{jadkins,dianar}@nwmissouri.edu
[2]Capital One Financial
Berkeley, CA 94703
bclarance@gmail.com
[3]Waddell and Reed
Overland Park, KS 66202
selliott@waddell.com
[4]Federal Reserve Bank of Kansas City
Kansas City, MO 64198
Sean.Paddock@kc.frb.org

COVID-19 has disrupted the entire functionality of the world. It has changed people's daily activities and challenged us to adjust our life styles to accommodate the need for social distance. This in turn has impacted how industry and education operate. The overall structure and procedures within industry and education have had to conform to this new normal. Our panel will explore and discuss the impacts that COVID-19 has had on industry and education, including how decisions made by one entity impacts the other. We want to (1) address the disruptions to employees and the challenges they face, (2) explore impacts on the interview and hiring processes and how it affects graduates, (3) review changes in productivity across the different industries, (4) examine the impact to leadership and promotion opportunities, and (5) consider the challenges and adjustments for businesses, educators, and students.

Our panel will examine how these impacts affect the job outlook, recent and future graduates, and the skill sets that are now needed.

Joni Adkins is an Associate Professor in the School of Computer Science and Information Systems at Northwest Missouri State University and has been teaching for 20 years. She teaches information systems courses at the graduate and undergraduate level and serves as the School's Assistant Director. She is a past president of Delta Mu Delta, a business honor society and helps coordinate the annual Professional Advisory Team meeting.

Diana Linville is a Senior Instructor in the School of Computer Science and Information Systems at Northwest Missouri State University and has been teaching for ten years. She has taught a range of undergraduate courses and is the computer programming I coordinator. She has served as a professional advisory team coordinator for eight years, is an advisor for ACM, serves on the Maryville R-II technology advisor board, and is involved with the NC-WIT Aspirations in Computing Awards.

Bilal Clarance is a senior director of Data Engineering at Capital One, he spent nearly 7 years at Apple leading teams in the Apple Online Store. Bilal has worked for various startups and shortly ventured to build his own business. He is a northwest alumn and played for coach Tappmeyer on the Northwest men's basketball team. He then spent time playing overseas before settling down in California with his wife Dia Clarance (maiden name McKee), also a former collegiate athlete and Northwest alumn.

Shanon Elliott recently joined Waddell & Reed as the Director of Application Development and Data Integration for Asset Management. Prior to joining Waddell & Reed during the covid shutdown, he spent the last 20+ years in for American Century Investments where he also served as IT Intern coordinator. He has served on Northwest's Professional Advisory Team for the past 10 years.

Sean Paddock is a manager at the Federal Reserve Bank of Kansas City, where he's served in a variety of roles across the organization. He currently leads a cross-functional team focused on improving business outcomes through more effective use of data. Sean has also served as an adjunct instructor at Rockhurst University for the past three years, where he's taught a variety of courses within university's Masters in Data Analytics program.