The Journal of Computing Sciences in Colleges

Papers of the 36th Annual CCSC Eastern Conference

October 23rd-24th, 2020
Hood College
Frederick, MD

Baochuan Lu, Editor
Southwest Baptist University

John Wright, Regional Editor
Juniata College

Volume 36, Number 3 **October 2020**

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

Table of Contents

The Consortium for Computing Sciences in Colleges Board of Directors

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

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, School of Computing and Engineering, 5110 Rockhill Road, 546 Flarsheim Hall, University of Missouri - Kansas City, Kansas City, MO 64110.

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 "Crystal" Peng, Associate Editor, (816) 584-6884, crystal.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 2020 CCSC Eastern Conference

On behalf of the CCSCE 2020 Conference Committee, we would like to extend a warm welcome to those attending this 36th Annual Conference.

This year's conference was scheduled to be held at Hood College in Frederick, Maryland. In light of a global pandemic, our options were to cancel the conference or find a different, safe venue. The organizing committee decided to move the conference online, and we are delighted to see the response and everyone's participation. It proves that CCSCE is not just a conference but a community. Besides being the first CCSCE held online, this conference is a real testament to our community's ability to problem-solve, persevere, and move forward in the face of adversity.

Thank you to the educators from all levels of the computing sciences, including computer science, information systems, information technology, and so on, and the students in the related fields for attending the conference and contributing to the success of this Eastern Region Conference.

With the contributions from many of you, we have two days of excellent programs planned for the professional enrichment of our audiences, which include an invited keynote, paper presentations, workshops, tutorials, poster presentations, and a programming competition for the students. This year the conference had 26 professional paper submissions, out of which we have accepted 14 papers for an acceptance rate of 54%. All papers underwent a double-blind review process with on average papers that were reviewed by 3 reviewers.

The conference is supported by faculty from multiple institutions who served on the Conference Committee, as reviewers, etc. We want to express our sincere gratitude to everyone involved in making this conference a reality. Special thanks to all those who helped develop the program, coordinate the paper reviews, organized the programming competition, coordinated the keynote speaker, managed the production of the proceedings, coordinated panels, workshops, tutorials, nifty ideas, and lightning talks. Our gratitude also goes to the judges for the poster awards, the session chairs, and student volunteers. We also appreciate the continuous effort and support from the CCSC Eastern Region Steering Committee, and we are very grateful for the generous supports of the CCSC National Partners, Sponsors, and Vendors.

Lastly, this conference is unique because we also have a bitter-sweet "change of guard." Our colleague, John Wright (Juniata College), who has been the steward of several CCSCE conferences over the years, is stepping down as the CCSC Eastern Representative. John's contributions to our community and the CCSCE conferences are innumerable, and his work and always helpful disposition has been nothing but exemplary. We thank him for his tireless support and welcome Mike Flinn (Frostburg State University) as the new Representative.

We hope you have an excellent and productive conference!

George Dimitoglou and Jiang Li
Conference Co-Chairs
Hood College

2020 CCSC Eastern Steering Committee

Elizabeth Adams ..... James Madison University
Steven Andrianoff ..... St. Bonaventure University
Karen Anewalt ..... University of Mary Washington
George Benjamin ..... Muhlenberg College
Elizabeth Chang ..... Hood College
Vincent Cicirello ..... Stockton University
Michael Flinn ..... Frostburg State University
Sister Jane Fritz ..... St. Joseph's College
Nathan Green ..... Marymount University
David Hovemeyer ..... Johns Hopkins University
John Meinke ..... University of Maryland University College
Karen Paullet ..... Robert Morris University
Donna Schaeffer ..... Marymount University
Jennifer Polack-Wahl ..... University of Mary Washington
John Wright ..... Juniata College

2020 CCSC Eastern Conference Committee

George Dimitoglou, Co-Chair ..... Hood College
Jiang Li, Co-Chair ..... Hood College
John Wright, Papers, Web Site ..... Juniata College
Elizabeth Chang, Papers ..... Hood College
George Dimitoglou, Papers ..... Hood College
Pranshu Gupta, Panels, Workshops, and Tutorials ..... DeSales University
Becca Flinn, Posters ..... Frostburg State University
Mian Qian, Posters ..... Frostburg State University
Olumide Kayode, Posters ..... Frostburg State University
Steven Kennedy, Programming Contest ..... Frostburg State University
Dave Hovemeyer, Programming Contest ..... Johns Hopkins University
Donna Schaeffer, Volunteers ..... Marymount University
Nathan Green, Treasurer ..... Marymount University
Michael Flinn, Regional Board Representative .... Frostburg State University

Reviewers — 2020 CCSC Eastern Conference

Rana Alabdan ..... Robert Morris Univ., Moon, PA
Faleh Alshameri ..... Marymount Univ., Arlington, VA
Karen Anewalt Cockrell The Univ. of Mary Washington, Fredericksburg, VA
Ian Barland ..... Radford Univ., Radford, VA
Charles Border ..... Rochester Institute of Technology, Rochester, NY
Elizabeth Chang ..... Hood College, Frederick, MD
David Conner ..... Christopher Newport Univ., Newport News, VA
Ian Finlayson ..... The Univ. of Mary Washington, Fredericksburg, VA
Reva Freedman ..... Northern Illinois Univ., DeKalb, IL
Alessio Gaspar ..... Univ. of South Florida, Tampa, FL
Grigoriy Grinberg ..... Montgomery College, Gaithersburg, MD
Jim Knisely ..... Bob Jones Univ., Greenville, SC
April Kontostathis ..... Ursinus College, Collegeville, PA
Juan Jenny Li ..... Kean Univ., Union, NJ
Edward Lindoo ..... Regis Univ., Denver, CO
Robert Marmorstein ..... Longwood Univ., Farmville, VA
Alex Mbaziira ..... Marymount Univ., Arlington, VA
John McManus ..... Randolph-Macon College, Ashland, VA
John Pais ..... Ladue Horton Watkins High School, St. Louis, MO
Sofya Poger ..... Felician Univ., Rutherford, NJ
Veena Ravishankar ..... The Univ. of Mary Washington, Fredericksburg, VA
Donna Schaeffer ..... Marymount Univ., Arlington, VA
Gregory Schaper ..... Moravian College, Bethlehem, PA
Richard Scorce ..... St. John's Univ., Jamaica, Queens, NY
JoAnna Shore ..... Frostburg State Univ., Frostburg, MD
Sajedul Talukder ..... Edinboro Univ., Edinboro, PA
Ana Valentin ..... Marymount Univ., Arlington, VA
Alla Webb ..... Montgomery College, Gaithersburg, MD

The "What's Next Economy"*

Keynote Address

Jonathan Aberman
Dean, School of Business and Technology
Marymount University, Arlington, VA 22201

The COVID-19 pandemic has changed our society and economy already, with many more challenges and changes to come. Jonathan Aberman, Dean of Marymount University School of Business and Technology is a national expert on innovation, economic trends and entrepreneurship. He believes that the United States has entered a new phase, which he describes as the "What's Next Economy?" He will discuss some of the hallmarks of the What's Next Economy and what it means for technologists, educators and policy makers.

Academic Credentials

Jonathan's academic qualifications are in the fields of economics and law. He earned BA degrees from George Washington University (Political Science and Economics) and Cambridge University (Law). He subsequently obtained a graduate degree in Economics from the London School of Economics (MSc in Economics). He completed his legal education at Cambridge University (MA) and New York University (LLM).

Speaker Bio

Jonathan Aberman is a highly respected thought leader on entrepreneurship and innovation. His experience as a venture investor, innovation consultant, university professor and media commentator gives him a 360-degree perspective on entrepreneurship and technology innovation. He is identified as a leader of change and influence in print and television media, recognized by Washingtonian magazine as a "Tech Titan," by the Washington Business Journal as a member of the "Power 100" and by the Commonwealth of Virginia as one of its "50 Most Influential Entrepreneurs" in the Commonwealth." Forbes Magazine described him as the "unsung hero" of the successful attraction of Amazon HQ2 to Arlington, Virginia.

Jonathan has unique expertise in the melding of national security and entrepreneurship with technology commercialization. He has worked on these matters with a number of national security agencies and universities including DOD Laboratory establishment, DARPA, the Air Force, the Army, the Department of Homeland Security, George Mason University and the University

of Maryland at College Park. His activities regarding entrepreneurship and national security resulted in the formation of Tandem National Security Innovations, a national innovation community that has helped national security entities such as DARPA, DHS, SOCOM, JSOC and others find and engage with nontraditional sources of innovation.

Through his closely held investment business, Amplifier Ventures, Jonathan has helped to start 16 technology businesses since 2005. Among these businesses are successful companies in cybersecurity, mobile communications, internet and mobile content and software. Before starting Amplifier Ventures, he had a 15-year career as a partner and senior manager of a number of national law firms, and prior to that he was an associate in the investment banking industry in London, England. He uses his commercial experience to provide valuable strategic advice, business model and customer discovery acumen and deal management. His experiences and contacts in the venture capital industry inform his views on technology innovation and provide a hard earned commercial perspective that he applies to his work.

Jonathan is a committed educator. Prior to joining Marymount he spent 11 years as a Lecturer of Entrepreneurship at the University of Maryland's Smith School of Business, where he taught a broad range of courses including business formation, corporate finance, business strategy, and family entrepreneurship.

He also has a significant media footprint. He hosts What's Working in Washington, a podcast and weekly radio show on Federal News Network's WFED (AM 1500) examining innovation in one of the least understood business communities in the country. From 2015 until joining Marymount, Jonathan had a regular column, first in the Washington Post and then in the Washington Business Journal, covering the region's business community, innovation, entrepreneurship and economic development. He is often quoted in national publications including Bloomberg, CNN, Wall Street Journal and The New York Times, and he actively participates on panels and serves as a keynote speaker for innovation, entrepreneurship and technology events.

During his career Jonathan has helped to create many successful communities of entrepreneurs. In 2011, he founded Startup Virginia, Startup Maryland and StartupDC as part of the Obama Administration's StartupAmerica Initiative. These efforts resulted in the formation of regionally impactful programs including 1776 Accelerator and the iCorps program. In 2012 he formed FounderCorps, a not for profit that aggregated many of the Greater Washington region's most prominent entrepreneurs and promoted mentorship. In 2018, he formed the Tandem Product Academy, an educational program focused on teaching founders and senior executives essential skills necessary to scale technology product-based businesses.

Programming With the Cloud*

National Partner Session

Laurie White
Cloud DevRel
Google for Education

While there's a lot to learn about cloud computing, the cloud can also be used in classes as fundamental as programming courses with little change to the material being taught. The cloud can provide a uniform programming environment for students regardless of the computers they use to access it remotely. It can provide computing resources beyond what some students may have on their own computers. And there are even some cloud services that can be used to make even the simplest programming assignments more interesting.

Techniques to Effectively Teach a Course Online*

National Partner Session

Yamuna Rajasekhar
zyBooks

With universities in the COVID environment moving to online instruction, many instructors are having to teach courses online. In order to achieve the same student performance as in-person instruction, instructors have to break away from conventional teaching methods and adapt their teaching techniques. This talk presents an overview of zyBooks, which are interactive, online textbooks for the STEM fields that have proven to increase student confidence in STEM courses. The talk also outlines techniques to make online instruction interactive and engaging for the student. Additionally, student performance results are presented from an online computer science course taught at the University of California, Riverside.

About the Speaker: Yamuna Rajasekhar received her Ph.D. in Electrical Engineering from the UNC Charlotte. She served as a faculty member at Miami University where her research was focused on assistive technology, embedded systems, and engineering education. She is currently a Content Developer at zyBooks, a startup that develops highly-interactive, web-native textbooks for a variety of STEM disciplines.

Virtual Cluster for HPC Education*

Linh B. Ngo and Jon Kilgannon
Computer Science
West Chester University of Pennsylvania
West Chester, PA, 19380
{lngo,jk880380}@wcupa.edu

Abstract

For many institutions, it is challenging to procure and maintain resources to teach parallel and distributed computing. While existing dedicated environments such as XSEDE are available, they often have high level of utilization, leading to difficulty in supporting hands-on in-class sessions, especially for larger classes. This work describes the design and development of a Linux-based distributed cyberinfrastructure (CI) to address this problem. Unlike typical production-level environment, the CI is designed to be dynamically customized and deployed on a federal cloud resource. Besides computing, the CI provides a job scheduler, message passing, networked storage, and single sign-on mechanisms. Configurations of these components can be adjusted prior to the automatic installation process during deployment. Scalability is demonstrated, both as the number of cores and shared storage nodes increases, showing that the proposed cluster emulates a large-scale system.

1 Introduction

For the majority of teaching-focus higher education institutions, institutional missions often emphasizes community services, liberal arts, teaching quality, accessibility, and commitment to diversity [8]. These institutions usually distribute financial and human resources to meet their primary teaching and student service goals across all disciplines. This makes it difficult to support large-scale cyberinfrastructure resources. While there is no previous study regarding

the availability of computing resources at smaller institutions, anecdotal evidence points toward a clear lack of local resources for educational purposes [4]. Even though there are large-scale national infrastructures such as XSEDE, existing utilization from non-research institutions on these resources is low and grows at a rate much smaller than that of research institutions. Furthermore, high utilization rate from prioritized activities leads to increased wait time, particularly during the day. This prevents effective implementation of in-class hands-on learning activities.

Efforts have been made to develop affordable computing clusters that can be used to teach basic parallel and distributed computing (PDC) concepts. One approach is to boot a temporary networked computer lab into a pre-configured distributed computing environment [3]. Combined with a small-scale multiprocessor build-out hardware kit, we have the ability to create inexpensive and portable mini clusters for education [2]. Alternatively, advances in virtualization technologies have led to solutions that support the creation of virtual computing clusters within existing computer laboratories. These clusters can either scale across all resources [10] or are comprised of many mini clusters for learning at individual levels [7]. Without leveraging existing on-premise resources, reduction in hardware costs leads to approaches that lean toward the development of personal computing clusters. The costs can range from around $3,000 [1] to $200 [11]. In both scenarios, they also require additional administrative effort, which could either be facilitated by student teams, supported by institutional staff or require time effort from the instructors. This presents challenges to institutions with limited resources, teaching responsibilities are high, and typical students are not prepared to take up advanced Linux system administration tasks.

We present an approach that leverages cloud computing to provision a cyberinfrastructure (CI) on which a full-fledged virtual supercomputer can be designed and deployed. While conceptually similar to [7], our work does not rely on premade virtual machine (VM) component images. Instead, we utilize an academic cloud [9] to create *blueprints* that correspond to components of a supercomputer infrastructure. At the cost of longer startup time, this allows us to provide a high level of customization to clusters deployed through the platform. The individual tasks in our CI blueprint are carried out as direct automated Linux instructions which helps providing more insights into how the system works. In additional to larger deployment supporting entire classes, the blueprint also allows smaller cluster (taking only a portion of a physical node) to be deployed should students wish to study on their own.

The remainder of this paper is organized as follows. Section 2 describes the design and deployment environments of the proposed cloud-based CI. Section 3 presents and summarizes various administrative and performance scaling tests

regarding operations of the cloud-based CIs. Section 4 concludes the paper and discusses future work.

2 Design and Deployment

We design and deploy a CI that supports the following components: Single-Sign-On (SSO), scheduler, networked file system, parallel file system, and computing nodes. The selected cloud environment is CloudLab, a federally funded academic cloud infrastructure.

2.1 CloudLab

Funded by the National Science Foundation in 2014, CloudLab has been built to provide researchers with a robust cloud-based environment for next generation computing research [9]. As of Fall 2019, CloudLab boasts an impressive collection of hardware. At the Utah site, there are a total of 785 nodes, including 315 with ARMv8, 270 with Intel Xeon-D, and 200 with Intel Broadwell. The compute nodes at Wisconsin include 270 Intel Haswell nodes with memory ranging between 120GB and 160GB and 260 Intel Skylake nodes with memory ranging between 128GB and 192GB. At Clemson University, there are 100 nodes running Intel Ivy Bridges, 88 nodes running Intel Haswell, and 72 nodes running Intel Skylake. All of Clemson's compute nodes have large memory (between 256GB and 384GB), and there are also two additional storage-intensive nodes that have a total of 270TB of storage available.

In order to provision resources using CloudLab, a researcher needs to describe the necessary computers, network topologies, and startup commands in a resource description document. CloudLab provides a browser-based graphical interface that allows users to visually design this document through drag-and-drop actions. For large and complex profiles, this document can be automatically generated via Python in a programmatic manner. Listing 1 describes a Python script that will generate a resource description document that requests six virtual machines, each of which has two cores, 4GB of RAM, and runs CentOS 7. Their IP addresses ranges from 192.168.1.1 through 192.168.1.6.

Listing 1: A CloudLab profile written in Python to describe a 6-node cluster

```
import geni.portal as portal
import geni.rspec.pg as pg
import geni.rspec.igext as IG

pc = portal.Context()
request = pc.makeRequestRSpec()
link = request.LAN("lan")
for i in range(6):
```

```
  if i == 0:
    node = request.XenVM("head")
    node.routable_control_ip = "true"
  elif i == 1:
    node = request.XenVM("metadata")
  elif i == 2:
    node = request.XenVM("storage")
  else:
    node = request.XenVM("compute-" + str(i))
    node.cores = 2
    node.ram = 4096

  node.disk_image="urn:publicid:IDN+emulab.net+image+emulab-ops:CENTOS7-64-STD"
  iface = node.addInterface("if" + str(i-3))
  iface.component_id = "eth1"
  iface.addAddress(pg.IPv4Address("192.168.1." + str(i + 1), "255.255.255.0"))
  link.addInterface(iface)
pc.printRequestRSpec(request)
```

The resource description document instructs CloudLab to provision and instantiate the experiments. Once all resources are allocated and images for the computing components are booted on top of bare metal infrastructure, users are granted complete administrative privilege over the provisioned infrastructure. CloudLab also supports a direct integration between publicly readable git repositories and its profile storage infrastructure. This minimizes the effort needed to modify an existing profile while still maintaining a complete history of previous changes.

2.2 Virtual Supercomputer Blueprint and Deployment

The default blueprint allows users to customize the capacity of their virtual supercomputer. They can decide the number of compute nodes, the number of parallel file system nodes, and the number of CPUs and size of memory per node. While it is possible to request complete physical nodes, we decide to initially deploy all nodes as virtual machines. Only the head node has a public IP address, which supports an SSH connection. Configuration and installation tasks are automated and grouped into individual files. Each file is responsible for the deployment of one service. These services include LDAP directory services for passwordless SSO (LDAP server/client), a distributed file system running NFS (NFS server/client), a parallel file system (BeeGFS management, meta, and storage servers/client) [5], a scheduler (Slurm server/client) [6], and a parallel programming library (OpenMPI). Figure 1 illustrates a dependency graph that represents the workflow among providers and consumers of the deployed services.

These services are split across several servers both to spread the workload and to emulate an environment similar to those seen in large-scale supercom-

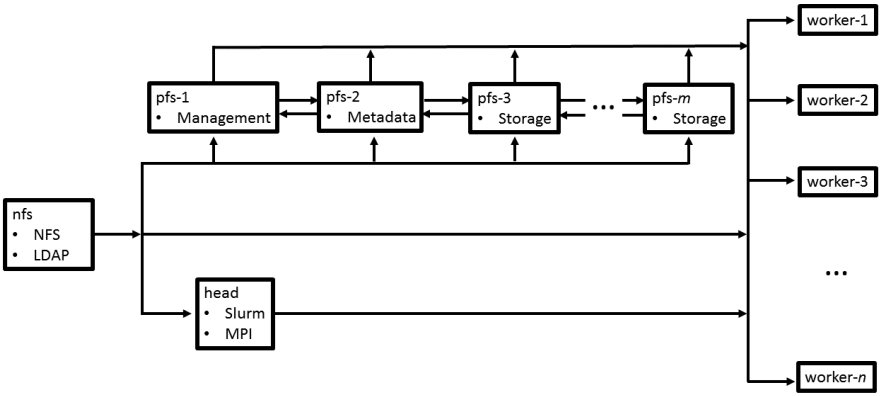


Figure 1: Services provided by each server component

puters. To reduce the footprint of cloud resources, the current blueprint groups related services on the same server. It is possible to modify the blueprint to adjust this service placement. The number of nodes for the computing component and for the parallel file system components can be set at deployment time.

LDAP and Single-Sign-On: Our CI forwards the port of the SSH server on the head node to allow users to sign in remotely. LDAP provides user accounts with uniform UIDs and GIDs across all nodes in the CI. All nodes in the CI authenticate accounts against LDAP, enabling a streamlined environment for all tasks, including passwordless SSH connections between nodes in the server and shared network storage. The automatic deployment of LDAP is facilitated through the use of Debian preseeding and configuration scripts. A pre-configured list of users and groups is included to allow repeated deployment and can be modified. LDAP is the first component to be deployed before any other service.

Filesystems: By default, each instance on CloudLab comes with a total storage space of 16GB. It is possible to attach storage blocks to each instance to serve as expanded local scratch for the compute nodes. Two remote network storage infrastructures are included with the CI blueprint. The first is a network file system (NFS) that is setup on its own node. The NFS filesystem provides four directories to be mounted across the compute and head nodes: */home*: provides shared user home directories, */software*: provides

shared client sofware, */opt*: provides shared system software, and */mpishare*: contains MPI sample codes for students. Housing home directories on the NFS server provides uniform access to user files across the entire CI and allows the user to easily run MPI scripts on all compute nodes. It also makes password-less sign-on between nodes simpler, as each user will have a one SSH key in the shared home directory. The second remote network storage infrastructure is BeeGFS, a parallel file system. BeeGFS consists of three primary services: management, metadata, and storage. In the current default blueprint, we provision one node for the management service, one node for the metadata service, and two nodes for storage. Before deploying the CI, it is possible to customize the blueprint to include more storage servers (improving parallel I/O). It is also possible to merge all three services onto one or two nodes, at the cost of performance. The BeeGFS service configuration is stored in a simple JSON file which loads from the GitHub repository, allowing the user to have the same PFS architecture each time the CI is deployed but also allowing the architecture to be quickly updated before deployment. Once deployed, the BeeGFS storage is mounted as */scratch*. Under this directory, each individual user account in LDAP automatically has a subdirectory, named after the user's login name. These user scratch directories are available across head and all compute nodes.

Scheduler: We include SLURM as our default scheduler in the blueprint, as it is the most popular scheduler across XSEDE sites. The NFS filesystem enables distribution of configuration files for various SLURM components across all nodes, including those of MariaDB, the back end database for SLURM. To automate the configuration process, we created boilerplate configuration files containing dummy information and updated these at deployment time.

Application (OpenMPI): Once SLURM has been installed and configured, OpenMPI can be deployed. Because the virtual supercomputer is deployed in a cloud environment and is also attached to the Internet for user access via SSH, there are several network interfaces on each node. For OpenMPI to run properly, it must be configured at install time to use only the internal IP network. This is achieved through MPI's configuration files, which are loaded with the names of network interfaces provisioned at deployment.

Compute Nodes: These run the clients for LDAP, NFS, BeeGFS, SLURM, and OpenMPI, allowing them to obtain single sign-on, share directories, mount scratch space, and to participate in the parallel computing process. The */home*, */opt*, */software*, and */mpishare* directories are mounted on each compute node on the NFS server, and all compute nodes are provided scratch space on a BeeGFS parallel file system.

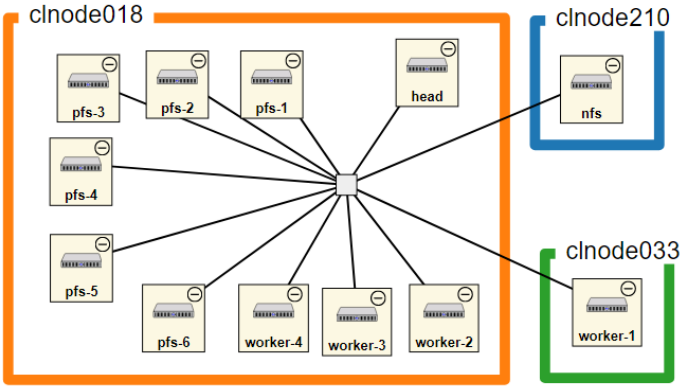


Figure 2: Example CloudLab deployment: Three physical computing nodes.

2.2.1 Deployment

The hardware for our CI is requested from CloudLab using CloudLab's Python scripting support. The same Python script is used to launch accompanying Bash scripts that deploy and configure CI software components. The Python and Bash scripts are stored in a GitHub repository and updates are automatically pulled by CloudLab, allowing the latest version of the CI to be incorporated into the corresponding CloudLab profile with each run.

The entire system deploys and configures itself automatically over the course of several hours, depending on the number of nodes requested. The user who deploys the system can place sample scripts, information, or data in the */source* directory on GitHub, and those files will be automatically copied into the shared scratch directory accessible by all compute nodes. Figure 2 illustrates a sample deployment of a supercomputer with one head node, one NFS node, six BeeGFS nodes (one metadata, one management and four storage nodes), and four compute nodes. The nodes in this deployment are spread across three physical systems hosted at CloudLab's Clemson site. A slightly smaller (in term of node count) deployment hosted at CloudLab's Wisconsin site is shown in Figure 3. This deployment only has four BeeGFS nodes (one metadata, one management and two storage nodes) instead of six.

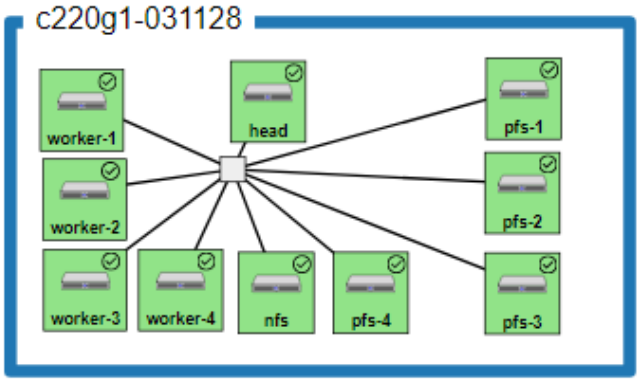


Figure 3: Example CloudLab deployment: A single computing node.

3 Operational and Scaling Tests

The tests described in this section aim to demonstrate that operations of the deployed CI are similar to those of an actual supercomputing infrastructure. Furthermore, scaling behaviors should reflect real world environment. That is, as we increase the number of computing cores, performance improvements can be observed.

3.1 Operational Tests

With LDAP, users are able to SSH directly into the deployed CI via the head node without having to have a CloudLab account. This enables adhoc training scenarios where only temporary accounts are needed. Once users logged into the head node, interactions with the scheduler such as viewing the computing nodes' status (*sinfo*), submitting jobs (*sbatch*), and watching the queue (*squeue*) can be done as usual. This is demonstrated in Figure 4.

3.2 Scaling Behavior Tests

To evaluate scaling behavior, two tests are carried out. The first is an MPI program that estimates integral of x^2 using trapezoids. A SLURM submission script runs and times the MPI program using increasing numbers of processes. Calculated speedups are shown in Figure 5. With 32 total CPUs available (8 core per worker node), near linear speedups are observed as the number

```
merino@head:~/source/openmpi$ sbatch mpiio_bigwrite.pbs
Submitted batch job 2
merino@head:~/source/openmpi$ sbatch static.pbs
Submitted batch job 3
merino@head:~/source/openmpi$ sinfo
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
edge*        up   infinite      4  alloc worker-[1-4]
merino@head:~/source/openmpi$ squeue
             JOBID PARTITION     NAME     USER ST       TIME  NODES NO
DELIST(REASON)
                 3      edge      mpi   merino PD       0:00      4 (R
esources)
                 2      edge      mpi   merino  R       0:11      4 wo
rker-[1-4]
merino@head:~/source/openmpi$
```

Figure 4: Standard interaction with the scheduler for job submissions

of processes increase from 1 to 2 and then to 4. With 8 and 16 processes, speedups are still observed but with a much reduced rate. At 32 processes, speedup begins to stagnate and perceivably, decrease.

The second test measures the influence of parallel I/O via BeeGFS. This is done by using MPIIO routines to write 512MB of random data to a file. Speedups are measured as the number of processes increases from 1 to 32. and visualized in Figure 6. The figure indicates that speedup remains unchanged for 1, 2, 4, and 8 processes. At 16 and 32 processes, the writing speed is doubled.

The behavior shown in Figure 6 can be explained as follows. First, the BeeGFS is setup with only two storage nodes, therefore, the maximum possible speedup for read/write access is two. Default MPI settings prioritize placements of tasks on CPUs belong to the same compute nodes. With up to 8 processes running on the same virtual computing node, writing activities are limited by a single virtual network connection and therefore unable to take advantage of the two BeeGFS storage servers. With more than 8 processes, speedups are observed, but capped at 2 due to similar logic from the storage servers' perspective.

4 Conclusion and Future Work

It has been demonstrated that from an operational perspective, the CI can provide learners with a working environment similar to that of a supercomputer. It enables organizations to become more proactive in CI training for users and professionals without incurring additional risks in term of performance reduction or system errors due to these educational activities. The materialization

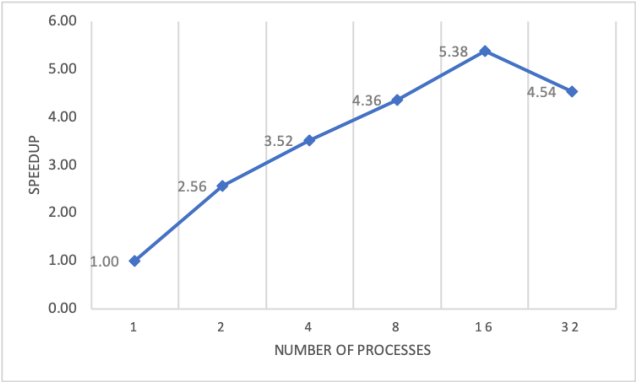


Figure 5: Speedup as number of processes increase: Compute-intensive tasks

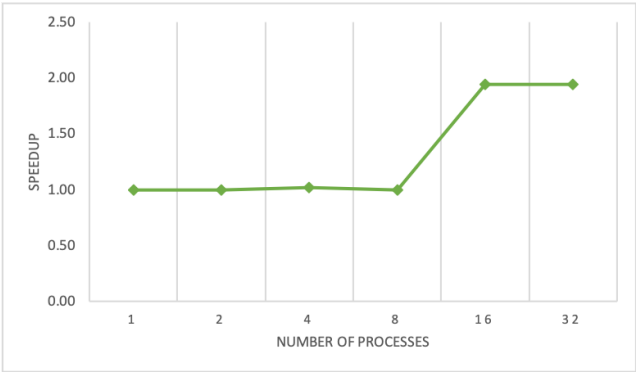


Figure 6: Speedup as number of processes increase: Write-intensive tasks

of a dynamic mini "supercomputer" in the cloud demonstrated in this work provides the motivation for a number of future projects. One such example is the development of additional blueprints to enable site-specific educational materials. Another example is a project that explores whether it is possible to fit a toy cluster on a personal computing device. While VM-based solutions exist, for a multi-node supercomputer emulator, the performance overhead on storage, memory, and CPU can be significant on a laptop. We will explore how various components of the supercomputer emulator can be packed into containers. This helps to understand the minimal amount of CPU and RAM needed to run an emulated cluster and whether scaling behaviors can be observed. All source codes developed for this work are made available on GitHub [`https://github.com/WCU-EDGE/MiniSup`].

References

[1] Joel C Adams and Tim H Brom. Microwulf: a beowulf cluster for every desk. *ACM SIGCSE Bulletin*, 40(1):121–125, 2008.

[2] Ivan Babic, Aaron Weeden, Mobeen Ludin, Skylar Thompson, Charles Peck, Kristin Muterspaw, Andrew Fitz Gibbon, Jennifer Houchins, and Tom Murphy. Littlefe and bccd as a successful on-ramp to hpc. In *Proceedings of the 2014 Annual Conference on Extreme Science and Engineering Discovery Environment*, page 73. ACM, 2014.

[3] Sarah M Diesburg, Paul A Gray, and David Joiner. High performance computing environments without the fuss: the bootable cluster cd. In *19th IEEE International Parallel and Distributed Processing Symposium*, pages 8–pp. IEEE, 2005.

[4] Jeremy Fischer, Steven Tuecke, Ian Foster, and Craig A Stewart. Jetstream: a distributed cloud infrastructure for underresourced higher education communities. In *Proceedings of the 1st Workshop on The Science of Cyberinfrastructure: Research, Experience, Applications and Models*, pages 53–61. ACM, 2015.

[5] Jan Heichler. An introduction to beegfs, 2014.

[6] Don Lipari. The slurm scheduler design. In *SLURM User Group Meeting, Oct. 9*, volume 52, page 52, 2012.

[7] Paul Marshall, Michael Oberg, Nathan Rini, Theron Voran, and Matthew Woitaszek. Virtual clusters for hands-on linux cluster construction education. In *Proc. of the 11th LCI International Conference on High-Performance Clustered Computing*, 2010.

[8] Christopher C Morphew and Matthew Hartley. Mission statements: A thematic analysis of rhetoric across institutional type. *The Journal of Higher Education*, 77(3):456–471, 2006.

[9] Robert Ricci, Eric Eide, and CloudLab Team. Introducing cloudlab: Scientific infrastructure for advancing cloud architectures and applications. *; login:: the magazine of USENIX & SAGE*, 39(6):36–38, 2014.

[10] Elizabeth Shoop, Richard Brown, Eric Biggers, Malcolm Kane, Devry Lin, and Maura Warner. Virtual clusters for parallel and distributed education. In *Proceedings of the 43rd ACM technical symposium on Computer Science Education*, pages 517–522. ACM, 2012.

[11] David Toth. A portable cluster for each student. In *2014 IEEE International Parallel & Distributed Processing Symposium Workshops*, pages 1130–1134. IEEE, 2014.

Internet Research Agency's Campaign to Influence the U.S. 2016 Elections: Assessing Linguistic Profiles via Statistical Analysis*

YuLin Bingle, William Burke, Micheline Al Harrack
Larry Blankenship, Nguyen Khoanam, Christopher Sokol
Sara Sadat Tabatabaei
Marymount University
Arlington, VA 22207
ybingle@marymount.edu

Abstract

We document the linguistic structure of Russia's Internet Research Agency's social media and disinformation campaign to influence the 2016 U.S. Elections. Using the Discover Linguistic Inquiry and Word Count 2015 computerized text analysis tool, we researched the linguistic profiles of the Clemson University-collected Internet Research Agency tweets and retweets on a word-by-word basis. In our research, we selected a 95% confidence level model and a word count ratio analysis of a 99% confidence level. Our analysis indicate the Internet Research Agency executed a persistent and synchronized strategy during the periods of pre-election, year of election, and post-election. We offer that our study will show: a) policy leaders an example of a sophisticated adversary's social media disinformation campaign; b) cybersecurity planners and defenders the strategy and tactics used in this campaign and thereby develop mitigation plans and actions; and c) researchers the future opportunities for study and analysis.

1 Background

U.S. Federal Government leaders to include the U.S. Intelligence Community have publicly asserted that Russia sought to sway the 2016 U.S. Elections through a hacking and influence campaign [17]. A significant component of the Russian campaign involved the Russian Internet Research Agency (IRA) based in St. Petersburg, Russia.

The 2018-2019 Special Counsel Robert Mueller Investigation concluded the IRA conducted the earliest Russian interference operations, a social media campaign designed to provoke and magnify political and social friction in the United States [9]. In July 2018, the U.S. Justice Department indicted 13 Russian nationals for interference with the 2016 U.S. Presidential election [15]. The indictment named the IRA as central to a Russian effort, beginning in at least March 2014 and largely through social media, to sow discord in the U.S. political system. The indictment outlined evidence of IRA-linked accounts on Facebook, Twitter, Instagram, and Google. The IRA used these accounts to pose as Americans and use their social media messages to try to amplify divisive issues among voters [9].

The purpose of this study is not to establish whether the Russian-affiliated IRA conducted interference operations to influence the 2016 U.S. Elections as was established by the U.S. Intelligence Community and Federal Government heads. Further, the study will not assess the specific impact of IRA disinformation on the U.S. elections as we do not have access to the dataset to support or evaluate the measures of success or failure of IRA actions against its American target audience. Rather, we intend to understand the linguistic profiles the IRA actors used in crafting social media messages on Twitter to achieve their goals.

With access to Clemson University's dataset of the IRA's nearly three million tweets and retweets between February 2012 to May 2018 [12], we ask the central question of what was the sophisticated and coordinated disinformation strategy the IRA used to further exacerbate and intensify the social divide in American politics? From the central focus, we refined two research questions (RQ) about the IRA's tactics on the Twitter social media platform:

RQ1. What is the difference in the linguistic profile used by the IRA actors in the years prior to the 2016 U.S. Elections, during the year of election, and the years after the 2016 election?

RQ2. What is the difference in the linguistic profile in social and political topics addressed by the IRA actors in the years prior to the 2016 U.S. Elections, during the year of election, and the years after the 2016 election?

2 Literature Review

The authors found two categories of literature that underpin the foundation of this study. The first represented results of U.S. Government investigations underpinned by assessment from the U.S. Intelligence Community that concluded that Russia, at the direction of the Kremlin, used social media in an attempt to influence the 2016 U.S. presidential election [9, 16, 17]. These sources provided the formal and conclusive findings of the U.S. Government.

The second category of literature consists of research papers that explained how words can influence emotions and the decisions people make. Earlier works analyzing the sentiment of words used by influential users and opinion leaders on the Twitter platform led to better understanding of Twitter users' ability to persuade their target audience [1]. More recently, studies correlated Twitter sentiment analysis with Google search query trends and concluded that social media users can use the correlation to tailor their messages to influence audiences in specific geographic regions [5]. Researchers continue to assess this hypothesis to include work on the methods to classify Twitter messages and other social media messaging. One recent study classified 1.5 million Twitter messages from 13 languages (Albanian, Bulgarian, English, German, Hungarian, Polish, Portuguese, Russian, Serbo-Croatian, Slovak, Slovenian, Spanish, and Swedish) resulting in 138 sentiment models [8].

From literature research and analysis, we caution that a single social media message will not provide the granularity from which to derive sentiment analysis. Instead, aspect level sentiment analysis is more granular than sentence analysis and even more so with document analysis. Zainuddin, Selamat, and Ibrahim in their 2018 study [21] demonstrated the ability to substantially "improve the accuracy performance from the existing baseline sentiment classification methods by 76.55, 71.62 and 74.24%, respectively" by using aspect-based feature extraction and aspect-based sentiment analysis. Their aspect-based analysis stemmed from principal component analedysis (PCA), latent semantic analysis (LSA), and random projection (RP) feature selection methods.

Other researchers have advanced aspect-based analysis of the written word with pioneering research that assessed emoji data as part of sentiment analysis models [6]. Yang, Macdonald, and Ounis [19] combined sentiment analysis with the ability to identify election Twitter messages to account for situations when specific words are not accounted for in a linguistic analytic model. Budiharto and Meiliana [2] also conducted research by combining sentiment analysis from Twitter data with computerized text analytic methods to develop a framework they used to predict Indonesian Election results in 2019. Relatedly, Budiharto and Meiliana have used the framework to predict election results in the UK, Spain, and France, and shared their algorithm for future researchers to test and refine. Researchers have accepted the challenge by attempting to classify,

organize, and quantify Twitter sentiment in real time, tackling the big data challenges, and incorporating machine learning to assess the sentiment in the social media distributed environment [3].

3 Methodology

We employed an inductive, mixed methods research design [4], by combining both qualitative and quantitative analysis. Using the Clemson University-collected IRA tweets [7], we first applied qualitative analysis to parse the data that will be relevant to the research questions. Then using the Discover Linguistic Inquiry and Word Count 2015 (Discover LIWC2015) computerized text analysis [10] to research the linguistic profiles of IRA tweets and retweets on a word-by-word basis. Completing the LIWC2015 quantitative analysis, we then used the linguistic profiles results to inductively analyze the data to understand better the two research questions posed for this study.

We selected the LIWC sentiment analysis tool for this study because of its reputation and credibility in correlating linguistics and sentiment. Still, we decided it important to understand the capabilities of the LIWC tool and to compare it to other available free and fee-based tools accepted as scholarly tools for linguistics analysis. In their 2012 study [20], Young and Soroka compared eight commonly used scholarly sentiment analysis tools and scored the LIWC with the highest correlation against the Lexicoder Sentiment Dictionary.

Another group of researchers [11] in 2016 evaluated and benchmarked 24 different sentiment analysis tools using 18 "gold standard" labeled datasets[11] from multiple sources. They ranked LIWC as the number two performer for the 3-class sentiment (positive, neutral, and negative) test and number five performer for the 2-class sentiment (positive and negative) test from among the 24 total analysis tools benchmarked.

In addition to receiving positive ratings for its analysis tools, LIWC also gained endorsement earlier this year for its language translation capability. With United Nations (UN) translated documents as baseline, Windsor, Cupit, and Windsor [18] used the LIWC tool to translate portions of original untranslated UN documents and then compared the LIWC translation against the UN translated versions. The researchers' findings showed the LIWC sentiment analysis tool noted differences between the human versus machine translated text and evaluated effects of both translations on the reader, confirming only very small effect differences [18].

3.1 Dataset Description and Data

We used a public dataset comprising of tweets from the IRA [13]. As outlined in Figure 1, Filtering Process of the Internet Research Agency's February 2012

to May 2018 Twitter Messages, we filtered and refined the 2,973,371 tweets from 2,848 Twitter handles against the specific dates the IRA actors posted their Twitter messages. Because the timeline of the entire IRA data set is displayed in Greenwich Mean Time (GMT) and this study required relevant data points, we considered closing times of voting stations of individual U.S. states because tweets after poll closings would no longer influence election results. Hawaii is the western most state, so its polls was the last to close for the 2016 U.S. Election. Hawaii local evening time on 8 November 2016 is GMT the following day, 9 November 2016, thus the latter date is the cutoff for IRA Twitter messaging to influence the U.S. Election results.

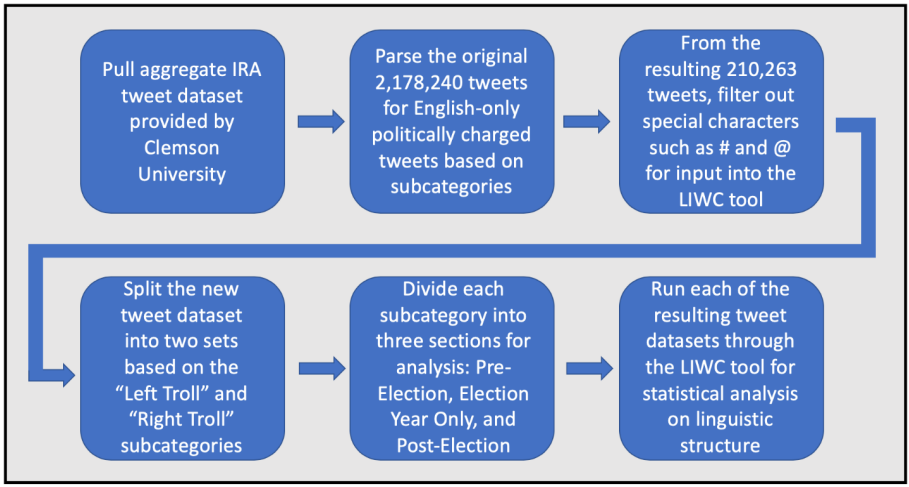


Figure 1: Filtering Process of the Internet Research Agency's February 2012 to May 2018 Twitter Messages

Clemson University professors, Linville and Warren [7], through their research developed a taxonomy on the IRA trolling that separated the dataset into five distinct categories of Right Troll (RT), Left Troll (LT), News Feed, Hashtag Gamer, and Fearmonger (Results section). For this study, we focused on the RT and LT taxonomies of the IRA dataset as they are relevant to our research questions.

Linville and Warren found the RT handles tweeted nationalist and right-leaning populist messages, consistently supported the Republican presidential candidate, and routinely disparaged the Democratic Party [7]. Remarkably, the RT handles rarely messaged traditional Republican themes and often spread divisive messages about mainstream and moderate Republicans [7]. In contrast, the LT handles broadcasted socially liberal messages with significant focus on

cultural identify such as gender, sexual, religious, and especially racial identify [7]. Yet, just as the RT handles criticized mainstream Republican politicians, the LT handles similarly criticized mainstream Democratic politicians, especially the Democratic presidential candidate [7].

Linville and Warren's findings support the insights from our study's literature review, principally that the Russia-affiliated IRA conducted a social media campaign to provoke and magnify political and social friction to influence the 2016 U.S. Elections. Further, the Clemson researchers showed the IRA RT handles supported the Republican presidential candidate, but the LT handles disparaged the Democratic presidential candidate.

3.2 Data Processing

Referencing Figure 1, we downloaded the Clemson University aggregated IRA Twitter dataset, filtered 2,178,240 Twitter messages from the original nearly three million tweets, and parsed 210,263 English language, politically-charged tweets.

To validate the ground truth of the IRA Twitter dataset, we randomly selected 100 handles or authors of the English-language tweets to validate their authenticity. Using Recorded Future (RF) database and analysis, we found ninety-nine of the 100 handles and their Twitter messages in the RF database and therefore assess the IRA dataset as authentic. RF is an expert cyber threat intelligence platform. The RF curated information confirmed the St. Petersburg-based IRA is a Russian troll factory and was intending to divide Americans and disrupt and interfere with the U.S. Election process. Twenty-seven of the 100 IRA handles we tested have 1000 or more references in the RF database and are connected to Russian troll tweets.

After validating the authenticity of the IRA dataset, we filtered out special characters to allow more effective LIWC tool analysis. With the remaining 210,263 English tweets sans special characters, we separated the dataset into RT and LT taxonomies and further divided the RT and LT tweets into three subcategories of Pre-Election, Post-Election, and Election year only. We then ran these final subcategories of IRA Twitter datasets through the LIWC tool for linguistic structure statistical analysis.

4 Results

Figure 2, Internet Research Agency Dataset Variables and Definitions, outlines the variables we used to assess the linguistic structure statistical analysis in the LIWC tool. Figure 3, Linguistic Structure Statistical Analysis of Internet Research Agency Right Troll Taxonomy, and Figure 4, Linguistic Structure

Statistical Analysis of Internet Research Agency Left Troll Taxonomy, depicts the result of our linguistic structure statistical analysis using the LIWC tool.

IRA Dataset Variables and Definitions	
REYOT	Right Troll Category: Election Year Only Timeframe (01 Jan 2016 - 09 Nov 2016)
RPost	Right Troll Category: Full Post Election period (10 Nov 2016 - 22 March 2018)
RPre	Right Troll Category: Full Pre-Election period (25 Nov 2014 - 09 Nov 2016)
LEYOT	Left Troll Category: Election Year Only Timeframe (01 Jan 2016 - 09 Nov 2016)
LPost	Left Troll Category: Full Post Election period (10 Nov 2016 - 30 May 2018)
LPre	Left Troll Category: Full Pre-Election period (06 Feb 2012 - 09 Nov 2016)

Figure 2: Internet Research Agency Dataset Variables and Definitions

After running the filtered data sets using LIWC 2015, the year of election only tweets, pre-election tweets, and post-election tweets, we also ran additional sets filtered by RT and LT for the corresponding time periods. The word count for the pre-election period is nearly equivalent to the post-election period with a ratio of 1.036 resulting in an effective relative increase in the operational consistency. The word count ratio of the right troll post-election to the right troll pre-election showed a significant increase of 1.29; while the left troll post-election showed a ratio increase of 2.34 compared to the pre-election benchmark.

ANOVA

	df	*SS*	*MS*	*F*	*Significance F*
Regression	**3**	**3.21E+12**	**1.07E+12**	**1.14E+11**	**6.11471E-73**
Residual	**14**	**131.771**	**9.412216**		
Total	**17**	**3.21E+12**			

	Coefficients	*Standard Error*	*t Stat*	*P-value*	*Lower 95%*	*Upper 95%*	*Lower 95.0%*	*Upper 95.0%*
Intercept	**-0.95076**	**0.990827**	**-0.95956**	**0.353556**	**-3.075872047**	**1.174355**	**-3.07587**	**1.174354537**
REYOT	**-2.67727**	**0.058603**	**-45.6851**	**1.22E-16**	**-2.802956023**	**-2.55158**	**-2.80296**	**-2.551575793**
RPost	**0.92399**	**0.070288**	**13.14583**	**2.88E-09**	**0.773237938**	**1.074742**	**0.773238**	**1.074742155**
RPre	**2.8227**	**0.105867**	**26.66271**	**2.12E-13**	**2.595638108**	**3.049762**	**2.595638**	**3.049762143**

Figure 3: Linguistic Structure Statistical Analysis of Internet Research Agency Right Troll Taxonomy

The ratio of words charged with positive emotions pre-election was 1.27 compared to the post-election as a whole, while the LT was balanced with almost a consistent ratio of 0.99 for pre-election positive words compared to

ANOVA

	df	*SS*	*MS*	*F*	*Significance F*
Regression	**3**	**3.214E+12**	**1.07143E+12**	**8587024841**	**4.4E-65**
Residual	**14**	**1746.816**	**124.7725721**		
Total	**17**	**3.214E+12**			

	Coefficients	*Standard Error*	*t Stat*	*P-value*	*Lower 95%*	*Upper 95%*	*Lower 95.0%*	*Upper 95.0%*
Intercept	**-5.44322**	**3.4867667**	**-1.561109173**	**0.1408133**	**-12.9216**	**2.035147**	**-12.9216**	**2.035147304**
LEYOT	**-9.45722**	**0.7772151**	**-12.16808906**	**7.8223E-09**	**-11.1242**	**-7.79026**	**-11.1242**	**-7.79026222**
LPost	**-0.83586**	**0.3765656**	**-2.219700873**	**0.0434633**	**-1.64352**	**-0.02821**	**-1.64352**	**-0.0282101**
LPre	**11.63184**	**0.3983019**	**29.20357163**	**6.0466E-14**	**10.77756**	**12.48611**	**10.77756**	**12.4861101**

Figure 4: Linguistic Structure Statistical Analysis of Internet Research Agency Left Troll Taxonomy

post-election positively charged ones. The RT scored a ratio of 1.40 positively charged words during the pre-election time compared to the post-election period. In contrast, the negatively charged emotion words used during the pre-election period equated to a ratio of 0.76 compared to the post-election period as the RT increase by 31% in its use of negative emotions compared to the pre-election time. The LT followed a reverse pattern with a ratio of negative emotions words was a high of 1.58 compared to the post-election time period. The LT focus was evenly divided on social topics pre- and post-election. The RT had a focus of 1.22 pre-election compared to the post-election as social topics dropped 80%. The RT statistics also showed a dramatic drop in topics focused on cognitive process post-election with a ratio of 1.61 pre-election compared to the post-election. The drop in LT focus on cognitive process topics was less pronounced with a ratio of 1.23 for pre- versus post-election.

A stepwise regression analysis accounted for word counts, analytic, clout, authentic, tone, words per sentence, words longer than six letters, words present in LIWC 215 dictionary, positive emotions category, negative emotions category, social, cognitive process, perceptions, bio, drives, relativity, and informal speech. We selected a model with a 95% confidence level resulting in significance level of 5 percent. Our results were valid on a 99% confidence level and supported the findings from the words count ratio analysis. The significance of the RT was almost a 0% in the pre-election time period and was most pronounced in the year of election with a p value of 1.22E-16.

While analysis of the LT during pre-election, post-election, and year of election showed an approximate 0% significance value for the pre-election and year of election period, there was a 4% significance level for the post-election time period.

When we ran both the RT and LT dataset together as a reflection of reality of the tweets, the RT was the most significant overall during the span of pre-election, year of election, and post-election while the LT post-election had the highest significance. The LT pre-election was equally significant as the RT during the election year and both were slightly more significant than the RT after the election.

These statistics indicate the IRA synchronized RT and LT activities by coordinating both trolls to act simultaneously or separately. The statistical model shows the result was most significant when each troll acted separately and most balanced when both the RT and LT broadcasted in concert.

5 Discussion

We selected a 95% confidence level model and our words count ratio analysis was valid to a 99% confidence level thus making our findings statistically credible. Our analysis show the IRA conducted a persistent operation during pre-election, during the year of election, and post-election to influence the 2016 U.S. Elections. Using Linvill and Warren's RT and LT taxonomies, we tracked linguistic traits from the IRA tweets dataset the researchers provided.

The LIWC analysis of the filtered and parsed IRA Twitter messages show the IRA RT focused more on positively charged words than negatively charged words during the pre-election period. During post-election however, the IRA had an increase of 31% in negatively charged words related to topics aligned to the RT profile. During pre-election, the RT focused its tweets on social topics when compared to cognitive process topics. For post-election, the RT tweets show a dramatic drop in topics focused on cognitive process and social activities.

Our analysis showed that LT tweets dramatically increased during the post-election period. The LT tweets were consistent in focusing on positively charged words prior to election and during the post-election period. We noted the LT used more negatively charged words before the election than post-election.

IRA tweets broadcasting social topics remained at similar levels during pre- and post-election periods. There was a less pronounced change in the sentiments of tweets about the cognitive process topic for pre-election as compared to post-election.

6 Conclusion

This study concluded the Russia-affiliated IRA conducted a social media campaign to provoke and magnify the social and political divisiveness among Americans to influence the 2016 U.S. Elections. With the Clemson University provided IRA Twitter dataset, we used the LIWC analytic tool to understand the sophisticated and coordinated disinformation strategy the IRA used to further exacerbate and intensify the social divide in Americans politics. In our research, we selected a 95% confidence level model and a word count ratio analysis of a 99% confidence level. Our analysis indicate the IRA executed a persistent strategy during the periods of pre-election, year of election, and post-election.

Linvill and Warren's taxonomy [7] to classify the RT and LT Twitter activities and characteristics provided invaluable insight to our methodology development. We created a filtering process that enabled an innovative way to assess linguistic traits using the LIWC statistical analysis tool.

The IRA's RT emphasized using positively charged emotions words before the election. After the 2016 election, the RT increased by 31% its use of highly charged words that align with its RT profile. Our analysis showed LT Twitter activities increased during the post-election period. For both RT and LT, their activities remained consistent with positively charged tweets during the year of election period and consisted of negatively charged words before the election. With the combination of access to the IRA Twitter dataset and the LIWC tool, we seized the opportunity to conduct linguistic structure statistical analysis to gain insight to the IRA social media campaign.

Efforts to prevent future interference and influence from foreign nations' social media campaign is inherently challenging. Variance in capabilities among threat actors and their use of technical means such as virtual private networks and traffic pivoting may mask geographical identifiers. Fingerprinting potential nation-state threat actors' linguistic profiles may provide foundation to mitigate future elections interference. However, as significant variance may exist in linguistic skill among threat actor groups, nuanced fingerprinting may be needed for each threat group.

References

[1] C. Bigonha, T. N. C. Cardoso, M. M. Moro, M. A. Gonçalves, and V. A. F. Almeida. Sentiment-based influence detection on twitter. *The Brazilian Computer Society*, 2011. http://dx.doi.org/10.1007/s13173-011-0051-5.

[2] W. Budiharto and M. Meiliana. Prediction and analysis of indonesia presidential election from twitter using sentiment analysis. *Journal of Big Data*, 5(1):1–10, 2018. http://dx.doi.org.proxymu.wrlc.org/10.1186/s40537-018-0164-1.

[3] C. A. Calderón, F. Mohedano, M. Álvarez, and M. V. Mariño. Distributed supervised sentiment analysis of tweets: Integrating machine learning and streaming analytics for big data challenges in communication and audience research. *Empiria*, 42:113–136, 2019. http://dx.doi.org.proxymu.wrlc.org/10.5944/empiria.42.2019.23254.

[4] J. W. Creswell and J. D. Creswell. *Research design: Qualitative, quantitative, and mixed methods approaches*. SAGE Publications, Inc., Thousand Oaks, CA, 2018.

[5] E. D'Avanzo, G. Pilato, and M. Lytras. Using twitter sentiment and emotions analysis of google trends for decisions making. *Program*, 51(3):322–350, 2017. http://dx.doi.org.proxymu.wrlc.org/10.1108/PROG-02-2016-0015.

[6] M. Li, E. Ch'ng, A. L. C. Yee, and S. See. Multi-class twitter sentiment classification with emojis. *Industrial Management & Data Systems*, 118(9):1804–1820, 2018. http://dx.doi.org.proxymu.wrlc.org/10.1108/IMDS-12-2017-0582.

[7] D. L. Linvill and P. L. Warren (n.d). *Troll Factories: The Internet Research Agency and State-Sponsored Agenda Building*. Clemson University, Columbia, SC, 2018.

[8] I. Mozetič, L. Torgo, V. Cerqueira, and J. Smailović. How to evaluate sentiment classifiers for twitter time-ordered data? *PLoS One*, 13(3), 2018. http://dx.doi.org.proxymu.wrlc.org/10.1371/journal.pone.0194317.

[9] R. S. Mueller III. Report on the investigation into russian interference in the 2016 presidential election: volume I of II. Retrieved from https://www.justice.gov/storage/report_volume1.pdf.

[10] J.W. Pennebaker, R.L. Boyd, K. Jordan, and K. Blackburn. The development and psychometric properties of liwc2015. University of Texas at Austin. Retrieved from https://repositories.lib.utexas.edu/bitstream/handle/2152/31333/LIWC2015_LanguageManual.pdf.

[11] F. N. Ribeiro, M. Araújo, P. Gonçalves, M. André Gonçalves, and F. Benevenuto. Sentibench - a benchmark comparison of state-of-the-practice sentiment analysis methods. *EPJ Data Science*, 5(1):1–29, 2016. http://dx.doi.org.proxymu.wrlc.org/10.1140/epjds/s13688-016-0085-1.

[12] O. Roeder. (2018a, July). Why we're sharing 3 million russian troll tweets. Retrieved from https://fivethirtyeight.com/features/why-were-sharing-3-million-russian-troll-tweets/.

[13] O. Roeder. (2018b, August). We gave you 3 million russian troll tweets. here's what you've found so far. Retrieved from https://fivethirtyeight.com/features/what-you-found-in-3-million-russian-troll-tweets/.

[14] Y. R. Tausczik and J. W. Pennebaker. The psychological meaning of words: LIWC and computerized text analysis methods. *Journal of Language and Social Psychology*, 29(1):24, 2010. http://dx.doi.org.proxymu.wrlc.org/10.1177/0261927X09351676.

[15] U.S. District Court for the District of Columbia. (2018, July 13). Case 1:18-cr-00215-ABJ. Retrieved from https://www.justice.gov/file/1080281/download.

[16] U.S. House Permanent Select Committee on Intelligence. (2018, March 22). Russian Active Measures. Retrieved from https://www.congress.gov/115/crpt/hrpt1110/CRPT-115hrpt1110.pdf.

[17] U.S. Senate Select Committee on Intelligence. (2018, May 8). Russian targeting of election infrastructure during the 2016 election: summary of initial findings and recommendations. Retrieved from https://www.intelligence.senate.gov/publications/russia-inquiry.

[18] L. C. Windsor, J. C. Cupit, and A. J. Windsor. Automated content analysis across six languages. *PLoS One*, 14(11), 2019. http://dx.doi.org.proxymu.wrlc.org/10.1371/journal.pone.0224425.

[19] X. Yang, C. Macdonald, and I. Ounis. Using word embeddings in twitter election classification. *Information Retrieval*, 21(2–3):183–207, 2018. http://dx.doi.org.proxymu.wrlc.org/10.1007/s10791-017-9319-5.

[20] L. Young and S. Soroka. Affective news: The automated coding of sentiment in political texts. *Political Communication*, 29(2):205, 2012. http://dx.doi.org.proxymu.wrlc.org/10.1080/10584609.2012.671234.

[21] N. Zainuddin, A. Selamat, and R. Ibrahim. Hybrid sentiment classification on twitter aspect-based sentiment analysis. *Applied Intelligence*, 48(5):1218–1232, 2018. http://dx.doi.org.proxymu.wrlc.org/10.1007/s10489-017-1098-6.

Teaching Introduction to Programming Languages with a Database Twist*

Suzanne W. Dietrich
School of Mathematical and Natural Sciences
Arizona State University
Phoenix, AZ 85069
dietrich@asu.edu

Abstract

Programming languages are fundamental to the computing discipline. Students need to learn multiple programming paradigms as well as their underlying principles as part of their degree. A programming language course covering the imperative, functional, and logic language paradigms is typically a difficult course for students as they learn new ways of problem solving. This paper reports on the pedagogy and experience of teaching such a course in the context of a computing degree in a college of interdisciplinary arts and sciences with only CS2 (Java) as a prerequisite. The choice of Python, Erlang, and Prolog provides a sequence of languages that incrementally adds new features to be mastered by the students. The pedagogical approach uses a combination of skeletal notes, reading quizzes, practice exercises, and active learning to engage students in the learning process. For each language, a programming assignment supporting depth of learning uses a database approach to create relationships between the data and to answer realistic questions over the data.

1 Introduction

The computing B.S. degree program in our college of interdisciplinary arts and sciences provides an alternative degree to the conventional computer science degree offered in an engineering college. The lower-level courses include

CS1 and CS2 in Java, Introduction to Programming Languages, Data Structures & Algorithms, Discrete Mathematical Structures, Brief Calculus, and Statistics. The upper-level courses emphasize databases, networks, and cybersecurity. There is an experiential learning degree requirement, consisting of internships and/or research. The most recent introduction to the lower-level core courses was the Introduction to Programming Languages course with CS2 as its only prerequisite. Adding this course provided a programming language component that was missing from our degree and enhanced pathways for transfer students from Computer Science at the local community colleges.

The design of the course was based on two main components. The first being the Computing Curricula 2013 [15] and the second was the coverage in the corresponding course in the engineering college of our university. The topics covered for the basic principles and concepts of programming languages were essentially derived from these sources. However, the choice of programming languages to use for the coverage of the imperative, functional, and declarative paradigms should be based on unit-specific objectives.

Our unit within the college offers bachelor degrees in computing, applied mathematics, statistics and the natural sciences. The applied mathematics and statistics students are required to take CS1 and CS2 as part of their degree programs. Some of these students double major or minor in computing. Due to the popularity of Python and its applications within the unit, Python is the imperative language choice. The selected functional language is Erlang, based on its applicability in industry for 24x7 Web applications and app development. Prolog, Programming in Logic, is the canonical choice for a declarative, logic language.

This paper reports on the experience of the resulting design of an Introduction to Programming Languages course that emphasizes connections with databases in programming assignments. Section 2 provides a brief overview of related work. The pedagogical approach used in the course across all three languages is presented in Section 3, along with a semester schedule of topics and assignments. Sections 4, 5, and 6 discuss the resources used for the languages along with the specifics of the programming assignments. Section 7 reflects on the experience.

2 Related Work

A background literature review searching for teaching the programming language paradigms course found many older publications on the topic. One of these [22] provided an excellent overview of known issues with language choice for the course, including: prior experience of the student, the choice of the CS1/CS2 language, the use of pure versus hybrid languages to illustrate each

paradigm, and the use of a universal language, such as Ada, to illustrate all of the paradigms. Another paper provided additional reasons "to better understand the difficulty of designing an effective curriculum" [5] for such a course, such as object confusion, breadth versus depth, effect of scale, indoctrination (tools influence problem-solving approach) and desensitization (work around shortcomings of a tool). Most computer scientists would probably agree that the design of the programming language paradigms course is difficult.

Some additional, more recent papers on the subject emerged. One alternative approach explored a methodology where students who were computer science majors would not be mere users of the programming languages but focus on the concepts as if they would become a programming language designer [16]. Another approach emphasized that "doing is better than seeing", although it is more difficult, by designing a one-semester course where students combined the language paradigms in the development of a compiler for a non-trivial subset of a functional programming language [11]. Another study recognized the difficulty of students even becoming users of a programming language in such a course and introduced short, practice exercises that improved learning outcomes for the topics addressed by the practice and increased the students' confidence [1].

The approach presented in this paper also recognizes the many challenges in designing such a programming language paradigms course. The following section provides an overview of the various pedagogies employed to help students learn how to use the programming paradigms, which also includes the incorporation of shorter, practice exercises.

3 Overview of Pedagogical Approach

Table 1 provides a weekly semester schedule for the programming languages course. The basic principles notes cover grammars, parsing and derivation trees, the compilation process, and compilation versus interpretation. The concepts of programming languages notes detail names, bindings and scope; data types; expressions and assignment statements; statement-level control structures; and subprograms (parameter passing and activation records). The basic principles and concepts of programming languages notes are based on coverage from the following texts: [4], [8], [17], [18]. Each language is introduced in the context of these fundamental concepts. The course employs various pedagogical approaches, including skeletal notes, short programming exercises, reading quizzes, and active learning exercises. An overarching goal of the pedagogy is to engage students in active participation in their learning as well as to reflect on what they have learned.

Skeletal notes is a term that references a set of incomplete lecture notes,

Table 1: Semester Schedule

Week	Topics	Assignments
1	Language Overview in Degree Program Basic Principles	BNF WebQuest
2	Concepts of Programming Languages	
3	Python: Strings, Decisions, Loops	Reading Quizzes Codingbat
4	Python: Tuples, Lists, Sets, Dictionaries	Reading Quizzes
5	Python: Modules, IO, Exceptions	Reading Quizzes Review Exercises
6	Python: Classes and Inheritance	Python Quiz Assign Program
7	Active Learning	BankAccount in class
8	Event-based with TKinter	TKinter in class
9	Midterm Review	MIDTERM
10	Erlang: Tuples, Lists, Strings	Codingbat
11	Erlang: List Processing, Maps, Files, CSV	Assign Program
12	Erlang: Review Exercises	Erlang Quiz
13	Prolog: Overview and Database Connection	Codingbat
14	Prolog: findall and Review Exercises	Assign Program
15	Higher-order programming: All 3 languages	Review for Final

where students are responsible for filling in some of the parts. For each language and for each data structure, the class notes provide an overview of the data structure, its fundamental operations and methods. There are a couple of examples in the notes that the instructor illustrates on the language in its IDE. Each data structure includes a skeletal notes page that contains a table of 6-8 problems over the data structure with space for students to write the expression/code to answer the question and to record the response from the execution of the code. Before the start of the next class, students must complete the notes page and submit their attempt on the course management system for participation points. They are allowed to work with another student in the class and they are asked to document that collaboration. The day that the notes are due, there is a brief class discussion answering any major questions

students may have. The next class, after reviewing the students submissions, common mistakes are discussed and corrected.

Practice is an essential component of learning programming languages. Once the fundamentals of the language are covered, the students are introduced to the codingbat.com site. In class, the inOrderEqual and twoAsOne codingbat examples provide students with illustrative samples. Students are then given an assignment using additional exercises from codingbat.com to explore the language. The examples chosen (countHi, in1to10, alarmClock, repeatSeparator, mixString) provide students with practice on string processing, comparison operators, nested comparisons, and iteration/recursion. The students are given a document that has the description from the codingbat site, and includes space for the student to record the code that they ran on codingbat and what they learned from writing that code. They are also asked to include a screen shot of the test cases that passed. The students are instructed that it is not the answers that are important but the journey of learning. The students submit the document for participation credit.

There is a programming assignment on each language to provide a depth of learning. Each semester a different data set from [21] is selected to provide realistic but cleaned data to use in the assignment. Students write programs in each language to process the data and answer questions over the data. More details appear later in the paper with respect to the programming assignment in each language.

After covering the language and before a quiz, students have an in-class exercise to complete a set of "Review Exercises". These designed exercises have students apply different facets of the languages:

1. letter_count: Return a dictionary/map of the count for each letter appearing in the parameter string.

2. both_strings: Return a string that includes the characters appearing in both parameter strings.

3. repeats_num: Determine whether the sequence of numbers has a repeated number.

4. order_summary: Given a list of tuples representing an item, price, and quantity on an order, return a tuple with the total cost of the order and the total number of items.

Students are broken down into groups of 4-5 and assigned one of these exercises to work out as a team. Then volunteer groups are asked to record their answer on the white board for class discussion. Pictures are taken and posted on the course management system. Students are strongly encouraged to code the review exercises themselves and verify.

For quizzes and exams, students are provided with a one-page, two-sided reference sheet for the syntax of each language. For Erlang and Prolog, this reference sheet also includes examples of code, such as inOrderEqual, twoAsOne, and list processing. When the assessed midterm is returned in class, students fill out a questionnaire, also known as an *exam wrapper* [12], to reflect on their preparation, what they would do differently for the next assessment, and provide feedback to the instructor. These are collected for instructor feedback and returned to students for the final exam.

4 Python

Our CS1 and CS2 courses use the Java programming language. The goal for learning Python is for students to have the same level of knowledge as Java. Due to the diverse backgrounds of students and the amount of transfer students, an interactive Python textbook has been adopted [10] so that students can fill in their knowledge gaps. Table 1 illustrated the topic coverage in Python and that each week, students are assigned reading that covers those chapters and must complete the reading quizzes for participation credit. The course also utilizes some of the *code checks* from the text for participation credit and in-class exercises.

There are several new concepts for those students who have only been exposed to Java, such as functions, tuples, lambda expressions, and list comprehensions. The syntax for a lambda function is introduced in the context of sorting lists of tuples where the sort requires an order that is not the default. List comprehensions are an elegant short-cut syntax for creating lists.

The Python programming assignment reads in a csv file of clean data and processes the data into objects, using concepts from object-oriented database design [6]. In a weather example with stations reporting observations, the students create Station and Observation classes and the list of observations for a station is a list of references to its associated observation object instances. The students are required to create a dictionary of these objects to essentially store the "database extent". Using this data structure, students answer specific questions by coding the navigation of the data and perhaps creating additional data structures in the process. Due to the increased enrollment of the class (now capped at 60), a free replit classroom [14] for the Python language assists students with checking their output with the expected output. The programs are still assessed based on a visual inspection of the code with a detailed rubric because the assignment description provides strict guidelines on the format of the objects and data structures to be incorporated.

5 Erlang

Erlang is a dynamically typed, functional language that has a lexical structure similar to Prolog for atoms, variables, and list processing. Erlang includes maps, tuples, lists, and strings, as well as lambda expressions and list comprehensions. Thus, Erlang is chosen as the next language in the sequence, building on these similar concepts from Python as well as functions. Erlang only supports recursion and not iteration, so the list processing is similar to Prolog with a head and a tail, however, the Erlang rules are functional and utilize pattern matching. The first rule head that matches is chosen, and variables are bound once through the pattern matching. The coverage of Erlang is restricted to sequential Erlang since the only prerequisite is CS2 and not operating systems. Students are directed to both a free online book [9] and the Erlang site [7] for online documentation.

There are many new concepts for students to grasp with Erlang, including its functional, pattern matching approach as well as the use of recursion instead of iteration. The processing of lists using recursion with the head and tail construct is a skill that students must learn to master with Erlang.

The Erlang programming assignment is similar to Python with the same data but a value-based relational database approach stores the relationships between the data as id values as components of tuples, which are stored in maps. In the weather example, there are station and observation tuples that are associated by the station's unique code. To simplify the assignment, students are provided with a template for the program that handles the reading of a csv file, returning a list of strings for each line. They must complete the program to create the maps and answer the questions over the data. Students use an online Erlang replit environment to complete their assignment because an Erlang classroom was not available as of Spring 2020. Therefore, students are given a compiled Erlang file and they must call a function in this module to validate their data structures before submission.

6 Prolog

The fundamentals of Prolog as a logic language are covered, including the details of clauses, resolution, and a refutation proof so students understand that Prolog provides binding to variables when it has found a refutation and backtracking continues searching when asked for the next answer. However, the emphasis on Prolog is quickly switched to using Prolog facts to store data and its declarative rules to compute the answers to database queries. This is a practical approach that students appreciate. Replit does not support Prolog. However, SWI-Prolog has an online interpreter [20]. There is an online resource

for Prolog [2] and our students have online access to [3] through the university library.

The codingbat and review exercises in Prolog are quite similar to that of Erlang with a fundamental difference. While both Erlang and Prolog must pass the information that it needs, Erlang functions return values but Prolog must return values by adding an extra argument to bind the result to a variable. Since the class does not cover extensions to fundamental Prolog, the review exercise question that creates a letter-count map is changed to the creation of a list of letter(Letter, LetterCount) tuples for each letter in the string.

The Prolog programming assignment uses the same set of data, now stored as Prolog facts. For the weather example, the facts are stored as station and observation tuples, with relationships using the value-based ids. The assignment asks students to answer queries over that data in Prolog. There are five queries that typically involve returning a list of tuples sorted in a particular manner. Since asking a database query typically means to find all answers, the Prolog predicate findall(Template, Goal, Bag) is introduced, which returns a list of bindings for the Template satisfying the Goal in the Bag variable. Similar to a list comprehension, the Template represents the term for inclusion in the list where the bindings are satisfied by the Goal.

7 Reflection

The last week of the semester, while students are working on their Prolog assignment, the class begins a reflection phase, looking back on what was learned at the same time that higher-order programming is introduced. Essentially, higher-order programming is where code can be treated as a value, passed into or returned from a block of code. The lambda functions that students used in Python and Erlang as sort functions are examples of higher-order programming. The course notes emphasize the filter and map higher-order functions available in the languages, although in Prolog the predicates are called include and maplist. To show the importance of handling data as streams of information, an overview of Java streams is covered to illustrate the filter and map in Java.

As the class reviews context-free grammars for the comprehensive final, class time is allocated to show students Prolog's support of definite clause grammars. In fact, SWI-Prolog provides a grammar example to run [19]. In addition to this demonstration, an example grammar from earlier in the semester is modified to a definite clause grammar and the parse trees that were hand-drawn at that time are visually represented by Prolog.

Learning three new programming languages is a challenge for most students. They are learning new ways of thinking about problem solving. Erlang and

Prolog's reliance on recursion instead of iteration is particularly challenging for students who have only completed CS2 and have not taken the full Data Structures and Algorithms class. There are a significant number of students who rise to the challenge and embrace the new paradigms. Although anecdotal, the exposure to multiple programming paradigms, and in particular Prolog, has helped students in learning SQL in the required database class. Also, the exposure to a functional language helps students learn LINQ (Language INtegrated Query) [13] in the advanced database course.

References

[1] T. Austin, C. Horstmann, and H. Vu. Explicit short program practice in a programming languages course. *Journal of Computing Sciences in Colleges*, 33(4):114–122, 2018.

[2] P Blackburn, J Bos, and K Striegnitz. Learn prolog now. `http://www.learnprolognow.org/`.

[3] M Bramer. *Logic programming with Prolog*, volume 9. Springer, 2005.

[4] Y Chen and W Tsai. *Introduction to programming languages: Programming in C, C++, Scheme, Prolog, C#, and SOA, 5th ed.* Kendall Hunt Publishing Company, 2017.

[5] W R Cook. High-level problems in teaching undergraduate programming languages. *ACM Sigplan Notices*, 43(11):55–58, 2008.

[6] S W Dietrich and S D Urban. Fundamentals of object databases: Object-oriented and object-relational design. *Synthesis lectures on data management*, 2(1):1–173, 2010.

[7] Erlang. Erlang programming language. `https://www.erlang.org/`.

[8] M Gabbrielli and S Martini. *Programming languages: principles and paradigms*. Springer Science & Business Media, 2010.

[9] F Hebert. Learn you some Erlang for great good! `https://learnyousomeerlang.com/`.

[10] C S Horstmann and R D Necaise. *Python for everyone, 3rd ed.* Wiley Publishing, 2018.

[11] K D Lee. A framework for teaching programming languages. In *Proceedings of the 46th ACM Technical Symposium on Computer Science Education*, pages 162–167, 2015.

[12] M C Lovett. Make exams worth more than the grade: Using exam wrappers to promote metacognition. *Using reflection and metacognition to improve student learning: Across the disciplines, across the academy*, pages 18–52, 2013.

[13] E Meijer. The world according to linq. *Commun. ACM*, 54(10):45–51, October 2011.

[14] Repl.it. Repl.it - the collaborative browser based ide. `https://repl.it/`.

[15] M Sahami, A Danyluk, S Fincher, K Fisher, D Grossman, E Hawthorne, R Katz, R LeBlanc, D Reed, S Roach, et al. Computer science curricula 2013: Curriculum guidelines for undergraduate degree programs in computer science. *Association for Computing Machinery (ACM)-IEEE Computer Society*, 2013.

[16] S Samuel. Teaching programming subjects with emphasis on programming paradigms. In *2014 International Conference on Advances in Education Technology (ICAET-14)*. Atlantis Press, 2014.

[17] M L Scott. *Programming Language Pragmatics, 4th ed.* 20016.

[18] R W Sebesta. *Concepts of programming languages, 11th ed.* Pearson Education, Inc, 2016.

[19] SWI-Prolog. Swish – grammar example. `https://swish.swi-prolog.org/example/grammar.pl`.

[20] SWI-Prolog. Swish – swi-prolog. `https://swish.swi-prolog.org/example/examples.swinb`.

[21] E Tilevich, C A Shaffer, and A C Bart. Creating stimulating, relevant, and manageable introductory computer science projects that utilize real-time, large, web-based datasets. In *Proceedings of the 46th ACM Technical Symposium on Computer Science Education*, pages 711–711, 2015.

[22] J Traxler. Choosing languages to teach programming paradigms and programming styles. *WIT Transactions on Information and Communication Technologies*, 7, 1994.

A Scalable RPG Project for Object-Oriented Software Development*

Robin M. Givens
Computer Science Department
Randolph-Macon College
Ashland, VA 23005
robingivens@rmc.edu

Abstract

This paper describes the development and outcome of a multi-part, object-oriented project based on role-playing games for a second-year software development course. The objective of the project was to provide an experience with a progressive, large-scale project focused on object-oriented design, implementation, and testing. The first phases of the project emphasized object-oriented programming, and in the final phase students controlled all of the design, implementation, and testing process. The role-playing game genre was chosen for its natural correlation to object-oriented design and its popularity among students. We present an example course schedule, the details of the project, the effects of the COVID-19 pandemic, positive student outcomes, and ideas for scaling the project.

1 Introduction

Our software development course is a second-year course that introduces students to a new programming language (Java) and covers object-oriented development and software design principles. A significant goal of the course is to provide a multi-part software development project that incorporates software design principles, object-oriented programming (OOP), and unit testing. The focus of this paper is the introduction of a three-part programming project

based on role-playing games (RPGs) to the software development course. For the first part of the project, students program interacting classes representing weapons, heroes, enemies, and the game map. In the second part, students extend their classes using principles of inheritance. For the final phase, students take complete ownership of the design, implementation, and testing of the remainder of their RPG game. This project was assigned in Spring 2020, and though the course was impacted by the COVID-19 pandemic, students showed enthusiasm for the project and retention of course material.

Game development has been used as a pedagogical tool for courses throughout the computer science curriculum. Instructors have converted early content-specific projects to game-based projects [3], created game projects for all levels of computer science [7, 11], and redesigned an entire curriculum to utilize electives focused on game development [9]. In an effort to engage and retain students, game programming has been seen in first and second semester introductory courses [2, 3, 6, 12], and has also been used to teach artificial intelligence [8], large-team project management [13], and a senior project course [10]. The connection between game design and object-oriented design has been particularly recognized [5, 6, 12, 14].

As a well-established gaming genre, RPGs provide an attractive avenue for student engagement, and the character-ability design of the games allows for natural object-oriented design and implementation. The RPG genre, in particular, provides a way to engage women in computer science as research has found that more than half of PC RPG gamers are female [4].

RPGs grew out of the rise in the popularity of fantasy literature throughout the mid-1900s and made their debut through tabletop games, such as Dungeons and Dragons, and text-based adventure games, such as Colossal Cave Adventure, in the 1970s. RPGs offer players control of a main character in a fantasy setting who must accomplish a series of quests. RPGs tend to have well-defined stories, settings, quests, combat, items, and characters who have properties such as actions, abilities, experience, levels, health, and inventory. Today, computer-based RPGs are as popular as ever, often feature advanced graphics, range from single-player to massively multiplayer online RPGs (MMORPGs), and can offer collaborative, cooperative, or competitive gameplay [1].

Using a phased RPG project to teach programming concepts has been explored previously in introductory computer science courses. An in-class RPG project emphasized object-oriented concepts for a second semester computer science course in [12]. In [6], students worked on an RPG project based on the game Rogue for the final quarter of a 3-quarter introductory programming course. We extend these works by assigning a multi-part RPG project for a second year object-oriented software development course. In our course, students learn a new programming language, build larger pieces of the game during each

phase, and control the entire design, implementation, and testing process for the final phase of the project.

In this paper we describe the placement of the software development course in our computer science major, the topics the course covers, and changes made to the course as a result of the COVID-19 pandemic in Section 2. In Section 3, we describe the three part object-oriented RPG project in detail, including alterations due to COVID-19. Section 4 considers student responses to the object-oriented project, and Section 5 provides ideas for utilizing and expanding the project in the future.

2 Environment

Randolph-Macon College is a small, private, liberal arts college with an established and growing Computer Science Department. The first two semesters of computer science are taught in Python through introduction to computer science (CS1) and data structures (CS2). CS1 explores fundamental principles of computer science including a final section on introductory classes and objects, and CS2 covers the implementation, use, efficiency, and applications of many common data structures. Students must pass CS1 to take CS2 and similarly must pass CS2 to take software development. All three courses have a lab component, are required for the major, and must be passed with a C- to count for the major.

Though students can take software development as early as their sophomore year, the course has many students in their junior or senior years as it is not currently a prerequisite to any required major course. Most students who take the course are learning Java as their second programming language, though a handful of students have already taken a class in C/C++ or had some experience with Java in high school. All students who take software development have experience creating classes in CS1 and CS2, but that experience is limited to the basics (constructors, setters, getters, string representation, and equals) in CS1, data structures in CS2, and only in the Python language. Our software development course introduces Java; OOP concepts such as inheritance, abstract classes, interfaces, and polymorphism; and software design topics including the Eclipse IDE, UML, unit testing, documentation, and GUI design.

Table 1 provides an example course schedule of weekly topics and accompanying labs and projects for the software development course. Labs cover the same topic as the lecture of the week and projects cover material from all preceding weeks. This schedule provides time for students to work on their projects in lab, and in Spring 2020, students were allowed, but not required, to work in pairs on their projects. Labs are given 1 week and projects are given

2-3 weeks for completion, and project release dates are up to two weeks before their scheduled lab time.

For this course, labs are singular assignments unrelated to other labs, though the material may overlap. After the introduction to the Java programming language and the first project, lecture examples and the object-oriented RPG projects cover new material by extending the previous examples and projects. This design gives the students the experience of a large software development project. The development of the three part object-oriented RPG project is the focus of this paper.

Table 1: Example Course Schedule. Randolph-Macon has a 14-week semester, excluding breaks, with longer class sessions (1 hour 3-day courses and 1.5 hour 2-day courses).

Week	Lecture/Exams	Labs/Projects
1	Intro to Java	Lab 1
2	Ifs, Loops, Arrays	Lab 2
3	Files and Exceptions	Lab 3
4	Eclipse, JUnit, Documentation	Lab 4, Java Project
5	Classes, OOP	Lab 5
6	Interacting Classes	Lab 6
7	Review / Exam 1	RPG Project 1
8	Inheritance	Lab 7
9	Abstract Classes, Interfaces, Polymorphism	Lab 8
10	UML, Unit Testing	Lab 9
11	Review / Exam 2	RPG Project 2
12	GUI Design, JavaFX	Lab 10
13	Programming by Contract	RPG Project 3
14	Final Exam	

2.1 Changes Due to COVID-19

In Spring 2020, Randolph-Macon College closed and did not hold classes at the end of week 6 and beginning of week 7, then resumed classes online for the remainder of the semester. This caused significant disruption to the lecture, lab, and exam schedule, but projects were able to proceed with few changes. Exams were rescheduled online, we dropped the programming by contract lecture and the last two labs, and the RPG projects were slightly reduced in scope. More details on the project changes are provided in the following section.

3 Object-Oriented RPG Projects

The software development course provides a full introduction to the Java programming language before delving into software design and development through object-oriented programming. After the Java lectures and a project focusing on procedural Java programming, students work on three object-oriented programming assignments that build on each other. Students are given unit testers using JUnit to test the functionality of their classes for the first two object-oriented assignments, and are expected to use Javadoc for documentation in every assignment.

In the first RPG project, students build simple classes that interact with each other. In the next project, students extend those classes using inheritance. In the final project students choose how to expand their work through further development of their classes, focusing on gameplay, additional classes, an interactive GUI, and/or anything they propose that uses concepts from the course.

3.1 RPG Project 1: Interacting Classes

In this project students create the Weapon, Enemy, Hero, and Map classes for an RPG. The focus of this project is basic class development and functionality and creating and using instances of the classes. The Enemy, Hero, and Weapon classes cover basic class structure (instance variables, nonparameterized and parameterized constructors, setters, getters, string representation, and equals) as well as simple functionality related to the development of an RPG. Each of these objects has a type attribute (to contain descriptions such as "Ogre", "Wizard", and "Bare Hands") and a copy method to provide encapsulation. Enemies and heroes have characteristics such as a name, a weapon, health, and a level and abilities such as using the weapon and increasing/decreasing health. Heroes also have experience points (XP) and the ability to gain XP and subsequently level up. Weapons have a strength and the ability to power up.

The Map provides the foundation to start designing game interaction between heroes, enemies, and weapons. The map is designed as a grid with each location either having or not having an obstacle. The Map can be queried to determine if a location has an obstacle, obstacles can be added or cleared from a location, and a location on the map can get its obstacle by returning a default enemy object. If a request is made for an obstacle at a location that does not have an obstacle, a runtime error is thrown.

Due to the move online with COVID-19 in Spring 2020, students who had relied on lab computers to use Eclipse, and did not have machines powerful enough to run Eclipse, were unable to use JUnit. Similarly designed testers

were provided for these students, but the testers lacked the ability run an individual test if any part of the students code was unfinished. To simplify the development of the testing process, the health attributes and functionality were eliminated from the assignment. Some students, on their own design, added health into RPG Project 3.

3.2 RPG Project 2: Inheritance and Abstract Classes

The focus of this project is on inheritance through the creation of subclasses, abstract classes, and the use of interfaces. Students create a Magic interface, create a runtime error subclass to throw in the Map class, convert the Hero class to an abstract class, extend the Hero class with Wizard, Elf, and Halfling subclasses, and extend the Weapon class with Sword, Staff, and Bow subclasses. The Magic interface has a single method that returns the magical force (as a numerical value) of an object; implementation of this interface also requires that an object keep track of its magical force.

The Sword, Staff, and Bow subclasses of the Weapon class each have unique properties. The Sword class is a simple Weapon subclass. The Staff class implements Magic. The Bow class keeps track of the number of arrows it has and can use and craft an arrow.

The Wizard, Elf, and Halfling subclasses of the Hero class similarly have individual attributes and functionality. The Wizard's weapon is a staff, the Elf's weapon is a bow, and the Halfling's weapon is a sword. The weapon attribute stays in in the Wizard class, giving students practice with polymorphism and object casting when calling the superclass constructor and in new getter methods. The Elf class implements Magic, and the Halfling class has a hunger attribute that increases whenever XP is gained and decreases whenever an instance eats.

In Spring 2020 the scope of this project was reduced by not updating the Hero class after it became an abstract class. Students were not required to add any methods or make any existing methods abstract. Similarly, students were not asked to extend Java's runtime error class with an error specifically designed for the project.

3.3 RPG Project 3: Choose Your Own Adventure

For the third project students choose, design, and implement an expansion of their previous assignments. Students submit their UML design, JUnit tests, and a README file to explain and test their code. Students are given the following examples of directions they could go, and are expected to do work equivalent to at least two options, with emphasis on making a playable game:

- Create three Enemy subclasses that each have unique properties.

- Enhance the Map class.
- Create a game driver that allows the hero to move throughout the map and fight enemies.
- Implement and incorporate a new class or classes representing items such as inventory, companions, or assistants.
- Build a JavaFX applet that lets the user choose their hero type and name.
- Create/choose/design something to enhance the project.

In Spring 2020, this project was scaled back more than the others due to the time constraints of moving online. Students were asked to create UML design for only one new class they implemented, were not required to use JUnit or provide full tests, though they were expected to do tests that isolated functionally, and were only required to do the work equivalent to one option above.

4 Student Outcomes

Students responded well to the RPG project as a whole, and particularly to the final part in which they completed the entire design, implementation, and testing process on their own. The positive response could be seen in their creativity and thoroughness even during a last-minute online semester that saw many schedule changes and extraordinary time and family pressures.

Each project option for the final assignment was attempted by at least one student or pair, except using JavaFX which was not covered until the last week of class. The following is a list of some of the design, implementation, and testing elements students submitted for the final part of the project that showed their excitement and willingness to go above and beyond what was asked of them. Many of these items are related directly to the RPG genre but incorporate important object-oriented and software development concepts from the class.

- Creation of a Health interface for both enemies and heroes.
- Implementation of weapon subclasses to match new enemy subclasses. One of the most entertaining examples of this was a Karen class with weapon Screech and a FloridaMan class with weapon Gator.
- The addition of an Undead interface for new enemy subtypes.
- Full unit testing in JUnit for all new classes.
- Updating the Map class to randomly generate each new Enemy subclass.
- UML diagrams for the entire project.
- A simple, but completely playable, text-based game that appears to be unwinnable.

As part of the final exam, students were asked to provide their biggest takeaway from the course. Three students mentioned the RPG project as part of the learning process and two mentioned working on a larger, progressive project. The other outcomes given by the students (number of students) are inheritance (12), Java (10), Object-oriented programming/design (9), Eclipse (4), interfaces (3), testing/JUnit (3), encapsulation (2), abstraction (2), JavaFX (1), Javadoc/commenting (1), and polymorphism (1). Overall, students seemed to have a clear sense of the learning outcomes of the course.

Anecdotal evidences suggests that the RPG project, and particularly the design to implementation to testing process, had a positive effect on student understanding of objected-oriented programming. Twenty-seven students took the course in Spring 2020. Of these students, 20 scored higher on the final than the first exam with an average increase of 4.1 points among all students, and 19 scored higher on the final than the second exam with an average increase of 6.8 points.

Student performance on the second exam was the most varied. Although all exams were done online, the second exam covered material taught exclusively online, and students likely struggled with the online transition in conjunction with more difficult material: inheritance, abstract classes, interfaces, and polymorphism. Though instruction continued online, the final assignment could be a differentiating factor in the improved final exam grades. Improvement was greatest among students with the lowest course grades, though arguably these students had the most room to improve. Of the 13 students with the lowest final course grades, 12 had a higher grade on the final than the second exam with an average increase of 11.0 points.

Though these results look promising, further evaluation would be needed for conclusive results. In particular, comparing these results with other sections taught with different project approaches would be useful, however this is the first semester this course was taught by the author.

5 Conclusion

The purpose of the RPG assignment was to provide students with a large software development project in multiple stages and allow students to experience the entire design, implementation, and testing process. Students responded positively to the project and showed retention of important course concepts covered by the project including class design, encapsulation, inheritance, abstract classes, interfaces, polymorphism, unit testing, and UML. Notably, students seemed most excited by the third phase of the project in which they were required to complete the design, implementation, and unit testing from start to finish. Student performance in this part of the project combined object-

oriented and software design concepts with the fun and interesting aspects of an RPG.

The scope of the RPG project was slightly scaled back due to the COVID-19 pandemic. Similarly, the project can be scaled for other courses. To introduce procedural programming, students could finish small pieces of game and character logic that could be added to a mostly functioning game. To introduce basic objects, students could code simplified hero, weapon, and enemy classes without introducing inheritance or advanced abilities. For an advanced software development course, or a software development course that does not teach a new language at the beginning, students could take ownership of the design, implementation, and testing process at an earlier point or the beginning of the project. Students could also develop the game further and incorporate more software design principles.

Student interest in RPGs and the natural parallels between the genre and object-oriented design make RPGs an ideal foundation for an object-oriented project. Students responded enthusiastically to the multi-part, progressive RPG project during a semester that suddenly went online and caused significant upheaval in their schedules and lives. This response reinforces the understanding that student connection to the material can have a positive impact on performance and retention.

References

[1] Matt Barton. *Dungeons and Desktops: The History of Computer Role–playing Games*. A K Peters, Wellesley, MA, 2008.

[2] Jessica D Bayliss. Using games in introductory courses: tips from the trenches. In *Proceedings of the 40th ACM Technical Symposium on Computer Science Education*, pages 337–341, 2009.

[3] Katrin Becker. Teaching with games: The minesweeper and asteroids experience. *Journal of Computing Sciences in Colleges*, 17(2):23–33, December 2001.

[4] Andy Chalk. Researchers find that female PC gamers outnumber males, October 2014. [Online; posted 29-October-2014].

[5] Woei-Kae Chen and Yu Chin Cheng. Teaching object–oriented programming laboratory with computer game programming. *IEEE Transactions on Education*, 50(3):197–203, 2007.

[6] Jeffrey Edgington and Scott Leutenegger. Using the ancient game of Rogue in CS1. *Journal of Computing Sciences in Colleges*, 24(1):150–156, October 2008.

[7] Timothy Huang. Strategy game programming projects. *Journal of Computing Sciences in Colleges*, 16(4):205–213, April 2001.

[8] John E Laird. Using a computer game to develop advanced AI. *Computer*, 34(7):70–75, 2001.

[9] Timothy E. Roden and Rob LeGrand. Growing a computer science program with a focus on game development. In *Proceedings of the 44th ACM Technical Symposium on Computer Science Education*, pages 555–560, New York, NY, USA, 2013. Association for Computing Machinery.

[10] Karen Villaverde and Bretton Murphy. Game development using Greenfoot: Senior project. *Journal of Computing Sciences in Colleges*, 27(4):159–167, April 2012.

[11] Dana Vrajitoru and Paul Toprac. Games programming in computer science education. In *Proceedings of the International Conference on Frontiers in Education: Computer Science and Computer Engineering (FECS)*, pages 10–14, 2016.

[12] Mike Wallinga. Engaging CS2 students via a semester-long in-class game project. *Journal of Computing Sciences in Colleges*, 35(2):77–88, October 2019.

[13] Ursula Wolz and Sarah Monisha Pulimood. An integrated approach to project management through classic CS III and video game development. *ACM SIGCSE Bulletin*, 39(1):322–326, March 2007.

[14] Chong-wei Xu. Teaching OOP and COP technologies via gaming. In *Handbook of Research on Effective Electronic Gaming in Education*, pages 508–524. IGI Global, 2009.

Interdisciplinary Research Experience in Computer Science and Biological Sciences

Parrish Waters and Jennifer A. Polack
Biological Science, Computer Science
University of Mary Washington
Fredericksburg, VA 22401
rwaters@umx.edu, jenniferpolack@gmail.com

Abstract

Interdisciplinary research is critical to modern-day research. Today researchers should be able to work across disciplinary boundaries seemingly. However, without any practical experience, students graduating from Universities have minimal experience or training. This paper describes one interdisciplinary project that involved several computer science and biological science students and two mentors from each discipline. We description the project, the computer science perspective, the biological science perspective and a conclusion.

1 Introduction

Computer Science and biology are, by nature, an interdisciplinary field of study. But the challenge for any institution is to bring biologists together with computer scientists for productive collaboration. In today's world, science is forcing itself to become interdisciplinary by making researchers solve problems that lie outside the grasp of the individual disciplines.

Computer scientists learn how to build tools to increase efficiency and discover essential details not recognizable to the human brain. Biologists, in turn, research questions associated with living systems to find solutions that can increase quality of life. Because computer science provides the tools and biology provides the problems, the two should enjoy a happy marriage [4]. This is not

always the case; biologists are often in separate buildings from the computer scientists. Additionally, individuals are often highly scheduled and have little time for each other, and even when they do meet, they are usually of short duration [2]. This paper demonstrates how a biological scientist and a computer scientist collaborated to involve undergraduate students in a collaborative initiative.

2 Interdisciplinary Project Description

Understanding the dynamics of social interactions between animals is an important, but challenging goal of biologists. These interactions strongly influence an animal's quality of life, both in terms of immediate needs (e.g. food, shelter) as well as more distant (i.e. mating opportunities and evolutionary success) goals. The primary challenge to understanding social interactions arises when we attempt to track multiple animals simultaneously, with high fidelity - artificial intelligence and humans easily confuse individuals in real-time or video recordings. and Even when animals can be discerned, the tedium of watching and scoring hours of social animal behavior can deter the most diligent researcher. Recent technological advances in radio-frequency identification (RFID) provide a solution to these challenges. RFID tags are small (about the size of a grain of rice) devices that can be implanted under the skin of animals to provide a permanent, high fidelity identification label; tags are read by sensors, providing a unique alpha-numeric code for each animal. This technology has been used with pets for years, and is being incorporated into multiple methods used in studies involving laboratory rodent. In our laboratory, we use these tags to constantly track the position of animals in a social home cages. From these positional data, we can discern a suite of behavioral data including social affinity, sleep pattern, and feeding behavior. The only problem is the immense dataset that this technology provides. The position of each animal (up to six animals per cage in our lab) is read at 200 Hz for weeks, providing up to 1.7 million data points per day per animal. Needless to say, our datasets can get quite unwieldy – enter the need for computer science. Computer scientists create algorithms that can 'crunch' these datasets into manageable bits of information and construct visualization that represent each animal's behavioral patterns.

In our collaborative study at UMW, Dr. Polack, Dr. Waters, and an army of undergraduate researchers performed a study to construct and validate algorithms and programs that transformed these positional data into ethograms (long-term behavioral records) that shed light on the daily behavioral patterns of the individual and social behaviors of multiple mice living in a social cage. We termed our project the 'Mice Visualization Project'. Our compelling results demonstrate the scientific and pedagogical merits of this technology, and promise to address many of the challenges that this research field presents.

3 Computer Science Perspective

3.1 Software Engineering Course

The origins of the computer science portion of the Mice Visualization Project came from a request for software engineering projects. Software engineering is a four-credit senior-level course with real clients and real software. Students switch teams through the requirements, design, implementation, testing, and deliverable phases. Three student teams worked on the mice visualization project over two different semesters and three different sections of the course.

The software produced by the first group was limited and did not have longevity. Part of the problem was that the students did not listen to the recommendations of the computer science instructor on creating a stand-alone application and did not understand a few of the biological research requests. Overall the student team accomplished 2 out of the 5 tasks required of the software. Also, they hosted the software on a site only meant for development and made no plans for future hosting. But all was not lost. The computer science faculty member gained the most significant understanding of what was needed by the biological researcher — thereby allowing the computer scientist to be a better mentor and collaborator with future students and the biological researcher.

During the second semester of software engineering, two teams in two different sections attempted the mice visualization project again. Unfortunately, it was a complete failure. The student-produced software in both groups but did not meet the requirements of the actual project. Even though they were shown the current version produced by summer research, both projects got hung up on details of little importance and had group issues that never seemed to work out. Even though the project was not completed during the semester, the students learn the basics of software engineering and the pitfalls that come with developing software. From the instructor's point of view, they learned that not meeting with the client (biology professor) on a weekly basis and seeing the actual experiment was a substantial part of the problem.

3.2 Summer Science Research Colloquium

In between the two semesters of software engineering, the University has a Summer Science Research Institute program with computer science, math, biological sciences, chemistry, physics, and environmental science. Upper-level students are chosen to work with professors on research projects. Each student receives a $2800 stipend along with free room and board for ten weeks. The students may then choose to continue researching during the following school year. Also, the authors had a grant that paid for an additional student re-

searcher to participate in the ten-week research project. Therefore we had a total of three students working on the Mice Visualization Project.

All participants, students, and faculty mentors of the program are required to attend a weekly collaboration meeting, which lasts for approximately 60 minutes. Students are required to share their research successes, difficulties, and strategies from the previous week of work. Students discuss their project, and all other participants are urged to ask questions, brainstorm solutions, or suggest new ideas. These meetings are designed to form an informal collegiate environment between all involved. The relationship between faculty and students is a critical factor for undergraduate students [3].

Student selection is critical. Faculty mentors self select the students for summer research. Out of the three students, two were successful in immersing themselves in the project and completing selected portions of the project. Unfortunately, the third student never seemed to grasp what was asked of them. If students leave the ten-week program without realizing what they accomplished or what they learned, it is unlikely they will pursue research in the future. Successful research is two-fold problem mentors need to put in the time to help students understand how to research and what to research. However, the student must be self-driven to learn on their outside of mentoring. Finding the right balance between faculty mentoring and student self-learning is key to the process.

3.3 Computer Science Individual Study

Finally, the Mice Visualization Project was taken on by individual studies by three students over two semesters. Individual studies are open only students at least halfway through the computer science major and are intended to allow more advanced or specialized work by well-prepared and highly motivated students. The students were required to meet with the computer science professor every week and the biological science professor on a need to confer basis. The first semester of the individual study was to enhance the stand-alone application built during the summer science research. The second semester intended to transfer the project from a stand-alone application to a web-based application and add a predictive path taken model to the application.

The ability to self-learn and self-motivate impacted the mixed results. At the beginning of the semester, the research team sets a 15-week plan. Each week the mentor and students meet for minimally for 30 minutes, where the students and mentor review the previous weeks' work, set specific learning objectives, and formulating a plan on how to meet those objectives. The students who did not succeed on the whole always put the independent study after their traditional course work. Students who were semi-successful followed through their objective but had a hard time knowing where to research problems or

even how to start researching problems. The mentor always needed to find a solution or repeatedly point the students in the direction of finding the solution. The successful students no only follow through on the weekly objectives but met more frequently with the biologist, learned from previous lessons on how to solve weekly problems, and had effective time and information management.

4 Biological Science Perspective

Three biology students started work on this project during the Summer Science Institute, the summer research program mentioned in the previous section. One of these students had a year of experience working with the RFID program and was familiar with recording basic mouse behavior. We maintained three large cages (standard Type IV cage; 24 x 12 inch base), each containing four mice implanted with RFID tags. Each of these cages was placed onto a box containing a grid of 24 readers that covered the entirety of the cage floor. Thus, each mouse was constantly recorded by the RFID system (hereafter termed the ID Grid). This system was synced to a video camera; the software superimposed a unique icon on each mouse according to the positional data acquired by the RFID readers.

Under the direction of a faculty member (Waters) as well as the experienced undergraduate, these three students constructed a time-stamped (synced with the positional output of the RFID reader software) ethogram of each mouse's behavior. This ethogram included the behaviors listed in Table 1. We provided these behavioral data to the computer scientists, who interpreted behavioral patterns from the positional data and our behavioral observations. Additionally, the computer scientists created visualizations of the position and direction of mouse movement, as well as heat maps that represent each mouse's spatial preference in the cage.

This project provided these students an immersive 10-week research experience, during which they learned the processes involved in recording social animal behavior. Moreover, the collaborative nature of this project gave these biology students the opportunity to work with colleagues in a multidisciplinary effort to solve a complex problem. The biology students soon realized a limitation of scoring video by hand - scoring the social and individual behavior multiple mice takes around 5-10 minutes per minute of video (by even skilled observers). Therefore, these students were only able to score a couple of hours of behavior (out of 5 weeks of video recordings. They were thus able to see, first-hand, how working in a collaborative multidisciplinary group can push the impact of a study forward.

The most successful component of the software to date has been the visualization data (i.e. Mouse Visualization Project). This software is able to

provide information on the activity patterns of the mice (based on temporal patterns of movement – 'vector map'), and their social preference (overlapping the temporal patterns of spatial preference – 'heat map'), and create a visualization of these behaviors for presentation.

4.1 Failures (or Challenges)

An aim of this study is to use positional data from the RFID's alone to construct ethograms. This would be done by sharing the biological ethogram with computer scientists and have them interpret specific individual and social behaviors based on RFID data. These would include chasing, fighting, sleeping, etc. However, the positional data proved too rough for this type of analysis. The reading frame of each reader is around 4 inches, much larger than individual mice, and this results in the rapid movements of mice overlapping to the extent that we could not discern the mouse chasing from the mouse being chased. We are able to discriminate more gross behavioral patterns, such as active versus inactive, and sleeping. We are currently working on methods that can determine when animals are feeding, drinking, or in nesting boxes by limiting access to these resources and locating individual readers in these areas of the cage.

5 Conclusion

We believe that interdisciplinary research must increasingly become the standard rather than the exception [1]. Science students should participate in some collaboration with other scientists in their undergraduate career. Interdisciplinary research offers students the growth of knowledge, social benefits, and personal rewards. Even if the project is not 100% successful, both parties will learn about another discipline, see a different side to the problem, and gain a deeper understanding of their discipline. From the computer science perspective, the mentor would increase the number of meetings with both faculty members when researching independent students. While finding time is always an issue, the time put into a group meeting would be a significant benefit.

References

[1] Sally W. Aboelela and et al. Defining interdisciplinary research: Conclusions from a critical review of the literature. *Health services research*, 42(1p1):329–346, 2007.

[2] Thomas R. Cech and Gerald M. Rubin. Nurturing interdisciplinary research. *Nature Structural & Molecular Biology*, 11(12):1166, 2004.

[3] Brian K. Dean and Osamah A. Rawashdeh. An interdisciplinary undergraduate research experience program in electrical and computer engineering-lessons learned through 6 years of program operations. *2017 ASEE Annual Conference & Exposition*, 2017.

[4] Stanley Fields. The interplay of biology and technology. *Proceedings of the National Academy of Sciences*, 98(18):10051–10054, 2001.

Auto-Generated Game Levels Increase Novice Programmers' Engagement*

Michael J. Lee
Department of Informatics
New Jersey Institute of Technology
Newark, NJ 07102
mjlee@njit.edu

Abstract

A significant number of novices are learning programming using various online resources. Unfortunately, it is highly likely that these first-time learners will encounter obstacles that are too difficult to overcome on their own, especially as they reach more complicated concepts. Keeping these online learners engaged with the content is essential for them to learning programming, as their experience may have long-term effects on the way they view computing. One possible way to address this issue is to detect when a learner is having difficulties with a concept, provide them with automated help and encouragement, and present them with opportunities for more practice. For struggling learners, more practice will help them better understand the concept(s), and prepare them for later topics. We tested this by modifying an existing programming game, building on its existing frustration detector to provide learners with extra levels for more practice when necessary. We ran a controlled experiment with 400 participants over the course of 1.5 months. We found that the users who received extra levels when frustrated completed more core levels than their counterparts who did not receive these additional opportunities for more practice. Based on these findings, we believe that adaptive opportunities for more practice is essential in keeping educational game learners engaged, and propose future work for researchers and designers of online educational games to better support their users.

1 Introduction and Related Work

Over the last decade, there has been a significant increase in the number of novices learning programming on their own using various online resources such as Massive Open Online Courses (MOOCs) and educational games. However, critics have long claimed that online learning is not as effective as traditional classroom learning because of the absence of face-to-face interactions [5]. Attrition rates in introductory programming (CS1) courses may support these criticisms, with a worldwide survey showing that in-person CS1 classes and MOOCs have a 33% [3] and 95%+ [6] dropout rate, respectively.

The relatively higher retention rate for in-person courses may be partially explained by the personalized feedback students receive from their teachers. In a classroom, teachers can use various cues to determine the best way to help their students, such as giving them additional questions or practice. Unfortunately, many of these chances to help learners go unfulfilled in online contexts, which can negatively affect learning outcomes [7] and engagement [13]. In fact, studies have identified that a lack of motivation is a root cause for high dropout rates in online courses [13]. Without the help of more experienced individuals such as teachers, keeping students motivated to learn can be challenging, especially if they encounter obstacles that they cannot overcome on their own.

We may be better able to provide online learners with additional help and practice by adapting to their needs. For example, a system could detect when learners are having difficulties with a problem (i.e., frustrated), give them encouraging and helpful feedback hints, and provide them with an opportunity for more practice on the same type of problem(s). The idea of adaptive technology in educational systems is not new. Standardized testing such as the Graduate Record Examination (GRE) and the Graduate Management Admissions Test (GMAT) use Computerized Adaptive Testing (CAT) that adapts to test-takers' ability level by determining which question to ask next based on the correctness of the previous question [15, 22]. However, this type of on-the-fly adaptation might work well for individual users, but may be difficult to implement in classes with multiple students learning the same content at the same time. Course curricula are usually preplanned and difficult to deviate from once a class has started. Although teachers may have some flexibility to modify in-person course content in an ad-hoc manner, they have to consider that changing something will affect all of the students in the class. On the other hand, while online learning resources such as MOOCs, tutorials, and games also typically have fixed content/curriculum, many also have options for learners to access content at their own leisure and pace, without affecting other learners' experience in the same course.

We set out to create and evaluate a system that overcomes some of these limitations in current learning resources, providing an online curriculum that

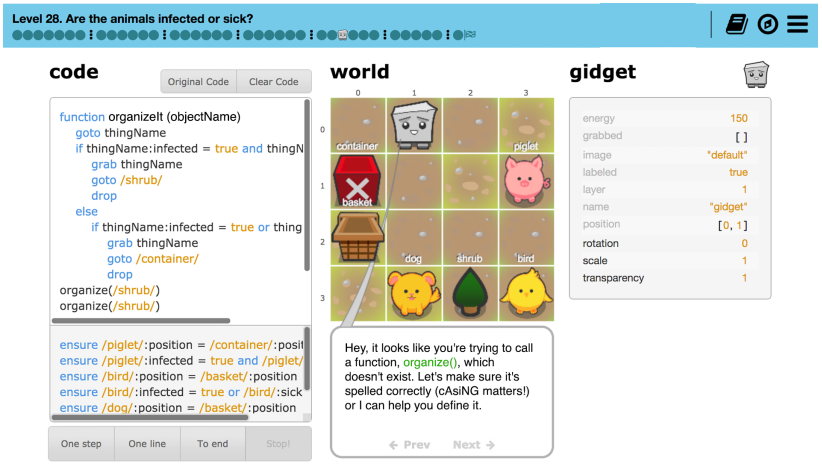


Figure 1: *A screenshot of the Gidget introductory programming game.*

adapts to the needs/skill of each individual player within an educational game. This project explores if extra automatically-generated levels, triggered by users' performance on previous levels, affects their motivation to complete more levels in an online educational programming game. We tested our game with and without this feature with 400 new users, tracking their progress through the game for one week (7 days) each, spanning a total of 1.5 months.

2 Method

2.1 The Gidget Educational Game

We modified our free, introductory coding game, Gidget *(helpgidget.org)*, for this study. The game has a total of 37 levels, where each level teaches a new programming concept (e.g., variable assignment, conditionals, loops, functions, objects) using a Python-like, imperative language [8, 11]. For each level, a player must fix existing code to help the game's protagonist. The objective of each level is to pass 1–4 test cases (i.e., statements that evaluate to `true`) after running the code. After the code runs, the game shows which test cases failed and succeeded. Each level introduces at least one new programming concept, becoming progressively harder in subsequent levels. Therefore, users are exposed to more programming concepts the farther they progress through the game. Finally, the game also includes a set of help features to help players overcome obstacles while coding on their own [8], including a coarse-grained frustration detector that provides encouraging hints/messages to those that are struggling with a level [9]. For this study, we extended the game's existing frustration detector (described in Section 2.2) to auto-generate and provide

learners with an extra level covering the same concept after they complete the current level. Details of the level generation process are detailed in Section 2.3.

2.2 Frustration Detection

We reused the frustration detector from our past work [9], which utilized coarse-grained predictors [17] to detect learners' frustration. When the system detects frustration, it provides customized feedback (and hints) to help re-engage the learner. For this study, we extended this frustration detector to additionally trigger an extra level-generator (described in 2.3). As outlined in [9], we defined frustration as, *deviations from the...*

1. ... average number of consecutive code executions with the same edit location
2. ... average time between code executions
3. ... average number of code executions
4. ... average time spent on a level
5. ... average time without any activity (idle time)

Deviation was defined as values exceeding two standard deviations from the calculated mean of any measure, as this threshold can be considered "unusual." Using a data set of 15,448 past users' game logs, we calculated all of the means and standard deviations for each of the frustration measures above. This data set was detailed, including individual players' time spent on level, idle time, all of their code edits, clicks, keystrokes, and execution button usage in the game.

2.3 Extra Level Auto-Generation Program

Once we determined that a learner was struggling with a level (reaching the threshold value of any of the measures listed in the previous section), we created a system to auto-generate an extra level for them to play *after* completing the current level. To help learners reach this newly generated level, the game includes various help features, including customized messages that are triggered by the frustration detector, which have been shown in our past works to help learners successfully overcome the issue(s) with their current level [9, 10].

The creation of our level auto-generator was guided by the work done in adaptive curricula within the intelligent tutoring and educational communities (e.g., [4, 12, 16]). For example, Baker et al.'s intelligent tutoring system gave off-task learners up to three additional exercises covering the same material, which led to learning gains that were comparable to their on-task peers [2]. To create the actual levels, we used program synthesis, which has been used by others in generating mathematical problems and proofs [1, 19], educational games [20], and introductory program auto-graders [18]. Since the programs we need to generate are mostly straightforward/uncomplicated, we create a repository of predefined programming patterns for different programming concepts,

and created a simple program synthesizer that generates levels that cover a specific concept, along with unique variable and function names, and different characters and obstacles that the player can interact with in the Gidget game.

In addition to the predefined programming patterns, we included a feature to create different world layouts for each level to differentiate and make them visually interesting for learners. The layout and puzzles of Gidget are similar to that of Sokoban puzzles—problems where a character must move items within a grid world to specific locations while avoiding environmental obstacles. We used existing procedural content generation algorithms for making Sokoban puzzles [14, 21] to generate solvable (i.e., there is at least one valid path to get an item from one position to another) Gidget levels with obstacles and objects for the character to interact with. Finally, because each generated level must contain broken code for the learner to fix (which serves as the game's instructional material), after verifying the new level is solvable using automated checks [14, 21], we used a scrambler (algorithm defined in [8]) to intentionally "break" the working code by injecting a certain number of bugs.

2.4 Pilot Study

Our goal was to provide extra practice for concepts that players struggled with, but not too many as to further disengage them. To better understand how many additional levels the system should provide a player, we ran a pilot study with 90 participants and up to 3 extra auto-generated levels for each original level (as suggested by Baker et al. [2]). We recruited these participants through the sign-up page of the game. When a new player was creating an account, they had the option to opt-in to a research study, with a high-level description of the research goals as not to prime them to the feature we were testing. Participants were randomly assigned to a condition having 1, 2, or 3 levels, where the number indicates the number of extra levels each participant would receive if they triggered the frustration detector. Each condition had 30 participants, with similar demographic characteristics.

We found that there was a statistically significant difference in levels completed among the three conditions ($\chi^2(2, N = 90) = 6.1398, p < .05$). Further post-hoc, pairwise analyses revealed that the players in the 1-level condition completed significantly more levels than both their 2-level condition ($W = 4.505, Z = -2.050, p < .05$) and 3-level condition ($W = 4.503, Z = -2.213, p < .05$) counterparts. This suggests that giving learners too many additional levels covering the same concept disengaged them from the game, leading them to quit. Based on these results, we modified our game to only include a maximum of one extra auto-generated level per original game level.

2.5 Participant Recruitment

We evaluated our system with a group of 400 new users of the game who were randomly assigned to the unmodified version of the game (the control condition) or the modified version of the game including the extra level-generator (the experimental condition). The sign-up screen asked users for their age, gender, e-mail address, a checkbox indicating whether they have prior programming experience, and a checkbox (with link to consent form) asking if they were willing to participate in a research experiment. For this study, we only selected users that indicated they were 18 years old or older, had no prior programming experience, and willing to participate in a research experiment. For selected participants, the game recorded their game condition so that they would only see the game version they were assigned to, even when coming back to play at a later time. Adapting the methodology from one of our prior studies [9], we set the observation time to 7 days (168 hours) per user to have a consistent evaluation window for all users. To promote quick account creation, we did not collect other demographic information such as ethnicity, geographical location, or education level. Participants were required to read and digitally sign an online consent form that briefly described the study. We were intentionally vague in our description of the additional levels, stating that we were "testing how extra practice can help with learning and engagement" to minimize any potential leading or biasing of participants who might be sensitive to potentially receiving additional levels (e.g., a user who perceives they are getting many levels covering the same concept might become disengaged if they feel this is a reflection of poor performance on their part). However, we e-mailed all participants a copy of the study procedures 7 days after the end of their individual observation window to debrief them, regardless of the condition they were assigned to. Prior to the debrief message (one day after their observation window ended), we sent an e-mail with a link to an optional online questionnaire. This was only sent to the experimental group participants, as only they experienced the extra auto-generated level feature. It asked them to rate their agreement to the following statements about their experience with the game on a scale from 1 ('strongly disagree') to 7 ('strongly agree'):

1. Having multiple levels covering the same concepts kept me engaged with the game.
2. I would have preferred fewer levels covering the same concepts.
3. Having multiple levels covering the same concepts helped me me better understand these concepts.

3 Results & Discussion

We report on our quantitative results comparing the outcomes from our two groups using nonparametric Wilcoxon rank sums tests, with a confidence of

$\alpha = 0.05$, as our our data were not normally distributed. The study used a between-subjects design, with 200 participants in the control condition group (aged 18-57; median 20), and 200 participants in the experimental condition group (aged 18-60; median 20). We compared demographics between the two groups, and found no significant differences between the control and experimental conditions by age or gender (99 males, 92 females, and 9 other or decline to state; and 102 males, 89 females, and 9 other or decline to state, respectively). We operationalized our key dependent variable, *engagement*, as the number of levels completed. Because those in the experimental condition saw more levels by design (i.e., a player in the experimental condition might be given extra auto-generated levels for extra practice), for all comparisons between the two conditions, we do not count the extra auto-generated levels, only comparing the core levels (i.e., the main levels that *all* players would see, regardless of condition) to measure how far they progressed in the game. We also report on the participants' responses to the optional questionnaire.

3.1 Experimental Group Complete More Core Levels

All participants in both conditions completed at least four levels. After adjusting for the auto-generated levels, the range of the core levels completed in the control and experimental conditions were 4-37 (median 11) and 4-37 (median 13), respectively. We used the game logs to verify that all experimental condition participants received at least one extra auto-generated level during their play time (with more occurring in later, more difficult stages). There was a significant difference in the number of levels participants completed between the two conditions ($W = 42401.5, Z = 1.8993, p < .05$), with the experimental group participants completing more core levels.

Since all of the participants indicated they were novice programmers, these results suggest that something about interacting with the extra levels had a significant positive effect on the experimental condition participants' engagement and ability to complete more core levels (and therefore expose themselves to more concepts) in the game compared to their control condition counterparts.

3.2 Unable to Compare Differences in Play Times

Next, we had planned to measure the differences in completion times for the core levels that participants completed. However, because everyone completed a different number of levels, we would only able to compare the levels that all 400 participants completed (i.e., Levels 1-4) to see if there were any differences in play times. Our logs indicated that only 9 of our 200 participants in the experimental condition received at least one of extra practice level during the first four levels, likely because the first few levels are very low in difficulty.

Therefore, we were unable to compare the differences between the control and experimental group play times since the majority of the experimental group (191/200, or 95.5%) did not experience anything differently from the control group for these common, completed levels.

3.3 Experimental Group Agrees Extra Levels Kept Them Engaged

For our optional questionnaire responses to the experimental group, we had a response rate of 25 out of 200 (12.5%). In our analysis, we we flipped the scale for Question 2 as it was stated negatively. Although we did not have a comparison group and response rate was low, we did find interesting trends.

For Question 1, which asked if the extra levels kept them engaged with the game, our learners gave a median score was 6 (range 4-7). This is promising, as the main goal for providing extra auto-generated levels was to give learners extra practice so they would be better prepared to complete subsequent levels. This also addressed one of our main concerns that giving learners more levels covering the topics they had just been struggling with, might be disengaging.

For Questions 2 and 3, our median scores were 4 (range 2-7) and 3 (range 1-5), respectively. Question 2 asked if learners would have preferred fewer levels covering the same concepts. Our data indicated that learners were neutral about preferring more levels, confirming our findings in our pilot study, where people did not necessarily want more levels. This makes sense, as too many levels covering the same concept can become disengaging, but too few might not help them with learning (even if they were not aware that the purpose of the next level covering the same concept was to give them more practice). This is further demonstrated in the outcome for Question 3, which asked learners if having levels covering the same concept actually helped them learn. Our participants were slightly negative from neutral, indicating that they might not have *believed* that having the extra levels were useful for learning. However, our game logs show otherwise, as the experimental group participants who experienced extra auto-generated levels completed significantly more levels than the control group participants who did not have this feature (see Section 3.1).

4 Conclusion

Our findings show that extra practice levels, triggered by a frustration detector, can significantly improve users' performance in an educational game. In our study, our experimental group participants (those who received extra practice levels for concepts that they struggling with) completed significantly more levels than their control group counterparts (who played the game without this feature). Designers for online resources teaching programming may

benefit from detecting when learners are struggling with a concept, and give them customized opportunities for extra practice on those same concepts.

We have several limitations to our study. We recruited participants who opted into a research study while signing up for an educational game. These participants may already have high motivation, and therefore may not be completely representative of the larger population. Next, we used on-the-fly auto-generated levels, where different users received completely different levels even for those covering the same concepts. Though these were generated by the same algorithm, the unique differences between these levels may have had an effect on learners, which also means it affected our results. However, we found that most learners were able to finish the extra levels the system gave them, and that they completed more subsequent levels, suggesting that the extra practice helped them. Next, we only surveyed experimental group participants and had a low overall response rate, which may limit the generalizability of the findings. In future studies, we might ask participants to complete all levels and questionnaire (adding free-response questions and/or conducting interviews to collect qualitative data). Finally, for future studies, we could use pre-post tests to measure how these new features affect players' learning outcomes.

Our study results show that extra practice levels, triggered by coarse measures to detect frustration, are sufficient in increasing online learners' level-completion performance. Our future work will examine these outcomes in more detail to isolate features that are causing these effects, and also how different types of extra practice levels might contribute to further potential differences.

5 Acknowledgements

This work was supported in part by the National Science Foundation (NSF) under grants DRL-1837489 and IIS-1657160. Any opinions, findings, conclusions or recommendations are those of the authors and do not necessarily reflect the views of the NSF or other parties.

References

[1] Chris Alvin, Sumit Gulwani, Rupak Majumdar, and Supratik Mukhopadhyay. Synthesis of geometry proof problems. In *AAAI Artificial Intelligence*, 2014.

[2] Ryan Baker, Albert T Corbett, Kenneth Koedinger, Shelley Evenson, Ido Roll, Angela Wagner, Meghan Naim, Jay Raspat, Daniel Baker, and Joseph Beck. Adapting to when students game an intelligent tutoring system. In *International Conference on Intelligent Tutoring Systems*, pages 392–401. Springer, 2006.

[3] Jens Bennedsen and Michael E Caspersen. Failure rates in introductory programming. *ACM SIGCSE Bulletin*, 39(2):32–36, 2007.

[4] Peter Brusilovsky. A distributed architecture for adaptive and intelligent learning management systems. In *Workshop "Towards Intelligent Learning Management Systems", Artificial Intelligence in Education*. Citeseer, 2003.

[5] Mark Bullen. Participation and critical thinking in online university distance education. *Int. Journal of E-Learning & Distance Education*, 13(2):1–32, 2007.

[6] Wenzheng Feng, Jie Tang, and Tracy Xiao Liu. Understanding dropouts in moocs. *Association for the Advancement of AI*, 2019.

[7] Starr Roxanne Hiltz. Collaborative learning in asynchronous learning networks: Building learning communities. 1998.

[8] Michael J Lee. Teaching and engaging with debugging puzzles. *University of Washington, Seattle, WA*, 2015.

[9] Michael J Lee. (re)engaging novice online learners in an educational programming game. *Journal of computing sciences in colleges*, 35(8), 2020.

[10] Michael J Lee, Faezeh Bahmani, Irwin Kwan, et al. Principles of a debugging-first puzzle game for computing education. In *IEEE VL/HCC*, 2014.

[11] Michael J Lee, Amy J Ko, and Irwin Kwan. In-game assessments increase novice programmers' engagement and level completion speed. In *ACM ICER*, 2013.

[12] Yanyan Li and Ronghuai Huang. Dynamic composition of curriculum for personalized e-learning. *FAIA*, 151:569, 2006.

[13] Lin Y Muilenburg and Zane L Berge. Student barriers to online learning: A factor analytic study. *Distance education*, 26(1):29–48, 2005.

[14] Yoshio Murase, Hitoshi Matsubara, and Yuzuru Hiraga. Automatic making of sokoban problems. In *PRICAI*, pages 592–600. Springer, 1996.

[15] Gregor M Novak, Andrew Gavrin, Christian Wolfgang, and Just-in-Time Teaching. Blending active learning with web technology, 1999.

[16] Neil Peirce, Owen Conlan, and Vincent Wade. Adaptive educational games: Providing non-invasive personalised learning experiences. In *IEEE DIGITEL*, pages 28–35, 2008.

[17] Ma MT Rodrigo and Ryan S Baker. Coarse-grained detection of student frustration in an introductory programming course. In *ACM ICER*, 2009.

[18] Rishabh Singh, Sumit Gulwani, and Armando Solar-Lezama. Automated feedback generation for introductory programming assignments. In *ACM SIGPLAN*, pages 15–26, 2013.

[19] Rohit Singh, Sumit Gulwani, and Sriram Rajamani. Automatically generating algebra problems. In *AAAI Conference on Artificial Intelligence*, 2012.

[20] Adam M Smith, Eric Butler, and Zoran Popovic. Quantifying over play: Constraining undesirable solutions in puzzle design. In *FDG*, pages 221–228, 2013.

[21] Joshua Taylor and Ian Parberry. Procedural generation of sokoban levels. In *INM Conference on Intelligent Games and Simulation*, pages 5–12, 2011.

[22] Brenda Cantwell Wilson and Sharon Shrock. Contributing to success in an introductory computer science course: a study of twelve factors. In *ACM SIGCSE Bulletin*, volume 33, pages 184–188. ACM, 2001.

Deep Learning in Detection of Mobile Malware*

Alex V Mbaziira, Jocelyn Diaz-Gonzales, Michelle Liu
School of Technology and Innovation
Marymount University
Arlington, VA 22207
{ambaziir, j0d88296, xliu}@marymount.edu

Abstract

Traditional malware detection techniques have been widely adopted in desktop and laptop computers. In this paper, we investigate how deep learning models can be adapted to detecting malware in mobile devices. We use a deep neural network (DNN) implementation of deep learning called DeepLearning4J (DL4J) to generate our models for mobile malware detection. The models detect mobile malware with accuracy rates ranging between 97% and 99% when applied to two types of malicious datasets. For our DNN models we vary the layers to determine effectiveness of the models in detecting mobile malware. Further analysis reveals that DNN models continue to improve in accuracy of detecting mobile malware as more layers are added to the models. This study demonstrates the practicability of using the DNN to continually learn from the past malware attacks and finally be able to predict new types of attacks. This paper sheds light on a new direction of examining malware prevention, detection and prediction and motivates future direction of exploring new strains of mobile malware that can be detected using machine learning.

Keywords— deep learning, mobile malware, neural networks, malware analysis, machine learning

1 Introduction

The first mobile malware was detected in 2004 when the Cabir strain infected Nokia devices via unsecured Bluetooth connections [4]. It stirred hornets' nest in the industry back then because of the common belief that only desktops were vulnerable to malware. Since then, protecting data on mobile devices has gradually drawn attentions in the information security field. Mobile malware is any malicious software and applications that exploits vulnerabilities in mobile applications and/or mobile devices to cause various types of damages. Some of those damages include data loss, application security compromise, potential threat to corporate network and data, and identity theft. One example would be the all-in-one Trojan that affects Android devices by forwarding calls, recording sound, conducting keylogging and deploying ransomware remotely [1]. According to one of the recent McAfee reports over 16 million mobile malware campaigns have been launched worldwide in the third quarter of 2017, which doubled the number a year ago [3]. As the threat of mobile malware is dramatically growing, there is a pressing need for more effective as well as innovative mechanisms to prevent and detect mobile malwares.

Some of the popular malware detection techniques currently being used are static, dynamic, and emulation techniques. Static techniques are a method where the mobile device is tested without running any of the detected malware. This technique involves analyzing the malicious code to determine possible defects and, or vulnerabilities which the malware exploits.

The dynamic technique on the other hand involves executing the malicious code to determine its behavior. The anomaly-based approach is based on such technique. Mobile devices would keep a record of activities and classify them as either normal or anomalous. To detect malware, the device would constantly check the devices features. Any behavior that differs from the baseline behavior would be classified as abnormal.

Another widely adopted malware detection technique is emulation-based technique which analyzes a file's behavior by opening the file in a virtual machine that simulates the real-time environment. This approach allows many different files and programs being tested at once and in a relatively parsimonious way without signaling as traditional anti-malware applications. However, emulation-based techniques may not work for all programs running on the mobile devices and it is slower than the other two methods. The main objective of this paper is to use a deep learning approach to generate malware detection models. Our approach to model generation is to study the relationship hidden layers in neural networks as well as nodes in deep learning models to malware detection accuracy rates. In this paper, we use a publicly available dataset to study application of deep learning to detect mobile malware [6]. We vary the hidden layers and nodes to determine the impact of the models in detecting mobile malware.

The rest of the article is organized as follows. The next section reviews relevant concepts and work on malware detection and machine learning. The third section focuses on the research methodology and provides the detailed steps involved in preparing datasets, generating and evaluating performance of the models. The

last section highlights the major contributions of this study as well as several future research directions.

1.1 Related Work

This section reviews the relevant literature in mobile malware detection and machine learning in order to provide background knowledge and overview of what has been done in this area.

1.1.1 Types of Machine Learning

Machine learning is the belief that computer systems can gain intelligence form data, figure out similarities and make choices with nominal assistance from a human. In general, there are four types of machine learning: supervised, unsupervised, semi-supervised and reinforcement learning [5]. The definition of each type of machine learning along with its respective merits and demerits are summarized in Table 1.

Table 1: Types of Machine Learning

Types of ML	Definition	Merits	Demerits
Supervised	Learn on a labeled dataset to determine what input goes with what output	• Can be trained to be specific • Data is known and detailed • Accurate results	• Not in the real time • A lengthy and complex process • Growing data can cause issues with getting results
Unsupervised	Extract features and patterns from unlabeled data	• In the real time • Less complex	• Not very specific • Results are harder to understand • Less accurate
Semi-supervised	Uses both labeled and unlabeled data for training	• Less expensive • Can be used with other method including classification and prediction	• Lack of a fully labeled training process • Can lead to a low performance with the wrong assumption being chosen
Reinforcement Learning	Learns from interacting with the environment and gains feedback from a reward system	• Can solve complex questions • Can correct its own errors • No training set needed	• Not for simple problems • Time consuming • Needs a lot of data

1.1.2 Malware Detection and Machine Learning

There are two major categories of malware detection systems: intrusion detection system (IDS) and intrusion prevention system (IPS). Anomaly-based intrusion detection systems (IDS) scan for patterns that might indicate an attempted attack or infiltration. Not only can these types of IDS detect intrusions that have been seen before, but it can also detect zero-day attacks or other unknown malware signatures based on a training period it goes through. However, IDS is not designed to stop an attack preemptively. An alarm or alert may be triggered, but no further actions would be taken till human support or intervention is present. Intrusion prevention

system (IPS), evolved from IDS, works by continually monitoring the device and automatically logging off a suspicious user or blocking the suspicious network traffic.

With respect to existing work on deep learning and mobile malware detection, one paper uses semi-supervised learning applying Deep Belief Networks (DBN) [8].Results from this study on implementing DBN models were compared against well studied classifiers for malware detection like Support Vector Machines, Naïve Bayes, Multilayer Perceptron, Logistic Regression and C4.5 Decision Trees [7]. Instead, our work uses DL4J deep learning models where we study the effect of the models in malware detection by varying the hidden layers and nodes.

Another paper explores android malware characterization and detection using deep learning [8]. Using features from static and dynamic analysis, the research also implements Deep Belief Networks by varying the number of neurons within the model to determine the model that gives better results. What differentiates our work from theirs, lies in the fact that we employed DL4J and varied both the hidden layers and nodes. Since the DL4J algorithm has convolutional layers, we also used input, hidden and output layers because data extracted from features is treated as text within the models.

There is also a study on deep learning and malware detection that develops a framework called Deep4MalDroid for detecting malware based on Linux Kernel System calls [2]. This paper uses graph-based features to evaluate malware detection models generated from Support Vector Machines, Artificial Network Networks, Decision Trees and Naïve Bayes.

The next section will elaborate on the methodology we have used to refine and enhance the prior works by using a deep neural network implementation of deep learning called DeepLearning4J (DL4J) for malware detection.

2 Methodology

We use two publicly available datasets to evaluate our models: Drebin and Malegenome [6]. Both datasets comprise malicious and benign instances which are also labelled. The Drebin dataset comprises of 15,036 instances while the Malegenone dataset comprises of 3799. To generate models, we used WEKA where 60% of the sample dataset was used for training and 40% was test set. The separate training and test sets were used in both datasets.

To evaluate performance of our models we examined precision (P), recall (R), F-measure (F), and Receiver Operating Characteristic (ROC) curves as shown in Tables 2 and 3. Precision measures the percentage of the relevant instances that the model declared malicious, while recall measures the number of malicious instances that are correctly detected.

To evaluate the training dataset using the DL4J algorithm, we varied the nodes from 1 through 10 and then 20, 30, 40, and 50 per layer. We kept adding one more dense layer and running the classifier with the same number of outputs as before. The same approach was replicated and repeated until we have done it for three dense layers with all the different outputs in each of the two datasets.

Table 2: Performance Evaluation Metrics of *Drebin* Models

Hidden Layers	Nodes Per Layer	Total Nodes	Precision	Recall	ROC
1	1	1	0.980	0.980	0.994
1	2	2	0.977	0.977	0.994
1	3	3	0.981	0.981	0.997
1	4	4	0.979	0.979	0.997
1	5	5	0.983	0.983	0.997
1	6	6	0.985	0.985	0.998
1	7	7	0.981	0.981	0.997
1	8	8	0.984	0.983	0.998
1	9	9	0.986	0.986	0.998
1	10	10	0.988	0.988	0.997
1	20	20	0.985	0.985	0.998
1	30	30	0.987	0.987	0.998
1	40	40	0.988	0.988	0.999
1	50	50	0.989	0.989	0.999
2	10	20	0.984	0.983	0.998
2	20	40	0.982	0.982	0.998
2	30	60	0.987	0.987	0.999
2	40	80	0.984	0.984	0.999
2	50	100	0.989	0.989	0.998
3	10	30	0.982	0.982	0.997
3	20	60	0.985	0.985	0.999
3	30	90	0.986	0.986	0.998
3	40	120	0.991	0.991	0.999
3	50	150	0.987	0.987	0.998

3 Results and Analysis

When using the DL4J classifier the number of correctly classified instances was higher than the baseline instances. Even in the lower number of output nodes the correctly classified instances were higher than the baseline.

The test results showed that the DL4J classifier correctly classified instances much better than the baseline method. When the baseline classifier was run, the number of correctly classified instances was only 69.47%. When using the DL4J classifier with one dense layer and one output, the correctly classified instances was remarkably increased to 98.68%. By increasing the number of outputs, the number of correctly classified items is on the rise as well. This indicates that the classifier is properly detecting malware. Table 4 and 5 show both datasets with output nodes set at one

Table 3: Performance Evaluation Metrics of *Malgenome* Models

Hidden Layers	Nodes Per Layer	Total Nodes	Precision	Recall	ROC
1	1	1	0.987	0.987	0.999
1	2	2	0.990	0.989	1.000
1	3	3	0.995	0.995	1.000
1	4	4	0.990	0.989	1.000
1	5	5	0.992	0.992	1.000
1	6	6	0.995	0.995	1.000
1	7	7	0.997	0.997	1.000
1	8	8	0.995	0.995	1.000
1	9	9	0.995	0.995	1.000
1	10	10	0.995	0.995	1.000
1	20	20	0.997	0.997	1.000
1	30	30	0.997	0.997	1.000
1	40	40	0.997	0.997	1.000
1	50	50	0.997	0.997	1.000
2	10	20	0.987	0.987	1.000
2	20	40	0.995	0.995	1.000
2	30	60	0.995	0.995	1.000
2	40	80	0.995	0.995	1.000
2	50	100	0.995	0.995	1.000
3	10	30	0.995	0.995	1.000
3	20	60	1.000	1.000	1.000
3	30	90	0.997	0.997	1.000
3	40	120	0.997	0.997	1.000
3	50	150	0.997	0.997	1.000

through ten and how the path is on an upward momentum.

The results for both types of models illustrate that they both performed better with the output nodes in the double digits. The results for more dense layers shown in Drebin had the best results with three dense layers and forty output layers and the result was that it classified 99.07 correctly. Malegenome also had the good results with three dense layers but with just forty outputs it correctly classified 99.74 percent of the instances correctly. This is a large boost for the correctly classified instances because both datasets baseline was only in the sixty percent realm.

With only one dense layer and the outputs set in the lower range it met the same range as the DroidFusion model which used the same datasets[6]. Once we changed the higher outputs the classifier continued to correctly classify as the DroidFusion model. DNN was able to reach and continue to reach the maximum that the other

Table 4: Predictive Accuracy of *Drebin* Models

# Layers	Nodes per layer	# Nodes	Accuracy%
1	1	1	98.01
1	2	2	97.74
1	3	3	98.14
1	4	4	97.94
1	5	5	98.27
1	6	6	98.47
1	7	7	98.07
1	8	8	98.34
1	9	9	98.6
1	10	10	98.8
1	20	20	98.54
1	30	30	98.67
1	40	40	98.8
1	50	50	98.87
2	10	20	98.34
2	20	40	98.20
2	30	60	98.67
2	40	80	98.40
2	50	100	98.94
3	10	30	98.20
3	20	60	98.54
3	30	90	98.60
3	40	120	99.07
3	50	150	98.67

approach reached. Our dataset improved the performance of detecting malware than the commonly used methods. The results corroborated that the use of DNN outperformed the DroidFusion model [6]. The basis of their model begins "by training base classifiers at a lower level and then applies a set of ranking-based algorithms on their predictive accuracies at the higher level to derive a final classifier" [6]. The model of this study uses different layers to test the device files and decide whether they are malware or not. DNN was able to correctly classify instances to the ninety ninth percentile while the DroidFusion could not reach higher than ninety eight percent range.

Even though DNN did not completely remove the risk of malware entering the device, it minimizes the risk to lower than one percent. DNN allows for mobile devices to think on their own and learn from the scanning and testing what is being run. Normal users do not have in-depth understanding as for how to detect and remove

Table 5: Predictive Accuracy of *Malgenome* Models

# Layers	Nodes per layer	# Nodes	Accuracy%
1	1	1	98.68
1	2	2	98.95
1	3	3	99.47
1	4	4	98.95
1	5	5	99.21
1	6	6	99.47
1	7	7	99.74
1	8	8	99.47
1	9	9	99.47
1	10	10	99.47
2	10	20	98.68
2	20	40	99.74
2	30	60	99.47
2	40	80	99.47
2	50	100	99.47
3	10	30	99.47
3	20	60	100
3	30	90	99.74
3	40	120	99.74
3	50	150	99.74

malware, therefore, it is important that the devices are able to do this without much human intervention. Our model demonstrates that it is possible because the device learns and quickly adapts itself to be able to capture more malware.

More dense layers and higher output nodes led the classifier to properly classifying more of the instances. Meanwhile the number of correctly classified instances continued to rise. The layers were able to communicate effectively with each other and that is how the classifier was able to detect more malware than others. The classifier was first tested on one layer with multiple output nodes options.

The results of this study show that DNN not only works in the field of mobile malware detection, but it also outperforms other approaches. DNN could be implemented on computer networks to protect organizations' servers, computers, and data that is stored on cloud servers. With hackers attacking more and more organizations to cause damages to both corporate assets and reputations, the need to secure data reaches at an all-time high. As enough layers being added in DNN the network would become almost impenetrable because of its inherent nature of keeping learning from the layers and neurons.

4 Conclusion

This paper used the multiple-layer DNN along with a classifier (DL4J) to detect and predict the presence of Android malware. One of the major findings is that as the more layers were added to the models, the detection rates for malware continued to improve. A limitation of DNN lies in the possibility of overfitting. In the future it would be interesting to see how the DL4J classifier would work on a different dataset with different training sets and test sets. This is particularly meaningful for the mobile security field to check whether the algorithm would keep preforming as well as it did in this study. In addition, there is also a need for further testing of the hidden layers in the neural network so that the best results of detecting malware can be found. The challenge that malware detection models are facing is to figure out the number of layers and connect them to the neurons that would give the model optimal results. Testing many different combinations will help find the best layer to neuron method for mobile devices. Studying different machine learning algorithms ensure that not only the best results are achieved for detecting the malware, but it will also ensure that its usability and the detection speed are feasible in real life situations.

References

[1] D. Bonderud. Mobile's Latest Malware Threat: The All-in-One Android Trojan. *Security Intelligence*, 2018.

[2] S. Hou et al. Deep4maldroid: A deep learning framework for android malware detection based on linux kernel system call graphs. In *2016 IEEE/WIC/ACM International Conference on Web Intelligence Workshops (WIW)*, pages 104–111, Oct. 2016.

[3] McAfee. Mobile threat report the next 10 years, 2018.

[4] Mobile world congress: The evolution of mobile security through the years: 2017. Accessed 2019-05-29: `https://securingtomorrow.mcafee.com/consumer/mobile-and-iot-security/mobile-security-evolution/`.

[5] P.-N. Tan et al. *Introduction to Data Mining.* Dorling Kindersley, 2014.

[6] S.Y. Yerima and S. Sezer. DroidFusion: A Novel Multilevel Classifier Fusion Approach for Android Malware Detection. *IEEE Transactions on Cybernetics*, 49(2):453–466, Feb. 2019. DOI: `https://doi.org/10.1109/TCYB.2017.2777960`.

[7] Z. Yuan et al. Droid-Sec: Deep Learning in Android Malware Detection. *Proceedings of the 2014 ACM Conference on SIGCOMM (New York, NY, USA)*, pages 371–372, 2014.

[8] Z. Yuan et al. Droiddetector: android malware characterization and detection using deep learning. *Tsinghua Science and Technology*, 21(1):114–123, Feb. 2016. DOI: `https://doi.org/10.1109/TST.2016.7399288`.

Development of a Configuration Management Course for Computing Operations Students*

Charles Border
School of Information
Rochester Institute of Technology
Rochester, NY 14623
cbbics@rit.edu

Abstract

The Operations side of deploying a modern computing application necessarily involves multiple groups working in concert to develop the application and the server side configuration that will support that application. This paper reports on efforts to develop a course that encourages students to dig into issues related to configuration management, security policy development, application auditing, business control issues, and most importantly, team work. While the course is entitled "Configuration Management" it is much more about students creating a process for secure iterative application deployment that borrows extensively from the DevOps movement.

Ansible, our chosen configuration management tool, is relatively easy to work with at the level of complexity that can be reached in an undergraduate class. What made this class different was the attempt made to create a process that would more closely mimic the Operations side of a DevOps workflow. Initial results from the class were encouraging and many lessons were learned.

1 Introduction

In the recent past only a few of the "Unicorn" companies such as Facebook, Amazon, Netflix and Google were concerned about being able to deploy software into production on distributed computing architectures multiple times

per day. To facilitate their ability to do this the "Unicorns" developed a series of practices such as Continuous Integration/Continuous Deployment (CI/CD), Test Driven Development, Agile Development and DevOps. The goal of these techniques was to be able to get the work of their thousands of developers into production as quickly, safely, reliably and consistently as possible. Most other large-scale companies have since adopted at least some of the practices of the "Unicorns" and these practices are now considered standard procedure for most large companies. This paper reports on an attempt to integrate the tools and techniques of the "Unicorns" into a Computing and Information Technology curriculum at a regional university.

The perspective taken in this paper is that the need is not so much for our students to work with the technology; it is to understand the mindset and the ideas behind the practices of the large organizations and to experience a version of this through the adoption of a process reminiscent of that of their practices. The fundamental problem that we face in attempting to model the practices of the large organizations is the problem of scale. Our students can get away with avoiding the issue of process because the teams that they work in and the individual projects they accomplish can be encapsulated in small-scale architectures over which they have complete administrative control. They do not have to learn to work together because they control the entire architecture as a single individual or as part of a team that is so small that it can quickly meet and make any changes that are required. Another issue that we have to deal with is that we tell our students too much and do not require them to develop the rules for their work. This is particularly the case with security policies.

In this class, I attempted to empower the students to develop their own security policies and write scripts to determine if the security policies were met. I provided scaffolding for this effort by providing examples that they could use and by discussing potential security policies in class on a regular basis. When you empower someone to make a decision, you have to live with the decisions that they make. Not doing so can create a trust issue that can be hard to overcome.

The class outlined in this paper is a reaction to these ideas and an attempt to constrain and facilitate at the same time student groups to better model the ideas and practices of the large organizations where many of these students will be employed. The paper is organized as follows: The paper begins with a literature review that attempts to build context around these ideas and their relevance in higher education. It then moves on to a discussion of the mechanics of the course and the methodology of organizing and running the course. After this, it discusses the results experienced in offering the course and concludes with ideas related to future offerings.

2 Literature Review

The "Unicorns" have been very forthright about the practices that they have developed to facilitate their operations. In books[6, 7, 8, 13], Facebook and Google groups[12, 4], Youtube and other videos[2] and presentations at well attended conferences and other events[14] the developers of these practices have carefully explained the ideas and techniques that they feel make them able to practice their particular craft. For academics who were listening, the call to change our curriculum was very clear.

Unfortunately, this call has not been well received. Artec, et al.[1] attempted to bring the ideas surrounding configuration management and DevOps (Infrastructure as code and Topology and Orchestration Specification for Cloud Applications (TOSCA), in their terms) into their curriculum. A problem that they ran into is they became so enamored with the rules of TOSCA that they lost the fluidity and experimentation inherent in the requirements to implement DevOps in a real world application deployment.

Olagunju[9], in the abstract of a Lightening Talks presentation discusses the differences and similarities of Agile and DevOps and makes a call for Information Technology students to understand how to work within a DevOps culture, but in the abstract does not discuss how this might be carried out.

Jiang and Kamali[5] present a more coherent view of the integration of Configuration Management (CM) in the IT curriculum in response to the changes in the IT Curriculum 2005. They discuss a 400 level course that they included in their curriculum called Application Configuration and Management whose goal was to expose students to the practices associated with CM largely from a developer's perspective. They spend a considerable amount of time in their article presenting the proposition that CM is a "Soft" skill that students should be exposed to after they have taken a sufficient amount of courses in more traditional "Hard" skills. They also discuss their surprise when this elective course became much more popular with their students in their networking and security tracks rather than their application development track.

2.1 Class Goals

This class was far from revolutionary. It had a small set of familiar outcomes associated with it and had the following goals for the students:

Operations as a process: the operational support of applications, or Operations is not a dramatic field. To be a competent practioner in Operations requires that you understand and follow a process that reliably deploys applications in a safe and secure fashion. Operations is about process.

Configurations as code treated like all other code that we work with: The title of the course is configuration management and one of the

secondary goals of the course is for students to understand how to work with Ansible as our chosen configuration management tool. However, a more important goal of the course is for Operations students to learn how to work with code. How to write code in a manner that allows others to understand and modify their code over time, how to utilize version control systems as a means to communicate with others as part of a team, and to create an auditable archive of configurations that can be controlled over time.

Web scale resilient architectures and processes: When the cloud first came on the scene as a viable architecture for application deployment many organizations looked to it as a way to save money while hosting traditional applications. This has not worked out. What has worked out well is organizations have found that deploying portions of applications to public and private clouds gives them resilience against regional outages and elasticity to respond to highly variable workloads. The problem is that the processes that they used to rely on to support local applications have needed to be adapted to support the new hybrid architectures.

Micro-service architectures (small, loosely coupled services connected via APIs): Workloads for internet-based applications are highly variable and users are notoriously impatient with perceived to be slow applications. Monolithic applications have a role in this field, but primarily as test beds for new ideas. The predominant architecture used to support web scale applications is the distributed micro-service architecture. This is the only architecture that provides a rapidly evolving migration path from small-scale new idea to support for a global user base spread across multiple public and private clouds. The labs in this class force the students to confront the messiness of the migration path from monolithic new idea to a single application deployed across multiple tiers.

Teamwork necessary to support large architectures: Large, distributed architectures require a large number of small, empowered teams to support them. Highly centralized decision-making paradigms cannot keep up with the rapid pace of change in the marketplace. Architectures need to be organized in such a fashion that teams are empowered to make changes they feel are necessary for their small corner of a large application to function effectively.

Testing: How do we know that our architecture will respond in the way that we need to changes in workload?

2.2 Class Format:

The class met on Tuesdays and Thursdays for an hour and a half. The class sessions were divided into lab classes and lectures. In lectures, I covered the concepts associated with DevOps and Continuous Integration and Continuous Deployment (CI/CD) as well as an overview of Ansible and its use. The class

had five labs, each completed over two weeks. The class was about process so I developed a predictable rhythm for the class and tried to stick to it.

Labs were due on Mondays, the following Tuesday was used as a project planning session at the end of which the groups were to hand in a project plan with students assigned to either Operations or Security, a list of deliverables with due dates, and a Gantt chart that outlined the dependencies across the deliverables. To facilitate the completion of this and other deliverables I gave students a form to be completed and handed in at the end of the Tuesday class (all forms are available upon request).

The general responsibilities and workflow of the Operations and Security teams were as follows:

Operations Team

- Goal: enhance production environment by creating a more resilient architecture that supports new functionality.
- Create a branch off of main in the versioning system.
- Make necessary modifications.
- Run modified scripts.
- Test functionality of newly created release candidate in the test environment.
- Merge modified script into the main branch.
- Run script to create new production environment.

Security Team

- Goal: ensure that the newly enhanced architecture lives up to the security and audit requirements of the organization.
- Based on required functionality for a lab, develop a list of security policies that the application should live up to.
- Create audit scripts that can be run against the Operations team's release candidate architecture to check to see if it lives up to the security policies.
- Meet with the Operations team to let them know the results of your audit and any changes that need to be made.

To help the students conceptualize the process I wanted them to follow and the points at which the two groups responsibilities intersected I developed the following flowchart:

Table 1: Interactions between the Ops team and the Sec team.

3 Methodology

As part of the introduction to the class, I carefully presented and reviewed the workflow and goals of the methodology I was asking them to pursue. I then divided the class into five groups of six. Because I did not know the students background or abilities I allowed the groups to self-select.

The students were assigned five labs for the course. Each lab built on the previous lab and reflected real issues in the iterative deployment of a web application. Because this class is about operations, and not application development, I based the labs around a basic PHP web site called "Explore California" that included multiple pages with hyperlinks, etc. As part of the fourth lab, students were asked to add a form generation component to the web site that used MariaDB as the backend database. Besides a general overview of the functionality I expected at the completion of each lab, I did not lecture about the steps involved in completing the labs. Instead, I expected them to do their own research to figure out what the operational and security issues were surrounding the increased functionality that came with each lab. I also very carefully laid out my justification for each of the labs. My goal in this was to help the students to understand that the series of labs they were working on represented some of the typical evolutionary paths that organizations pursue as an application becomes more popular and experiences increased workload.

3.1 Labs

The labs were developed to attempt to help students try the processes that we discussed in class.

Lab 1: The Golden Image: In this lab the students were to become more familiar with the Linux image that would be the basis for all their subsequent work. The Operations team was expected to write an Ansible script that would strip out all the unnecessary components of a CentOS image and prepare the image for their later work. The Security team was expected to develop a set of basic security policies and the scripts that they could run to make sure that the golden image satisfied those policies.

In an effort to help students to get started and also to give them an example of what my expectations were the following set of security policies that could serve as a starting point for their work was provided:

- All code must be stored in a versioning system that records changes made to the code and preserves a longitudinal records of who made the changes and why.
- All scripts must clearly be labeled as follows: the author of the script, the date it was written, the version number of the script, the date it passed

the Operations test suite, and the date it passed the security test suite, the problem the script is addressing, and the system that it impacts.

- Only packages needed to satisfy operational requirements may be installed on an operating system.
- No unnecessary user accounts may be installed on an operating system.
- Only file systems, directories and files needed to satisfy operational requirements may be installed on an operating system.
- Only open ports needed to satisfy operational requirements may be installed on an operating system.
- Only those SSH keys needed to satisfy operational requirements may be installed on an operating system.
- The Host file may not include any other entries than those required to satisfy operational requirements may be installed on an operating system.
- The Firewalld status must satisfy operational requirements of an operating system.

I also went over the types of Bash scripts that could be developed to create an audit script to illustrate that the security policies were satisfied.

Lab 2: Deploy a Monolithic (Single Server) Web Site ("Explore California"): Often when an organization has a new idea for an application their goal is to develop a minimum viable product version of the application to quickly test if there is sufficient user interest in the application to justify continued development. This is often done as a monolithic deployment where all the components of the application are deployed on a single server. This idea was the basis for the second lab.

Lab 3: Load Balancing across Multiple Versions of the Monolithic Application: In this lab students were tasked with deploying multiple versions of the monolithic application and a load balancer to spread traffic across the multiple versions.

Lab 4: Create a Loosely Coupled SOA by Adding a PHP Based Form Service Using a Single Instance of MariaDB: This lab required that students modify the "Explore California" web site to allow users to request information by filling out a form with their name and address. This form service was to be developed on a separate instance of MariaDB that would be accessed by all the instances of the web servers. While this is not a particularly difficult architecture to build, for this class it created a significant change to what the students were used to.

Lab 5: Release Engineering: Create a process where you can quickly switch from one version to the next. Adoption of DevOps and CI/CD methodologies are driven by the need to respond quickly, accurately and consistently to changes in the market. In this lab I try to get the students to think about how they could create a reusable, and consistent process for substituting a new version of an application for an old one.

3.2 Group Project:

When students are working on the labs that I assign them they are working on what I think is most important. In the group project students are encouraged to think about how the technologies we have discussed in class can be applied on issues they think are important. I discuss this group project every week during class and spend a lot of class time helping students to understand its importance.

The process that I use for the group project has four steps. In the first step I require each group to develop what amounts to a project plan or proposal for what they intend to do. I try to respond to each proposal within a day or two. Based on my feedback students are tasked with actually doing the work they have proposed to do. I usually give students at least a week or two to do this work. The next step involves students presenting their work to the class. After this they write a formal report outlining their successes and challenges.

4 Results

I ran this class for the first time in Fall 2018 with 30 students divided into six groups. The class was run a second time in Fall 2019. Most of the students in both iterations were from our Computing Information Technology program and had taken scripting and system administration courses prior to this course. I allowed the students to form their own groups and most of the students chose group members they knew from past classes. Based on their scripting and other programming background none of the students reported significant problems working with Ansible. Students were allowed to choose between three architectures to support their lab work, a university owned VMWare private cloud, VMWare workstation on lab machines, and any virtualization environment on their own laptops. The prime determinant of which architecture most groups chose was the degree to which the architecture was remotely accessible with the private cloud being the most easily accessed by all members of the group. VMWare on the lab computers required that anyone who wanted to access the architecture had to physically be in the lab (which has a fair amount of availability), and have shared access to the group's code repository. Relying on a student owned laptop to host a virtualization system was the worst from many

perspectives as the group would have to assemble to do their work. While I made this option available to the groups, I warned them of the problems of actually using it to host labs.

4.1 Student Performance:

While all the students in the class understood why the topics we were discussing were important, the groups who had participated in multiple Internships/co-operative work experiences seemed to come to the material more quickly. Students did not have much difficulty learning how to use Ansible. Some groups did have trouble deciding what needed to be done and then allocating the work across group members.

Dividing the groups into Operations and Security teams created a new dynamic that some groups had a difficult time getting used to. The role of the Security team in developing security policies which were to be included in the Operations team's configurations made it seem as though the Security team was creating work for the Operations team or, at least, making their work more difficult. This is a dynamic that I chose to set up and represented the work of real DevOps teams. My only effort to assuage this dynamic was to ask that group members switch from Operations to Security and vice versa throughout the class.

4.2 Student Feedback:

This was the first two times that I have run this class and student feedback was not altogether positive. Some students felt that the labs were too easy and some felt that they were too hard. Other students wanted me to spend more time telling them what to do and how to do it. I am happy with this kind of feedback and feel that I must have been doing something right.

5 Future Work

5.1 Labs:

I feel that I made a fundamental mistake in the way that I structured the class. One of the key ideas I was trying to stress was that in a DevOps workflow changes are made often and each change is small. By having only five labs, each of which made a dramatic change to the architecture I violated that idea. In the future, I hope to have many more labs each of which represents a small change in the architecture. I expect this to have the following benefits: The groups will learn to work together faster and more consistently. Since one of

the goals of the course is to get students used to working through a process, the more they can work the process the better they should learn it.

5.2 Group Project:

I am continually impressed by the projects that our students come up with and their ability to do interesting and relevant work. The projects were the best part of the class and allowed students to display both their command of the technology and their creativity.

5.3 Project planning:

Project planning has been a continual weakness for our students. In most of our classes students are told to complete a lab and they immediately start hammering on the keyboard trying to get it to work. In this class, the required project plan was my attempt to force the students to think more about developing a plan to complete their work before they started hammering on the keyboard. To a certain extent, I think that it worked. In a class about process the more planning we can have, the better.

5.4 Complexity of the Ansible scripts compared to industry based issues:

Some of the comments that I got from students dealt with the idea that the scripts that they wrote were either too hard or not hard enough. Below is a typical workflow from an industry based rolling update of a cluster of servers:

- Consult configuration/settings repository for information on involved server.
- Configure base OS on all machines and enforce desired state
- Identify a subset of the web application servers to update.
- Signal the monitoring system of an outage window prior to taking the servers off line.
- Signal load balancers to take application servers out of the load balanced pool.
- Stop the web application servers.
- Deploy the desired update in code, data, and/or content.
- Start the effected web application servers.

- Run appropriate tests of functionality on the renewed servers and code.
- Signal the load balancers to put the application servers back into the load balanced pool.
- Signal the monitoring system to resume alerts on any detected issues.
- Apply the above process to any other groups of application servers that need to be updated.
- Apply the above process to any other types of servers (such as database servers).
- Send email reports and audit reports regarding the status of the servers and the update.

While the above may reflect the needs of an industry application, I am not sure that for my class I add value to the student's abilities by forcing them to deal with all of this complexity. My question is, if I were to ask my students to do all of this work, how many of the weaker students would I lose and how much of this would just be busy work? Students today have very complicated lives and often have other issues that require their time other than classes.

5.5 Alternative Architectures:

Several student groups used the class project to implement one or more of our labs using Docker containers. Some even included Kubernetes as a container manager. The use of Docker containers represents real opportunity for this class as the small size and speed of implementation for Docker containers cuts back on the operational overhead of using virtual machines on our VMWare private cloud. There is an increased level of complexity around important issues such as persistent storage and networking with Docker. The question is does the increased complexity add value to the class given that we have access to our private cloud?

References

[1] M. Artec et. al. DevOps, introducing infrastructure-as-code. In *proceedings of the 2017 IEEE/ACM 39th IEEE International Conference on Software Engineering Companion*, IEEE, 2017.

[2] C. Dickson. A working theory of monitoring. In *USENIX*, LISA '13, Washington, D.C., USA, 2013.

[3] N. Forsgren et. al. *Accelerate: The Science of Lean Software and DevOps: Building and Scaling High Performing Technology Organizations*. IT Revolution Press, Portland, OR, USA, 2018.

[4] Google. Tech Dev Guide. `https://techdevguide.withgoogle.com/`.

[5] K Jiang and R. Kamali. Integration of configuration management into the it curriculum. In *Proceedings of SIGITE*, SIGITE, Cincinnati, OH, USA, 2008.

[6] Gene Kim et. al. *The DevOps Handbook: How to Create World-class Agility, Reliability, and Security in Technology Organizations*. IT Revolution Press, Portland, OR, USA, 2016.

[7] Gene Kim et. al. *The Phoenix Project: A Novel about IT, DevOps and Helping Your Business Win*. IT Revolution Press, Portland, OR, USA, 2016.

[8] Thomas Limoncelli. *The Practice of Cloud System Administration: Designing and Operating Large Distributed Systems, Vol. 2*. Addison-Wesley, 2015.

[9] A Olagunju. Revamping the it curriculum with agile and devops methodology. In *Proceedings of SIGITE*, SIGITE, Fort Lauderdale, FL, USA, 2018.

[10] O'Reilly. O'Reilly ideas. `https://www.oreilly.com/ideas`.

[11] Y. Pan. Security auditing course development. In *Proceedings of SIGITE*, SIGITE, Destin, FL, USA, 2007.

[12] Chuck Rossi. Rapid release at massive scale. `https://code.fb.com/web/rapid-release-at-massive-scale/`.

[13] T. Schlossnagle. *Scalable Internet Architectures*. Sam's Publishing, Indianapolis, IN, USA, 2007.

[14] USENIX. Usenix conferences. SRE Con: `https://www.usenix.org/srecon`, LISA: `https://www.usenix.org/conference/lisa19`.

The Effect of Gender on Student Self-Assessment in Introductory Computer Science Classes*

Ian Finlayson
Computer Science Department
The University of Mary Washington
ifinlay@umw.edu

Abstract

This paper will discuss the way that male and female students rate their abilities in introductory computer science courses. For the past two semesters, students in an introductory computer science course were given a survey at the end of the semester, asking about their experiences in the course. The survey asked students their gender, and also to rate themselves in the course as being either below-average, average, or above-average. The first semester used a traditional lecture-based course delivery. The second semester used team-based learning, in an effort to have students get a better sense of their own and their fellow student's abilities. In both courses however, female students rated themselves significantly lower than male students did, despite the fact that female students actually did better on average (though not statistically significantly better). This paper discusses these results in some detail, and talks about some future work which aims to ameliorate this issue.

1 Introduction

The gender gap in computer science refers to the fact that substantially more male students major in computer science than female students. According to the National Center for Education Statistics [5], only 18% of computer science degrees awarded in 2015 went to women. The gender divide is probably the

most widely researched issue in computer science education. Several papers have identified this as a serious problem, including [16], [15], [4], and [13]. We have known about the gender gap for a long time and it has not improved in recent years.

There are multiple reasons behind the gender gap. The one that will be examined in this paper is that female students just do not view themselves as being as good at computer science as their fellow students who are male. When students feel like they are below average in a discipline, it is natural that they will not choose to major in said discipline. The problem is that, as this study shows, those feelings are often not accurate – because female students *are* as capable in computer science as male ones.

For the past two semesters, a survey was given to students in an introductory computer science class. This class is taken primarily by students who do not intend to major in computer science and it fulfills a general education requirement. The first semester, we found that female students did indeed rate themselves lower than did the male students, despite the fact that women actually did *better* in the course on average.

After these disappointing results, significant changes to the structure of the course were made in an effort to ameliorate this issue for the next semester. Specifically, team-based learning was employed in place of a traditional lecture format.

The reasoning behind this decision was that, in a traditional lecture class, students don't necessarily get a feel for how they are doing relative to their peers. In an introductory class there are always a couple students who have some programming experience. These students then answer questions and seem to be naturals at computer science. A student lacking confidence might assume most of the class is like that and that they're behind when they really are not.

In team-based learning, students work together in teams to complete quizzes on material, and also work in their team during in-class activities. It was hoped that by doing this, students would get a more accurate picture of how well their classmates are doing in the class, and be more confident in their own abilities.

The same survey was then given to this second group of students. Unfortunately, while the results of the survey were slightly better, there was still a large difference in the way male and female students rated themselves.

This paper will discuss this disappointing result, and offer some commentary on why we believe this approach did not successfully close the gap between how male and female students rated their abilities. This paper presents a "negative result" of an approach that did not work, along with reasons why we believe this to be the case.

The rest of this paper is organized as follows: Section 2 will discuss related work, Section 3 will talk about the methodology of this study, including the

course design of the two classes, and the survey given. Section 4 will give the results and analysis of the surveys. Section 5 will provide a discussion on why we believe this approach was not successful. Finally, Section 6 will draw conclusions and also discuss our future ideas for working towards a solution to this important and challenging problem.

2 Related Work

This section will discuss other works that have looked into the way that male and female students assess their abilities in computer science and related disciplines. It will also discuss some related work on team-based learning.

In [6], Jones identifies eight reasons leading to a decreased number of women in STEM fields generally, including belief about intelligence and self-assessment. She presents a number of possible solutions to these issues, such as teaching the growth mindset to counteract students feeling that they are inherently worse at mathematics and engineering.

Correll [3] performed a study on the way male and female high school students assess their own mathematics abilities. She found that male students were more likely to rate their abilities in a favorable light than female students. She also found that male students have a lower bar for what constitutes success in mathematics than their female peers.

Margolis has written extensively on the barriers to entry in computer science. In [10], she explores several reasons for the lower number of women in the field. She cites the feeling that other students (usually males) are ahead of them as a primary reason that female students become discouraged in their computer science courses.

In [14], Rubio et. al. discuss gender differences in introductory programming classes. They had found that male students had higher learning outcomes than female students, and used physical computing (using Arduino boards) to close that gap. In our work, we did not find that women performed worse in introductory courses, but rated themselves lower anyway.

Alvarado and Dodd [1] discuss the gender gap at Harvey Mudd College, which has had success involving more female students into computer science. They have done this in part by introducing computer science with breadth-first topics, sending students to the Grace Hopper conference, and involving freshman students in research projects.

In [17], Werner et. al. propose the use of pair programming to help female computer science students. They posit that pair programming erodes the idea that computer science is a solitary endeavor, and that requiring team work is beneficial to female students especially. They found that all students in their study had improved confidence with pair programming, but women especially did.

Khan and Luxton-Reilly [7] also suggest that a primary reason for the gender gap is that female students view computing as only involving technology and not involving social interaction. They propose incorporating examples and exercises that relate to social science into computing courses. They suggest that this will interest female students in computer science more than typical examples and exercises.

There have been many studies on team-based learning, mostly which explore its efficacy in improving student learning outcomes. The definitive source on team-based learning is [11]. Two studies that look at team-based learning specifically for computer science courses are [8] and [9]. These papers find benefits in the team-based learning approach, but do not specifically look at how male and female students rate themselves.

3 Methodology

This study concerns two introductory computer science courses, taught in successive semesters. The first used a traditional, lecture-based delivery. In this course, the class met three days per week. The first two days, material was given to the students via lecture. The third day was used for in-class lab exercises. Students were allowed to work together in the lab, but most did not.

Team-based learning was used in the following semester. In this format, students were given readings on each week's material to complete outside of class. Then, on the first day of each week, students completed a Readiness Assessment Test, or RAT. These are short multiple choice quizzes that test a student's comprehension of the reading they completed.

In team-based learning, students complete the RAT *twice*, the first time individually and the second time with their team. When completing the RAT with their team, students used a scratch-off card to give their answers. This gives students immediate feedback on their answers. If they get a question wrong, they are able to guess again for a reduced number of points. Students working in a team need to come to a consensus on what to guess. In doing things this way, it was hoped that students would sometimes see that they got a question right individually that others in their team did not, even if they might assume that team member was further ahead then them. After the RAT, the instructor addressed any questions students had, and clarified any difficult material.

On the second weekly class meeting, the teams would work together on in-class activities. These included answering questions, solving small programming challenges, or finding problems in code. Sometimes the class would work on a larger program, with parts of the program assigned to each group to work on in their teams, then combined together as a class.

The third class meeting of the week was again a lab session. Unlike the previous semester, students were more likely to work together on the labs, because they had gotten to know their team members and became comfortable working with them.

In both semesters a survey, was given out to students. The survey asked several questions about students experiences in the course. One of the questions asked students "How do you feel you are doing in the course relative to your classmates?". The options given were "I feel that most others are doing better than me.", "I feel I am somewhere near the average.", and "I feel that I am doing better than most others.". Students were also asked to provide their gender. There were other questions about various other parts of the class, but these were not relevant for this paper. The survey was given anonymously, and was not graded.

4 Results

In the first semester, 23 of the 28 students enrolled completed the survey (82% response rate). There was a strong correlation between gender and how students rated themselves. If we assign a 1, 2, and 3 to each of the three ratings, then female students had an average of 2 while male students had an average of 2.57. Additionally, none of the male students chose the lowest category while four of the female students did.

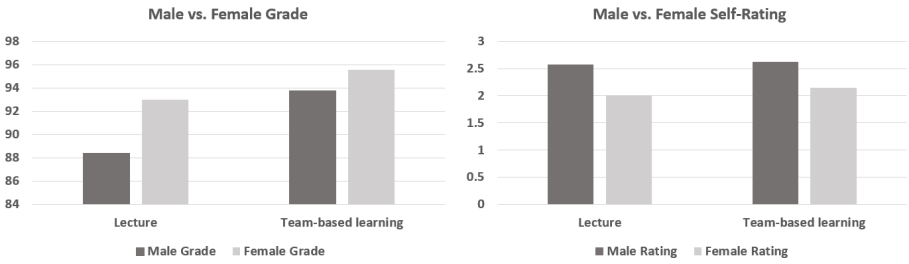
A t-test was run to determine if this difference was statistically significant. Before running the t-test, an f-test was first conducted to ensure that variance between the two groups was not equal. The independent sample, one-tailed t-test reveled that there was a strong statistically significant difference between the way that the male and female students rated themselves (t=-2.1, p=.026).

All of this is despite the fact that women actually performed *better* in the class on average with an average final grade of 92.97, as opposed to 88.39 for the male students, though this grade difference was not found to be statistically significant.

For the second semester, 15 of the 28 students completed the survey (54% response rate). This time the female student's self rating averaged 2.14 and the males averaged 2.62. So roughly the same gap as the previous semester existed, though both averages were slightly higher. Again, no male students rated themselves as below average, while two female students did. And again women actually had a better final grade average at 95.56, compared to 93.77 for men (the course had a bit of grade inflation due to the team RATs and some extra credit opportunities).

We again verified that the variance between the groups was not equal with an f-test, and then conducted an independent sample, one-tailed t-test. The

Figure 1: Summary of class performance vs. self-assessment between male and female students across the two sections



results of this test were a little less strong than the previous one (t=-1.5, p=0.079). Because the p-value is not less than 0.05, we did not quite find the statistically significant difference between the two groups that we had in the first semester. However, there is still a 92.1% chance that the variance in the groups was not caused by chance – i.e. that it's predicted by gender.

We then performed two additional t-tests to verify that the use of team-based learning had little to no effect on the way these groups of students rated themselves. First we compared the female students in the lecture-based course to the female students in the team-based learning course. The results showed no strong difference between the way these populations rated themselves (t=-0.449, p=0.331). We then compared the male students under the lecture-based course to the male students under the team-based learning course. Again, we found no significant difference (t=-0.197, p=0.424).

In conclusion, we found that under both classes gender *was* a strong predictor for how students rated themselves relative to their peers. We also found that the method of delivery (lecture vs. team-based learning) did *not* predict the students ratings.

5 Discussion

This section will provide some commentary on why we believe the use of team-based learning was not successful in closing the gap between the way that women and men rate themselves. In reflecting on this, we believe we have more insight into this problem than we previously had.

The adoption of team-based learning was made primarily because it would give students a closer look at how their fellow students solve problems, and give them a clearer idea of how they are performing relative to their classmates.

However, if a less-confident student reads the material and studies hard,

and then does better than another student on the RAT, she may not take that as a sign that she is as good, or better at computer science than the other student. She may instead just chalk it up to the fact that she read and the other student did not.

These false feelings of being behind others are related to imposter syndrome, which is the feeling many people have of being "faking it" and not actually deserving of success that they have achieved. Imposter syndrome is known to affect women more than it affects men [2]. Those with imposter syndrome often feel that their success comes from working harder than others have to, or just getting lucky, and isn't really deserved.

So we believe the primary problem with female students rating themselves lower is not really related to them actually not knowing how well they were doing in the class relative to their peers. It is instead due to them mistakenly believing that their success is not deserved. They believe there are "naturals" who are fit to be good computer scientists, and they don't see themselves as that regardless of how well they're currently doing.

6 Conclusions and Future Work

This insight has directed the future direction of this project. Rather than try to give students a more accurate view of their performance relative to their peers (which seems naïve in retrospect), we will instead focus on addressing the core of the problem, which is the underlying lack of belief in a student's own success and ability.

We propose to do that by addressing the issue head-on. In the next iteration of this project, we plan to explicitly talk about the issue of the gender gap in computer science, and imposter syndrome. Students will have readings on these topics and be asked to write reflective papers on them. We hope that talking about these issues openly will get students to think more deeply about how they view themselves.

Another idea we have is to talk about the idea of a fixed mindset vs. a growth mindset. Prior work [12] has shown that teaching a growth mindset in computer science classrooms can help students be more resilient and willing to take risks. We suspect it may also help students realize that there are no "naturals" at computer science and, conversely, anyone can work to become a successful computer scientist.

This paper presented the negative result that team-based learning made little difference in the way that male and female students rated their performance in an introductory computer science course. In both the traditional lecture-based class, and in the team-based learning one, female students rated themselves significantly lower than their male counterparts. This is an important problem because the fact that many women incorrectly see themselves as worse at computer science is a major cause of the gender gap in this field.

References

[1] Christine Alvarado and Zachary Dodds. Women in cs: an evaluation of three promising practices. In *Proceedings of the 41st ACM technical symposium on Computer science education*, pages 57–61, 2010.

[2] Pauline Rose Clance and Suzanne Ament Imes. The imposter phenomenon in high achieving women: Dynamics and therapeutic intervention. *Psychotherapy: Theory, Research & Practice*, 15(3):241, 1978.

[3] Shelley J Correll. Gender and the career choice process: The role of biased self-assessments. *American journal of Sociology*, 106(6):1691–1730, 2001.

[4] Allan Fisher and Jane Margolis. Unlocking the clubhouse: women in computing. In *Proceedings of the 34th SIGCSE technical symposium on Computer science education*, page 23, 2003.

[5] National Center for Education Statistics. Bachelor's degrees conferred to females by postsecondary institutions, by race/ethnicity and field of study: 2013-14 and 2014-15, 2015.

[6] Jenny Jones. Closing the gender gap. *Civil Engineering Magazine Archive*, 80(7):60–63, 2010.

[7] Nazish Zaman Khan and Andrew Luxton-Reilly. Is computing for social good the solution to closing the gender gap in computer science? In *Proceedings of the Australasian Computer Science Week Multiconference*, pages 1–5, 2016.

[8] Patricia Lasserre. Adaptation of team-based learning on a first term programming class. In *Proceedings of the 14th annual ACM SIGCSE conference on Innovation and technology in computer science education*, pages 186–190, 2009.

[9] Patricia Lasserre and Carolyn Szostak. Effects of team-based learning on a cs1 course. In *Proceedings of the 16th annual joint conference on Innovation and technology in computer science education*, pages 133–137, 2011.

[10] Jane Margolis, Allan Fisher, and Faye Miller. The anatomy of interest: Women in undergraduate computer science. *Women's Studies Quarterly*, 28(1/2):104–127, 2000.

[11] Larry K Michaelsen, Arletta Bauman Knight, and L Dee Fink. Team-based learning: A transformative use of small groups in college teaching. 2004.

[12] Laurie Murphy and Lynda Thomas. Dangers of a fixed mindset: implications of self-theories research for computer science education. In *Proceedings of the 13th annual conference on Innovation and technology in computer science education*, pages 271–275, 2008.

[13] Joan Peckham, Lisa L Harlow, David A Stuart, Barbara Silver, Helen Mederer, and Peter D Stephenson. Broadening participation in computing: issues and challenges. *ACM SIGCSE Bulletin*, 39(3):9–13, 2007.

[14] Miguel Angel Rubio, Rocio Romero-Zaliz, Carolina Mañoso, and P Angel. Closing the gender gap in an introductory programming course. *Computers & Education*, 82:409–420, 2015.

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

[16] Jennifer Tsan, Kristy Elizabeth Boyer, and Collin F Lynch. How early does the cs gender gap emerge? a study of collaborative problem solving in 5th grade computer science. In *Proceedings of the 47th ACM technical symposium on computing science education*, pages 388–393, 2016.

[17] Linda L Werner, Brian Hanks, and Charlie McDowell. Pair-programming helps female computer science students. *Journal on Educational Resources in Computing (JERIC)*, 4(1):4–es, 2004.

SPIFS:Short Project Instructional File System*

Robert Marmorstein
Computer Science Department
Longwood University
Farmville, VA 23909
marmorsteinrm@longwood.edu

Abstract

Understanding file systems is a key objective of both operating systems and systems programming courses. In order to enhance student understanding of file systems, it is common to ask students to implement parts of a file system using a high-level programming language. In this paper, we introduce a self-contained file system that can serve as the basis for such projects. The system includes a set of C++ classes that form the base of a very simple file system. The classes provide only the data structures and interface needed to implement the file system, but not implementations of the key file system operations. It also includes automated test cases for each of ten file system operations that students should implement.

1 Introduction

One of the key topics in many operating systems courses is the organization and design of file systems. For example, the CS2013 ACM curriculum recommendation[13] suggests that students master the following learning outcomes related to file systems:

1. Describe the choices to be made in designing file systems.

2. Compare and contrast different approaches to file organization, recognizing the strengths and weaknesses of each.

3. Summarize how hardware developments have led to changes in the priorities for the design and the management of file systems.

4. Summarize the use of journaling and how log-structured file systems enhance fault tolerance.

File system operations can also be an important topic in systems programming classes, which typically focus on understanding and using system calls and low-level libraries to accomplish system tasks.

However, file systems contain many abstract data structures and concepts that can be difficult for students to grasp. Students who are used to searching for everything in the age of Google may not even have much exposure to the use of a hierarchical directory structure. Diesburg and Berns[7] point out that one of the objectives of file system design is to hide implementation details from the user. While that creates a much better user experience, it also prevents students from mastering file system concepts through normal use. This is especially true on mobile devices, which make the file system nearly invisible. To address this, they provide a tool (Fileshark[7]) that allows students to visualize file system operations.

One excellent way to introduce students to file systems is to have them implement parts of a file system. Many operating systems courses include a project where students implement or extend parts of an operating system. For example, Andrew Tanenbaum introduced the Minix operating system[21] in 1987. Minix provides students with a fully functional clone of Unix that they can modify in clearly defined ways to explore operating systems concepts. The Nachos operating system[5] was released in the early nineties and provides a similar experience in which students implement synchronization constructs, file operations, and a virtual memory system. Nachos is based on a simulated MIPS system and, while still usable, has become a bit dated.

A team at Stanford produced a more modern version of Nachos called Pintos[17] that is written in C rather than C++. Rather than rely on a MIPS simulation, Pintos produces executable x86 code. It also comes with a set of test cases that students can use to check their implementations as they develop the software. Similarly, a team at Harvard produced the OS/161 project[11]. It has been suggested that OS/161 supports more realistic projects than Pintos, but also that it takes more time for students to complete the projects[1]. Another project of this type is the xv6 project[6] at MIT, which provides a modern version of Unix System VI.

Each of these instructional operating systems includes a component in which students design all or part of a file system. However, the file system

is heavily tied to the overall framework and the overhead that entails. As such, the file system project requires a significant investment of class time and student effort. Because implementing even the simplest operating system involves solving complex and abstract concurrency problems, these projects can be some of the most challenging assignments students tackle in an undergraduate program. In fact, as Freedman points out[8], this difficulty is one reason operating systems projects are so valuable – in addition to learning course content, students learn important lessons about programming, software design and project management. However, the time investment required to complete these projects means that they are suitable only for courses where the instructor can dedicate a major component of class time to supporting the project. Furthermore, these projects tend to be cumulative – an error in one of the first projects can affect student success in subsequent projects.

Another drawback to these projects is that they often require extensive setup. For example, Nachos requires a MIPS cross-compiler and tools for converting files to a specialized executable format[5]. Other systems require the use of virtual machines or simulators such as Bochs or Qemu[11, 6]. This makes them even more time intensive for students and faculty – and also increases the number of places where something can go wrong.

An alternative approach is to have students extend real world operating systems. Hess and Paulson[10]; Laadan, Nieh, and Viennot[14]; and Nutt[16] describe different projects that use Linux kernel extensions that are appropriate for undergraduate coursework. Schmidt, Polze, and Probert[19] describe projects that use the Windows kernel. Andrus and Nieh[2] instead use the Android mobile operating system. These projects have the advantage of realism, but the disadvantage of the additional complexity such realism requires.

A simpler approach to providing hands-on experience for operating system concepts is to assign projects that focus on a single concept or set of learning objectives. For example, Bynum and Camp[4] describe the use of the BACI concurrent language to teach synchronization principles such as deadlocks and mutual exclusion. Goldweber, Davoli, and Jonjic[9]; Buck and Perugini[3]; and Robbins and Robbins[18] describe different approaches to teaching scheduling using simulations. Sibai, Ma, and Lill[20] used the MOSS simulator to introduce students to memory management algorithms.

However, there seem to be few projects focused on solely on file system design. Instead, research has focused on ways to allow students to visualize file systems such as Fileshark[7] and the visualization tool described in [15].

2 A Basic File System

To address this gap, we present SPIFS (the Short Project Instructional File System). SPIFS provides a simple (less than 300 lines of code) self-contained base file system which students extend to implement ten basic file system operations: format, create, list, remove, rename, copy, open, close, read, and write. The file system is loosely inspired by the disk format used by Commodore DOS [12] to organize files on a floppy disk into a single flat directory, but without the complexity of disk geometry and block allocation. Spifs is implemented as a set of C++ classes that can be treated as a software library. Test cases consist of programs compiled against the library that create a file system object and then call the ten basic file system operations on that object.

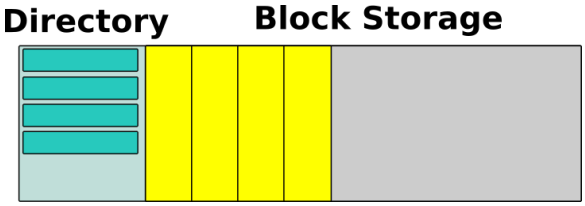


Figure 1: File System Layout

In SPIFS, the "disk" consists of a single dynamically allocated array of bytes. As shown in figure 1, we divide this array into two logical sections: a directory listing and a block storage.

We make several simplifying assumptions about the file system to reduce the complexity and make the project as accessible to students as possible. First, we assume that all files fit into a single 1KiB block. Second, we assume that the file system supports a single directory located at the beginning of the "disk" Third, we assume all operations are single-threaded. These assumptions can be weakened to extend the project and adjust the level of difficulty to suit the needs of a particular course.

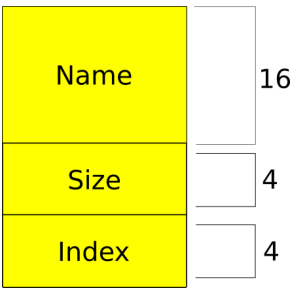


Figure 2: SPIFS Directory Entry

Conceptually, the directory structure consists of a sequential sequence of fixed-size entries. The structure of a single directory entry is illustrated in figure 2. Each entry contains a 16-byte name consisting of 15 alphanumeric

symbols plus a terminating null byte, a four byte integer representing the size of the file, and a four byte integer representing the start of the file block within the array.

We restrict filenames to alphanumeric characters and limit the size of the disk to 4GiB. For a 4MiB disk image, this gives a file system with about 4002 twenty-four byte directory entries and 4002 1Kib file blocks.

3 Testing

As students work on each of the ten operations, they can check their work by running twelve test cases provided with the base system. All twelve tests can be run using the "make test" command or test can be compiled and run individually from the tests/ folder. In addition, we encourage students to write their own tests and add them to the test suite by extending a "BaseTest" class. Providing test cases gives students immediate feedback about their progress. While the test cases aren't exhaustive, they do catch many of the ways students are likely to fail in each task.

4 Operations

The first operation students implement is a "format" operation. Since the file system consists of a single directory, the formatting task is trivial. We instruct the students to write " characters over all the filenames of the directory. The easiest way to do this is to simply copy asterisks over the whole disk, but the test case only checks the filenames, so other solutions are also acceptable.

The second task is to implement the "create" operation. This creates a new empty file by finding a free directory entry, copying the correct filename into that entry, finding a free file block, and initializing the size and index properties to correct values. Due to the simplicity of the file system, students do not need to create any additional data structures in order to allocate file blocks. They can simply map each directory entry to a corresponding file block. However, they must also properly handle error conditions, such as the existence of an existing file with the same name, an invalid filename, or a full directory.

The third task, implementing the "list" operation, requires students to iterate through the directory, printing the name and size of valid directory entries. In order to allow the testing framework to check that the list is correct, the function returns a string, rather than printing to the console.

The fourth task requires students to implement the "rename" operation. To do this, student must locate the provided filename within the directory and change it to the new name. They must properly handle error conditions, such

as renaming a file to an existing name, and attempting to rename a file that has been deleted or does not yet exist.

The fifth task requires students to delete a file. This can be handled by replacing the filename with "*" characters. As with real file systems, deleting the file does not necessarily delete the data associated with that file. The test cases verify that students check whether the file exists and has not already been deleted.

The remaining tasks require students to implement a data structure for storing "file handle" objects represent an open file on the system. Each file handle keeps track of the position at which subsequent reads or writes should begin. Implementing the "Open" task requires students to create a file handle, initialize it to point to the beginning of a file block, and return a file descriptor representing that object. Students must verify that the file exists and that there is room in the data structure for storing the file handle object.

Implementing the "Close" task frees the resources allocated for the file handle. The test cases verify that students properly handle closing an already closed file or using an invalid file descriptor.

The "Read" task asks students to implement a function that reads a fixed number of bytes from an open file handle into a buffer while the "Write" task asks them to write to the file from the buffer. In both cases, students must verify that the file descriptor is valid and that the number of bytes to be transferred does not exceed the size of the file.

5 Extending the Project

The base project can be assigned either to individuals or as a group project and is simple enough that most students can complete it within a week. It can be extended in many ways to make the project more challenging and to cover more file system concepts. Here are some relatively simple extensions that can make the content richer and more interesting:

1. Use a file allocation table to add support for multi-block files

   A file allocation table represents files as "chains" where each entry in the table consists of a single integer value representing either a free block, or the position of the next block in the chain. Blocks in the free storage of the disk are mapped one-to-one to FAT entries. Students can reserve a section of the disk array to hold the FAT and treat the directory index as the location of the first block. Most changes to the file system operations are trivial except for the create function, which must now search the FAT table for a free block. The delete, read, and write operations also require modification to correctly iterate through the linked list of blocks represented by the chain.

2. Implement multiple directories using inodes

   Instead of treating file blocks as indistinguishable blocks of data, students can treat them as structures containing a "type" field which can either be "free" "file" or "directory" In this manner, the system can represent a hierarchical directory structure with a "master directory" at the top level and subdirectories stored in file blocks. This requires the addition of new operations for creating and removing directories. It also requires extensive modification to the "List" "Open" and "Delete" operations to allow them to support paths containing multiple directories. The "Copy" operation must also be modified to allow students to move files from one directory to another. Optionally, the "Rename" function can be made to double as a "Move" function – a choice made by many modern operating systems, such as Linux.

3. Add file permissions to the directory entry object

   Since SPIFS does not have a concept of users or processes, it doesn't make much sense to implement access control lists or traditional Unix group and user permissions, but students can still modify the directory structure to allow files to be marked as "hidden" "read-only" or "append-only" This requires a minor change to the directory structure and a slight alteration of the list, read, and write operations.

4. Add support for multi-threading

   In the base project, all operations and test cases are single threaded, but it is fairly straightforward to use POSIX threads to enable parallel or asynchronous I/O operations. More complex tasks (including requiring students to implement a complete Readers/Writers system) are also possible extensions.

5. Ensure file system integrity with checksums

   One of the main concerns of a real file system is ensuring that data retains integrity and that storage errors are detected quickly. A simple way to extend the project is to replace the simple undifferentiated file blocks with a structure that adds a length field and a checksum or CRC code. Writing realistic test cases for this extension is somewhat tricky, but flipping random bits in the block storage area and verifying changes to the checksum can give a rough simulation of media errors.

6. Add routines that transfer data from the host operating system

   It is relatively trivial to add functions that copy files from the host operating system into Spifs and back out again, but adding these functions

allows students to perform more interesting experiments with their file system such as running a performance evaluation using different block sizes.

6 Conclusions

In our undergraduate operating systems course, we require students to implement projects from an instructional operating system at the beginning of the semester and use SPIFS as the final project of the semester. In our offering of the project, we split the project into two parts. In part one, students complete the ten base operations. In part two, students select an extension of their choice to implement. Both parts are graded for credit. In our experience, students can reasonably complete the project in a single week. Student feedback indicates that students view SPIFS as a much easier and more enjoyable project than the instructional operating system projects. Since it is independent of other projects it gives them a chance to "reset" and earn points toward their final grade that do not depend on previous work. In addition to giving students hands-on practical experience with file systems, SPIFS reinforces important programming concepts such as the importance of testing, data structure design, and source code management.

Grading and administering the project is also easy, since the source code requires no tools other than a C++ compiler and the score can be calculated largely from the number of test cases successfully passed. For better test coverage, the tests can be repeated using different disk image sizes.

7 Obtaining SPIFS

The latest version of SPIFS can be cloned from gitlab.com using the following command:

```
git clone https://gitlab.com/atomopawn/SPIFS.git
```

A sample handout for the project can be found in OpenDocument format at `http://marmorstein.org/~robert/SPIFS.odt`.

References

[1] Thomas Anderson and Michael Dahlin. Operating systems: Principles and practice web site (labs). `http://ospp.cs.washington.edu/labs.html`.

[2] Jeremy Andrus and Jason Nieh. Teaching operating systems using android. In *Proceedings of the 43rd ACM Technical Symposium on Computer Science Education*, SIGCSE '12, page 613–618, New York, NY, USA, 2012. Association for Computing Machinery.

[3] Joshua W. Buck and Saverio Perugini. An interactive, graphical cpu scheduling simulator for teaching operating systems. *J. Comput. Sci. Coll.*, 35(5):78–90, October 2019.

[4] Bill Bynum and Tracy Camp. After you, alfonse: A mutual exclusion toolkit. *SIGCSE Bull.*, 28(1):170–174, March 1996.

[5] Wayne A. Christopher, Steven J. Procter, and Thomas E. Anderson. The nachos instructional operating system. In *Proceedings of the USENIX Winter 1993 Conference*, USENIX'93, page 4, USA, 1993. USENIX Association.

[6] Russ Cox, Frans Kaashoek, and Robert Morris. xv6: a simple, unix-like teaching operating system. `https://pdos.csail.mit.edu/6.828/2019/xv6/book-riscv-rev0.pdf`.

[7] Sarah Diesburg and Andrew Berns. Fileshark: A graphical file system visualization tool. In *Proceedings of the 51st ACM Technical Symposium on Computer Science Education*, SIGCSE '20, page 1359, New York, NY, USA, 2020. Association for Computing Machinery.

[8] Reva Freedman. Using an operating systems class to strengthen students' knowledge of c++. In *Proceedings of the 51st ACM Technical Symposium on Computer Science Education*, SIGCSE '20, page 947–953, New York, NY, USA, 2020. Association for Computing Machinery.

[9] Michael Goldweber, Renzo Davoli, and Tomislav Jonjic. Supporting operating systems projects using the μmps2 hardware simulator. In *Proceedings of the 17th ACM Annual Conference on Innovation and Technology in Computer Science Education*, ITiCSE '12, page 63–68, New York, NY, USA, 2012. Association for Computing Machinery.

[10] Rob Hess and Paul Paulson. Linux kernel projects for an undergraduate operating systems course. In *Proceedings of the 41st ACM Technical Symposium on Computer Science Education*, SIGCSE '10, page 485–489, New York, NY, USA, 2010. Association for Computing Machinery.

[11] David A. Holland, Ada T. Lim, and Margo I. Seltzer. A new instructional operating system. *SIGCSE Bull.*, 34(1):111–115, February 2002.

[12] Richard Immers and Gerald G. Neufeld. *Inside Commodore DOS*. Reston Publishing Company, Reston, VA, USA, 1984.

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

[14] Oren Laadan, Jason Nieh, and Nicolas Viennot. Structured linux kernel projects for teaching operating systems concepts. In *Proceedings of the 42nd ACM Technical Symposium on Computer Science Education*, SIGCSE '11, page 287–292, New York, NY, USA, 2011. Association for Computing Machinery.

[15] Bruce Mechtly, Fritz Helbert, Dylan Cox, and Zachary Hastings. The visible file system: An application for teaching file system internals. *J. Comput. Sci. Coll.*, 34(4):24–31, April 2019.

[16] Gary J. Nutt. *Kernel Projects for Linux*. Addison-Wesley Longman Publishing Co., Inc., USA, 1st edition, 2000.

[17] Ben Pfaff, Anthony Romano, and Godmar Back. The pintos instructional operating system kernel. In *Proceedings of the 40th ACM Technical Symposium on Computer Science Education*, SIGCSE '09, page 453–457, New York, NY, USA, 2009. Association for Computing Machinery.

[18] Steven Robbins and Kay A. Robbins. Empirical exploration in undergraduate operating systems. *SIGCSE Bull.*, 31(1):311–315, March 1999.

[19] Alexander Schmidt, Andreas Polze, and Dave Probert. Teaching operating systems: Windows kernel projects. In *Proceedings of the 41st ACM Technical Symposium on Computer Science Education*, SIGCSE '10, page 490–494, New York, NY, USA, 2010. Association for Computing Machinery.

[20] Fadi N. Sibai, Maria Ma, and David A. Lill. Teaching page replacement algorithms with a java-based vm simulator. In *Proceedings of the 14th Western Canadian Conference on Computing Education*, WCCCE '09, page 22–28, New York, NY, USA, 2009. Association for Computing Machinery.

[21] Andrew S Tanenbaum. A unix clone with source code for operating systems courses. *SIGOPS Oper. Syst. Rev.*, 21(1):20–29, January 1987.

An Effort on Promoting K-12 Computer Science Education in Rural Region*

Jiang Li
Computer Science and Information Technology
Hood College
Frederick, MD 21701
lij@hood.edu

Abstract

This paper describes the effort that was conducted to promote K-12 Computer Science Education in rural region of IL-MO-IA tri-state area from 2016 to 2019. Computational, logic thinking and problem-solving skills are extremely important for students' success in the future. However, most schools in this rural area do not have enough resources, such as instructors, compared to metropolitan area schools. Efforts were made to provide Computer Science professional development (PD) opportunities, in hybrid format, to instructors who were interested in adopting CS related contents in their classrooms. In-person workshops were hosted during each summer, followed by multiple online sessions throughout the academic year. Our PD topics and contents were selected from various resources, based upon the CS principles framework and CSTA (Computer Science Teachers Association) standards. The feedbacks from the participants of these PD activities, based upon yearly surveys, has generally been very positive.

1 Introduction

In the 21st century, information technology is permeating many aspects of daily life. As one of the driving forces behind information technology, many aspects of computer science, such as big data, software and the internet are

being integrated into business and products throughout society. Computer science develops students' computational and critical thinking skills and shows them how to create, not simply use, new technologies. The knowledge and skills learned from studying computer science prepare students for careers in a variety of sectors.

Exposing K-12 students to computer science is extremely important for students' success in the future, however, only 35% of high schools in the US teach computer science, according to a study conducted by code.org and CSTA [5]. The study also indicates that in the past few years, more than 40 states have enacted one or more polices to make computer science education fundamental, which include creating a state plan for K-12 computer science education, implementing clear certification pathways for computer science teachers, require that all secondary schools offer computer science with appropriate implementation timelines and allow computer science to satisfy a core graduation requirement etc.

Systemic change, teacher engagement and development of teaching resources are required to bring CS to K-12[1]. Recommendations have been made to create a successful CS education program, such as ensuring teachers are prepared and supported, create continuity and coherence around learning progressions, make participation equitable etc.[2]. The importance of preparing CS teachers and possible certification models were discussed by CSTA Teacher Certification Task Force [3].

The study that focused on the CS education differences between rural and urban area, conducted by Google-Gallup indicates that "Students in the U.S., regardless of what type of community they live in, value CS and see it as important for their future careers, including 86% of rural and small-town students who say they are somewhat or very likely to have a job where they would need to know CS."[4]

2 Implementation

The project was designed to promote CS curriculum to the Quincy School District and its surrounding tri-state area (IL-MO-IA), by offering a sequence of in-person CS workshops and follow up online sessions to local and regional instructors as a professional development opportunity.

2.1 Background

According to government data, Illinois Report Card 2016-2017, the local school district students have the following traits, around 30% of students meet or exceed Academic Success criteria, which is measured by high school students

taking the SAT in Math and English Language Arts, it is lower than state average 39%. Around 45% of students are from low income families, among these students, only 13% meets or exceeds the SAT performance level, while 71% only partially meets, 38% approaching. There are 17% students with disabilities. Instructional Spending per Pupil is around $2.5k lower than state average and operational Spending per Pupil around $3k lower than state average.

While at the same time, some characteristics of instructors from our rural region are, most instructors do not have a CS background and desperately need support on gaining CS domain knowledge, most instructors plays multiple roles in their schools, teach math/business/technology/science, some are also sports coach, most instructors are the only instructor in their schools who teaches the subject. The demand for quality CS PD is high.

2.2 Recruiting

Our targeted audiences were local and regional in-service school teachers, who were teaching technology courses, or showed interests to offer such courses, or were considering to offer an adaptation of CS at their schools.

Other than using local media and news, recruiting was done through project partnership organizations. For example, Quincy Public Schools and Quincy Area Vocational Technical Center were developing a K-12 Computer Science curriculum. The development of the K-12 curriculum stems from Quincy business and community leaders who have identified CS skills as a major need in the local industry. They supported the development of this program and recruited other individuals through school districts. Other partnership includes School of Education at Quincy University, the local chapter of Illinois Computing Educators, West Central Region Education for Employment System #240 and Technology Education Association of Illinois Region #5.

Around 41% of our participants first heard about our PD opportunity through emails that were sent through our partnership organizations, another 38% from word of mouth from their colleagues.

2.3 Format

The hybrid format was used to deliver the contents and engage instructors throughout the academic year. In-person workshops were offered during the summer when most instructors were available. While the follow up discussion, reflection and assessment were held online through multiple platforms. The in-person workshop schedule cycled among fundamental activities, with an emphasis on hands-on learning of the curriculum early in the week transitioning to brainstorming and adaptation in the later part of the week. Trust and collaboration among participants were promoted. Google classroom was

used as the online platform to deliver the content knowledge and project-based learning experience throughout the academic year, along with collaboration on creating and sharing new lessons plans and projects. Mini-bonus stipend was used as a stimulus tool to encourage high involvement of activities conducted online.

2.4 Content

The contents in our PDs aim to improve our audiences' pedagogical content knowledge, to enhance the knowledge of purpose of CS education and to explore multiple teaching strategies.

CS principles framework as set by the College Board's AP Computer Science Principles curriculum and CSTA CS standards were introduced, to provide instructors better vision on the overarching picture about CS education. Software platforms, such as Scratch, MIT App Inventor, Berkley Snap, Khan Academy, code.org, code academy, as well as hardware platforms, such as Cozmo, Sphero, Wonder Workshop, Parrot Mini-drones, Lego Mindstorm were demonstrated as materials to use in classroom or develop new CS courses, based upon CSP frame work on creativity, abstraction, data and information, algorithms, programming, the internet and global impact. Discussions on which teaching strategies can be used to deliver certain contents were held, which gave instructors confidence and flexibility to teach in their classrooms. Multiple strategies were discussed, such as flip classroom, project-based learning, CS unplug, targeted learning etc.

2.5 Build a Community

One of the most important elements of our CS PD work was to create and grow the educator community that cares about CS education in the region. The project aimed to build a healthy self-sustainable CS education environment in our region, so instructors at different level can use appropriate channels to reach out for help and support when needed. Since schools in the rural community often are great distances apart with only one CS educator per school, it is even more important for them to have a close community.

Online platforms were used where participants can share their teaching experiences, observe each other's teaching, provide frequent feedback on teaching, design classes together, teach each other, and discuss teaching in general for the purpose of discovering solutions and addressing teacher's needs.

3 Survey Results

To improve, assess and determine the impact of our PD over the time, reviews, feedback and suggestions were collected every year. Pre-surveys were used to collect data before PD events and Post-surveys were used after.

One important factor that shows the impact of our project is the numbers of instructors who registered for and attended our PD, which are show in Table 1.

Table 1: Number of Participants

Academic Year	Number of Instructors Who Registered for PD	Number of Instructors Who Attended PD
2016-2017	25	20
2017-2018	35	25
2018-2019	54	40

General questions regarding teaching CS, such as "Entering this PD opportunity, my concerns about teaching CS are Understanding what CS is" and "Now that I have completed this PD opportunity, I am concerned about Understanding what CS is" were used in our surveys. Some results are shown in Figure 1 below. 2017-2018 data is used.

The post survey also called for ratings of different elements of the PD opportunity, the results are shown in Table 2.

Table 2: Ratings of Elements of PD

	Need some improvement	Met my expectations	Better than expected	Could not be better
Instructor/facilitator quality	0%	13%	30%	57%
Content (lesson plans, curriculum, etc.)	0%	17%	30%	53%
Format (lecture, interactive, etc.)	4%	4%	44%	48%
Timing/Schedule (days per week, duration of lectures, etc.)	4%	4%	35%	57%

Finally, more than 95% of participants of our PD indicated that it prompted them to seek other ongoing and next-level PD, hands-on/interactive content and would recommend this PD to other colleagues.

Figure 1: Survey Results

4 Discussion and Conclusion

From the positive data shown in the previous section, our project was concluded as a success. Quality PD was conducted to our audiences, who gained domain knowledge and were equipped with curriculum content to teach CS in their classrooms. This should in turn, help on achieving the ultimate goal, which is to provide our K-12 students the opportunity to explore and study CS, to and prepare them for further career opportunities. Some lessons were learnt on how to host successful PD. Reach out to educators in a variety of ways, schedule in-person PD wisely to be mindful of educators' time and location, provide hybrid in-person and online PD opportunities, build a community that supports educator teaching throughout the academic year, scaffold content appropriately, start with something interesting and focus on building confidence firsts, do not be afraid of trying new approaches, and be willing to learn from mistakes, collaborate with local and regional school districts, collaborate with other CS education organizations, learn from other successful PD events and think about the sustainability and scalability.

5 Acknowledgements

This project was sponsored by Google CS Educator grant.

References

[1] Valerie Barr and Chris Stephenson. Bringing computational thinking to k-12: what is involved and what is the role of the computer science education community? In *ACM Inroads*, volume 2, March, 2011, pages 48–54.

[2] P. Blikstein. Pre-college computer science education: A survey of the field. Mountain View, CA: Google LLC.

[3] J. Gal-Ezer and C. Stephenson. Computer science teacher preparation is critical. In *ACM Inroads*, volume 1, March, 2010, pages 61–66.

[4] Google Inc. and Gallup Inc. (August 2017). Computer science learning: Closing the gap: Rural and small town school districts. results from the 2015-2016 google-gallup study of computer science in u.s. k-12 schools (issue brief no. 4). Retrieved from `https://goo.gl/hYxqCr`.

[5] 2018 State of Computer Science Education. (2018). Retrieved from `https://advocacy.code.org/`.

Online System Modeling and Documentation using ROS Snapshot*

William R. Drumheller and David C. Conner

Department of Physics, Computer Science and Engineering
Christopher Newport University
Newport News, VA 23606
{william.drumheller.16, david.conner}@cnu.edu

Abstract

Robotic systems are complex systems running a number of software components to control the hardware. The Robot Operating System (ROS) is often used in such systems. The integration of these software components, even when the interfaces are heavily documented, can become a daunting task. As part of a typical Software Development Life Cycle, requirements and interfaces change for entire software systems as well as the individual software components that make up those systems.

Model-Integrated Computing (MIC) can be used to manage such complexity, and provides a means for validating the system integration throughout its lifecycle. Unfortunately, current MIC tools for ROS systems are limited and lack the ability to automatically gather information needed to make models for ROS entities. This paper presents a new tool, *ROS Snapshot*, that captures a snapshot of an existing ROS-based system during runtime, and fully documents the system using specified metamodels for each ROS entity. The resulting system model can then be used to document the current state of the system, which is especially important when upgrading versions (e.g. conversion from ROS 1.0 to ROS 2.0). We present a set of current applications for the ROS Snapshot tool, as well as future development plans to integrate the acquired system model into a full MIC system.

1 Introduction

When creating robotic agents for use in simulation or in live environments, a number of pieces of software are required for them to perform effectively. Writing and integrating software for these robotic agents can be difficult to accomplish; the Robot Operating System (ROS)[1] was designed to simplify development of such systems [1, 2]. While ROS provides a common communication framework, and addresses some of the technical challenges with the design and development of specific components (e.g. hardware drivers, perception, and control), it does not specifically address the challenge of integration and validation of a ROS-based system. Unenforced requirements and integration failures (e.g. mismatched message types or interfaces, inconsistent parameter values) can impact the performance of the entire system.

Integration challenges can be reduced through the use of Model-Integrated Computing (MIC) [3, 4]. Here, the development environment maintains a record of the component interfaces and requirements, and can use model interpreters to validate and enforce constraints between components [5]. MIC systems can produce artifacts such as configuration and deployment scripts that are verified by the defined model. This proposed infrastructure requires the existence of component models that can be arranged together, bound by integration constraints in order to enforce the correct connection of these models [6]. Without pre-existing models of the current components of any software system to use in the process of MIC, the user would be forced to create models for components for which the full details of the interface are largely unknown.

This paper presents a new `chris_ros_snapshot` tool for "taking a snapshot" of an existing ROS deployment and generating a complete software model of the ROS deployment as it exists at runtime. The tool is also useful for documenting the current system configuration. The tool is available open source at `https://github.com/CNURobotics/chris_ros_modeling`.

The paper provides a background of the ROS architecture and Model-Integrated Computing, including Domain Specific Modeling Languages and Environments. Three similar modeling tools for ROS-based systems are discussed. The `chris_ros_modeling` system and `chris_ros_snapshot` tool is presented and a specific demonstration is given. Finally, the paper discusses the long term plans for integrating this work into a generic MIC environment.

[1] `https://www.ros.org`

2 Background

2.1 Robot Operating System (ROS)

The Robot Operating System (ROS)[2] is a framework that attempts to manage the complexity involved with robotics software experiments and encourage the rapid prototyping of such experiments [1, 2]. ROS provides a structured communications layer that sits upon existing operating systems. Fundamental concepts of the ROS implementation include *nodes*, *messages*, *services*, and *actions*.

Computational processes of a ROS system are known as *nodes* or *nodelets* [1, 2]. Individual nodes communicate with each other by passing around *messages*. Messages are strictly typed data structures that can be composed to form more complex messages. Communication is facilitated by publishing or subscribing to a given *topic*. Nodes, which may exist on a number of different hosts, are connected at runtime in a peer-to-peer topology. A single node may serve the roles of publisher and subscriber to multiple topics. *NodeletManagers*[3] are used to group multiple *nodelets*[3] into a single process. Nodelets within a single manager can pass data as memory references to avoid the overhead of message publishing.

In order to accomplish simplified, synchronous transactions, calculations, or data access/conversions, a node may provide unique *services* that take a strictly typed message as input and another as output [1]. The peer-to-peer connection negotiation and configuration occurs using XML-RPC[4] for Remote Procedure Calls (RPCs) between nodes (nodelets). Services are "blocking" calls and as such are intended for simple calculations.

The *actionlib*[5] package provides the resources necessary to create client / server interfaces that execute preemptable goal requests for prolonged operations, as well as send periodic status updates back to the requesting client. The *ActionServer* and *ActionClient* communicate with each other via a ROS Action Protocol[5], which uses a defined set of basic ROS messages.

Fundamentally, ROS systems are Computer-based Systems (CBSs) [7] in which there are many components that enable their proper operation. ROS's modular structure is designed to allow different nodes, each providing specific computation (e.g. hardware interface, perception, planning) to run alongside each other [1]. ROS's modular design means that only modified nodes being debugged need to be restarted, reducing the difficulty of debugging.

[2] In this paper, we are specifically concerned with ROS 1.0 (`https://www.ros.org`); future work will extend our tool to ROS 2.0 (`https://index.ros.org/doc/ros2/`).

[3] `http://wiki.ros.org/nodelet`

[4] `http://xmlrpc.scripting.com`

[5] `http://wiki.ros.org/actionlib`

The ROS Computation Graph[6] is the peer-to-peer network of ROS processes (nodes/nodelets) that are working together to communicate and process data via messages, actions, and services. In ROS 1.0 the *roscore master*[7] node and the *parameter server*[8] play an important role in this Computation Graph. The ROS *roscore master* provides name registration and lookup for nodes to locate each other for the purposes of message sharing and service invocation[6]. The *parameter server* allows *parameter* or configuration data to be stored and retrieved by key, from a central location.

ROS provides several tools for accessing information about a running ROS setup. The `ros_comm` package[9] offers an API and individual tools like `rosnode`[10], `rostopic`[11], and `rosparam`[12], which allow for individual retrieval of basic node information, topic information, and parameter information, respectively. A Graphical User Interface (GUI) called `rqt_graph`[13] visualizes the ROS Computation Graph during runtime through the use of `ros_comm`'s `rosgraph` library. The GUI `rqt_graph` is able to collect and display the publication / subscription interconnections between nodes (with node and topic names), but is not able to display information about services, and parameters. These communications-related and graph introspection tools are provided to help in debugging deployed ROS systems.

Though ROS provides these tools for ease of debugging, they do not provide ease of integration as many configuration errors can only be caught at runtime. The use of just developer documentation for ROS packages, the software within the packages, and the software's dependencies are insufficient to ensure the proper integration of ROS nodes. As requirements change and updates are made to the components of the system, these changes can result in unintended consequences such as system failure.

2.2 Model-Integrated Computing

These types of integration and validation issues can be avoided through **Model-Integrated Computing (MIC)** [3, 4, 8, 9] as the behavior and the structure of a system are specified using models, which are updated as the structure and behavior of the system changes [9]. MIC is a method for developing, maintaining, and evolving larger, domain-specific software applications. More specifically, MIC is a model-based approach or methodology to software development that

[6] http://wiki.ros.org/ROS/Concepts#ROS_Computation_Graph_Level
[7] http://wiki.ros.org/Master
[8] http://wiki.ros.org/Parameter_Server
[9] http://wiki.ros.org/ros_comm
[10] http://wiki.ros.org/rosnode
[11] http://wiki.ros.org/rostopic
[12] http://wiki.ros.org/rosparam
[13] http://wiki.ros.org/rosgraph

allows for the automatic generation of applications and configurations from models that are created using **Domain-specific Modeling Environments (DSMEs)** [3, 9, 10]. One approach to MIC is **Model-Integrated Program Synthesis (MIPS)** [3, 10], which allows experts in a specific domain to create an integrated set of models that represent domain-specific systems. The models used in MIC must be a rigorous representation of the CBS, which must be capable of describing all aspects of the operation of the system [9]. Given such models, the DSME can be used for analysis or to create *artifacts* that can be executed on the desired target platform [10]. In the context of ROS-based systems, these artifacts would be the specific *roslaunch*[14] files and parameter configuration files used to deploy a particular version of ROS-based software.

A *model* is a mathematical abstraction that explains and possibly predicts the behavior of a physical artifact, which in our case, is a specific ROS deployment. *Metamodeling* is the act of modeling the process of creating models [10]. MIC uses the concept of metamodeling to define *metamodels* that describe the language in which domain-specific models are built [6]. The metamodeling language or meta-language is used to describe the languages that actually describe the domain models, but it should not be used to describe the domain models themselves [9]. The use of the meta-language defines specific domain languages, while the use of a specific domain language defines specific components of Computer-based Systems [11]. Thus, metamodeling is a process by which models that describe other models are constructed [6].

Since Computer-based Systems are defined by various domain models, the metamodels describe how these domain models should be organized by their ontology, syntax, and interpretation through a formal **Domain-specific Modeling Language (DSML)** [8, 9, 11].

As mentioned before, MIC allows for the automatic generation of artifacts (e.g. launch and parameter configuration files) from models that are created from using Domain-specific Modeling Environments. Since the metamodeling environment models the core layer of modeling [9], the goal of the metamodeling environment is to make rapid development of DSMEs possible [3]. Once a Domain-specific Modeling Language has been defined through the use of a metamodel, a meta-level translation can be performed to synthesize the DSME from the metamodel [10]. All of the main components of the domain are represented in the metamodeling environment, as models, thus creating the DSME [3]. Using such a DSME, a user can create *user objects*, which are instances of domain-specific models (models that define a specific information domain) [10].

There has been limited work in DSMEs for ROS-based systems; here we consider ROSMOD [12], HAROS [13], and ros_model [4, 14].

[14] http://wiki.ros.org/roslaunch

ROSMOD[15] is an open-source development toolsuite and runtime software platform for the rapid prototyping of ROS software systems [12]. ROSMOD builds upon the foundation of WebGME[16] in order to provide a DSME for the rapid prototyping of ROS-based systems [12]. Since ROSMOD is implemented as a web-based graphical interface using WebGME[17], it allows access to a version controlled database of models with integrated code [15]. Server-side plugins enable generation, compilation, and deployment of executable code, which does not require the user to modify the code[15]. Thus, ROSMOD is intended to replace the conventional *roslaunch* and parameter configuration files.

Figure 1 below represents the ROSMOD *Software* Metamodel [12]. The metamodel interface is used to create the ROSMOD Modeling Language (RML) [12]. Users do not see or interact with the metamodel as they define models of ROS software systems, which contain instances of the objects defined in RML.

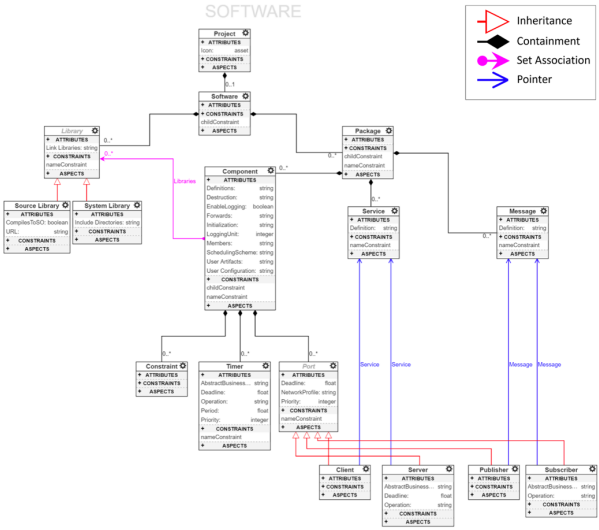


Figure 1: ROSMOD Software Metamodel from [12].

The use of RML allows for models to be validated, ensuring that the created models are correct [12]. With the use of plugins, models can be acted on to automatically generate software or facilitate deployment on desired hosts, providing a foundation where nodes can be modeled, modified, and deployed. By using such plugins to generate software or executable versions of the models,

[15] https://github.com/rosmod/webgme-rosmod
[16] https://www.isis.vanderbilt.edu/Projects/gme
[17] https://webgme.org

the software provides Model-Integrated Program Synthesis, a form of Model-Integrated Computing [3].

The major artifact provided by ROSMOD is the metamodel, which was explored in the earlier part of this chapter. ROSMOD attempts to provide an overarching metamodel that is advertised as being capable of modeling ROS Computer-based Systems.

HAROS[18] is a generic, plugin-driven framework that was created to use static analysis of ROS repositories in order to assess and report on code quality [13]. From this static analysis capability comes the opportunity to recover the ROS Computation Graph without having to consume more time, require more resources to configure and execute a software system, and lose a trail back to the actual implementation (source code). A metamodel was developed within the HAROS project to match the level of abstraction within the ROS documentation while also being traceable back to source code artifacts, which provides the benefit to a user of being more familiar and easier to reverse engineer [16]. The HAROS metamodel includes concepts such as *Node Instances*, *Topics*, *Services*, and *Parameters*, but does not currently include the concept of *Actions*. Since HAROS determines the ROS Computation Graph in a static manner, it is not possible for it to resolve specific information such as the value of a parameter; however, it does have the benefit of being able to extract remappings of node names, which does support the need for a higher-level *Node* (type) construct in the metamodel [16]. HAROS is also capable of providing visualization in the form of a graph for the model that it extracts based on the metamodel.

The ros_model [4, 14] framework uses custom ROS tools and HAROS to provide a partial solution to MIC, or as the creators have referenced it -- Model-Driven Engineering (MDE) [4, 14]. The ros_model creators, working in parallel with our effort, recognize the fact that most ROS software projects are created by the manual process of a human writing code; therefore, instead of enforcing an MDE environment strictly for the creation of ROS projects, they enable the importation of existing ROS projects as models into their environment [4, 14]. In order to accomplish this, a custom HAROS plugin extracts a model based on their metamodel in which package names, launch file names, node names, and the associated topics, services, and actions are extracted. The creators claim that ros_model overcomes challenges such as action extraction, non-C++ ROS projects, and non-Open Source Software (OSS) for which there is only a binary, all of which are not supported by HAROS [14]. The solution to support model extraction for non-C++ and non-OSS ROS projects involves the creation of a runtime system monitoring tool that makes calls to the *roscore master* to analyze the ROS Computation Graph. A web based system supports model extraction and common pattern extraction from either publicly available Git

[18] https://github.com/git-afsantos/haros

repositories (through a cloud-based solution) or from local packages (through a Docker container or local install). Additional tools were created to leverage this information including a pre-populated store of common message types and common patterns to advise users of violations such as the absence of required interfaces or a mismatch between messages for communicating nodes [14]. Since the metamodels correspond to matching Xtext[19], these constraints can easily be used to validate the extracted models, and then displayed with Eclipse[20] [4, 14].

2.3 Limitations

Experiments with `ros_comm` tools, ROSMOD, and HAROS revealed a number of limitations. Both `ros_comm` and HAROS are able to grab information about parameters; however, neither set of tools can provide information about actions, in and of themselves, nor further classification as to whether a node is a standalone node or a nodelet within a nodelet manager. The `ros_comm` tools work at runtime, but provide limited persistent output. HAROS performs static analysis of code, but cannot discover the complete ROS Computation Graph as it fails to discover all instantiated nodes that are actually present during runtime along with their associated topics. Although HAROS appears to be capable of extracting services due to its metamodel, experience from using it proves otherwise. Only `ros_comm`'s tools can provide the specification of messages and services and only HAROS can provide a mapping from nodes to parameters that read or write them. ROSMOD was mainly used to define new systems and did not work with existing launch and parameter files. For these reasons, we undertook the challenge of developing a new tool that will generate a complete system model of a deployed ROS system at runtime.

Based on the description of ros_model, it relies on HAROS to extract ROS service clients since they have found that information is not easily obtainable from the ROS Computation Graph (via the *roscore master*). In the case of non-C++ and non-OSS, HAROS cannot be used at all and when it can be used, it does not discover all of the nodes, services, and actions compared to a runtime analysis approach Further, when using the runtime system monitoring tool that is part of the ros_model environment, information related to the filesystem cannot be discovered (such as the location of the package and launch files related to a node) as well as information related to the client nodes of a specific service. When runtime analysis is invoked, the original information involved with a remap of a node or interface (such as a topic) is lost since the remap has already occurred at runtime; this prevents the original types from being extracted [14]. Overall, there appears to be no unified solution for these cases as the idea of model merging is not described.

[19] `https://www.eclipse.org/Xtext`
[20] `https://www.eclipse.org/ide`

3 ROS Modeling and ROS Snapshot Tool

As a step toward development of a full MIC system for ROS, we developed `chris_ros_modeling` and the ROS Snapshot (`chris_ros_snapshot`) tool for inspection of a running ROS system. Given the basic metamodels defined in `chris_ros_modeling`, the `chris_package_modeler` tool crawls the current ROS workspace and generates models of available packages, nodes, actions, messages, and services. The `chris_package_modeler` can be run offline to generate the specification models. On our moderately complex ROS workspace, this operation took less than eight minutes, and was significantly faster on most systems. This infrastructure is a stepping stone to offer a solution to the problem that ros_model presents with runtime analysis where filesystem information and type information (due to remaps) are lost.

After launching all ROS nodes as usual, the `chris_ros_snapshot` is executed on the same ROS network where it extracts a complete system model as instances of the `chris_ros_modeling` metamodels by querying the ROS Computation Graph and parameter server. The instantiated models are stored as dictionaries of dictionaries; this allows, for example, the topic name from a node publisher to be used to look up the message specification. The `chris_ros_snapshot` tool then stores the model of the deployed system. Both `pickle` and `YAML` forms permit reloading and manipulation of the model snapshot. The basic computation graph can be displayed using DOT[21] graphical format, and a simplified human readable text format can all be exported. The simplified human-readable form permits archiving of the entire system model at that point in time for documentation.

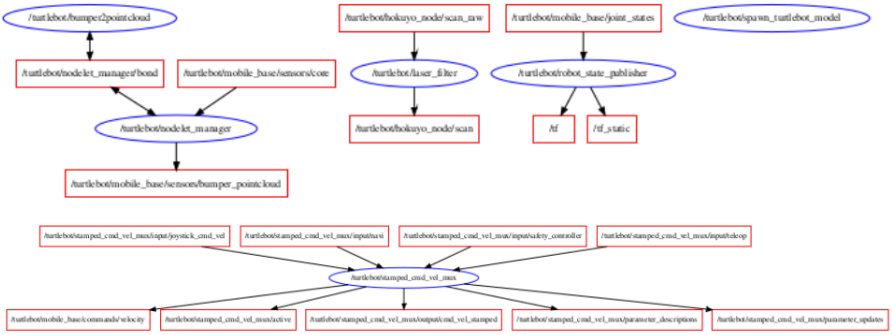


Figure 2: Extracted ROS Computation Graph from `chris_ros_snapshot`.

As an example, Figure 2 above shows the results of probing the launch [22]

[21] https://pypi.org/project/graphviz/

[22] https://github.com/CNURobotics/chris_ros_turtlebot

from [17]: `roslaunch chris_turtlebot_bringup chris_turtlebot_gazebo.launch`
This visually duplicates the results of `rqt_graph`, however, our software extracts a full system model including all node parameters and executable specifications, and bundles Action topics together. The `chris_ros_snapshot` completed its work in under one minute for this example.

To support the modeling aspects, we define the following metamodels:

- Specifications
  - `TypeSpecification`
  - `NodeSpecification`
  - `PackageSpecification`
  - `ServiceSpecification`
- Deployments
  - `Node`
  - `Nodelet`
  - `NodeletManager`
  - `Action`
  - `Machine`
  - `Parameter`
  - `Service`
  - `Topic`

The `Node` instances contain information about the name of the node, its published and subscribed topic names, and the name of the executable that is running as a node, along with process information such as the number of active threads and memory usage. `Nodelet` and `NodeletManager`, which are subclasses of the `Node`, provide additional detail about the organization of the deployed nodes. The `TypeSpecification` provides specification of the particular data structures involved with actions, messages, and services. `Topic` instances contain information about the name of the topic, the names of the publishing and subscribing nodes, and the type of the message that is being passed on that topic. Likewise, information bundled into actions and services are modeled via their respective metamodels.

A snippet of the YAML output of an instance of the Node metamodel is shown in Figure 3.

Figure 3: Extracted Instance of Node Metamodel (YAML) from `chris_ros_snapshot`.

From this YAML format, the entire system model can be reloaded; future work will enable exporting these metamodel instances to specific DSME instances for developing additional validation and generation tools [5].

Optionally, the `chris_ros_snapshot` tool can work with a modified version of the `ros_comm` package that stores some additional data in a custom *roscore master* during system startup. Our goal is to use this custom *roscore master* to capture items that are unavailable from its original public API such as the names of nodes that read and write specific parameters and, in the future, to explore the possibility of offering the names of nodes that a clients for a specific service. These are benefits that are not offered via the ros_model runtime approach.

We wrote a small script to compare and add to an extracted model from HAROS for the same turtlebot launch file. From programmatic and visual comparison of `chris_ros_snapshot` and HAROS output, it could be seen that HAROS did not discover all of the system nodes that actually exist during runtime. For the example shown in Figure 4, HAROS never extracted any information that included the `/turtlebot/stamped_cmd_vel_mux` node or any of its associated topics (since these topics were leaf topics), among another node and additional topics (shown as green circles) that were published and subscribed to by various nodes (shown as white circles).

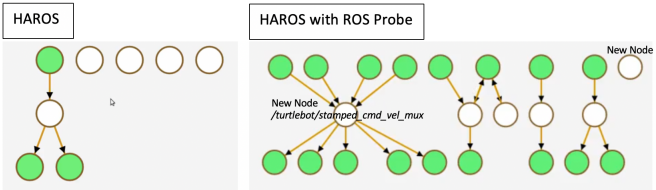


Figure 4: Comparison of Extracted Models from HAROS and `chris_ros_snapshot`.

For many large scale software projects, there is often a need to collect and maintain **Interface Control Documents (ICDs)**. In its current state, the `chris_ros_snapshot` tool is useful for documenting and archiving details about a specific deployment in a unified framework. Extracting the same information from a software archive (such as a repository) is not possible as some parameters may change at runtime and thus, may vary from those specified in parameter configuration files. Therefore, `chris_ros_snapshot` provides human-readable documentation of a ROS software system as it is currently running at the time of probe invocation.

4 Future Work and Conclusion

Currently the `chris_ros_modeling` system and the `chris_ros_snapshot` tool work to generate a software model of a ROS-based system using our defined metamodels. The output documentation is provided in `YAML`, `pickle`, and optionally simplify human readable and a computation graph visualization in `DOT` and `pdf` formats. In their current form, these output files are useful to document the state of a system, whether for use as ICD files or as part of system validation. The use of `YAML`, `pickle` formats allows the software models to be reloaded as software instances of the metamodels defined herein.

Our short term plans include developing an interface to HAROS for C++ projects to allow us to leverage their code inspection tools to further improve our specification models. The current `chris_ros_modeling` supports extensible metamodeling to incorporate additional attributes in addition to the standard ones provided by `chris_ros_modeling`. This feature will allow for us to handle the concept of merging data collected in both a static (if a C++ project) and runtime environment, which is not something currently offered by ros_model.

In future work, these software instances derived from `chris_ros_modeling` and `chris_ros_snapshot` will be used to generate a full system model within a Domain-specific Modeling Environment (DSME). One problem with existing tools such as ROSMOD is that the user must build the model by hand; while possible for new projects, this is impractical for existing projects, or for projects that incorporate existing launch and configuration files. For that reason, future work will define additional tooling to import the `chris_ros_modeling` metamodel, read in the `chris_ros_snapshot`'s output, and convert the models into a specific DSML format such as what is supported by ROSMOD or ros_model. While these translators have not yet been written, the `chris_ros_modeling` metamodel includes all of the required information for translation into any specific DSML. Given a complete system model within a DSME, a number of possibilities for system verification and synthesis open up [5].

Another area for future work will be to extend the `chris_ros_snapshot`

tool and metamodels to work with ROS 2.0 systems.

For detailed instructions on the `chris_ros_modeling` system and the use of `chris_ros_snapshot` in particular, see the open source release at `https://github.com/CNURobotics/chris_ros_modeling`. The `chris_ros_snapshot` tool can optionally work with a modified `ros_comm` repository to provide additional information. Follow the instructions in the associated `chris_ros_snapshot README` to invoke the custom `roscore master`.

References

[1] Morgan Quigley, Brian Gerkey, Ken Conley, Josh Faust, Tully Foote, Jeremy Leibs, Eric Berger, Rob Wheeler, and Andrew Ng. ROS: an open-source Robot Operating System. 2009.

[2] Morgan Quigley, Brian Gerkey, and William D. Smart. *Programming Robots with ROS: A Practical Introduction to the Robot Operating System*. O'Reilly Media, Sebastopol, CA, 2015.

[3] J. Sprinkle. Model-integrated computing. *IEEE Potentials*, 23(1):28--30, February 2004.

[4] N. Hammoudeh Garcia, M. Lüdtke, S. Kortik, B. Kahl, and M. Bordignon. Bootstrapping mde development from ROS manual code - part 1: Metamodeling. In *2019 Third IEEE International Conference on Robotic Computing (IRC)*, pages 329--336, 2019.

[5] Matt Bunting, Jonathan Sprinkle, David C. Conner, Sean Whitsitt, and Andrew Culhane. A domain-specific modeling approach to the rapid design and prototyping of autonomous vehicle software and systems. In *proceedings of AUVSI 2014*, May 2014.

[6] Sean Whitsitt and Jonathan Sprinkle. Modeling Autonomous Systems. *Journal of Aerospace Information Systems*, 10(8):396--413, 2013.

[7] J. Sztipanovits. Engineering of Computer-Based Systems: An Emerging Discipline. In *Proceedings of the IEEE ECBS'98 Conference*, 1998.

[8] J. Sztipanovits and G. Karsai. Model-integrated computing. *Computer*, 30(4):110--111, April 1997.

[9] J. M. Sprinkle, A. Ledeczi, G. Karsai, and G. Nordstrom. The new metamodeling generation. In *Proceedings. Eighth Annual IEEE International Conference and Workshop On the Engineering of Computer-Based Systems-ECBS 2001*, pages 275--279, April 2001.

[10] Greg Nordstrom, Janos Sztipanovits, Gabor Karsai, and Akos Ledeczi. Metamodeling - Rapid Design and Evolution of Domain-Specific Modeling Environments. In *Proceedings ECBS'99. IEEE Conference and Workshop on Engineering of Computer-Based Systems*, pages 68--74, March 1999.

[11] G. Karsai, G. Nordstrom, A. Ledeczi, and J. Sztipanovits. Specifying graphical modeling systems using constraint-based meta models. In *CACSD. Conference Proceedings. IEEE International Symposium on Computer-Aided Control System Design*, pages 89--94, September 2000.

[12] Pranav Srinivas Kumar, William Emfinger, Gabor Karsai, Dexter Watkins, Benjamin Gasser, and Amrutur Anilkumar. ROSMOD: A Toolsuite for Modeling, Generating, Deploying, and Managing Distributed Real-time Component-based Software using ROS. *Electronics*, 5(3), 2016.

[13] A. Santos, A. Cunha, N. Macedo, and C. Lourenço. A framework for quality assessment of ROS repositories. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4491--4496, Oct 2016.

[14] N. Hammoudeh Garcia, L. Deval, M. Lüdtke, A. Santos, B. Kahl, and M. Bordignon. Bootstrapping mde development from ROS manual code - part 2: Model generation. In *2019 ACM/IEEE 22nd International Conference on Model Driven Engineering Languages and Systems (MODELS)*, pages 95--105, 2019.

[15] M. Maróti, T. Kecskés, R. Kereskényi, B. Broll, P. Völgyesi, L. Jurácz, T. Levendovszky, and A. Lédeczi. Next Generation (Meta)Modeling: Web- and Cloud-based Collaborative Tool Infrastructure. *Journal of Aerospace Information Systems*, 2014.

[16] A. Santos, A. Cunha, and N. Macedo. Static-time extraction and analysis of the ROS computation graph. In *2019 Third IEEE International Conference on Robotic Computing (IRC)*, pages 62--69, Feb 2019.

[17] David C. Conner and Justin Willis. Flexible Navigation: Finite state machine-based integrated navigation and control for ROS enabled robots. In *IEEE SoutheastCon 2017*, pages 1--8, 3 2017.

Experiential Learning: Preparing Students for the Workforce through Faculty Mentorship and Feedback in Campus-based IT Projects*

Susan S. Conrad
Marymount University
Arlington, VA 22007
{sconrad}@marymount.edu

Abstract

Preparing students so that they can excel in the workplace is a paramount objective of academic institutions. With student debt on the rise, students need to validate their educational investment with success in the workplace. One measure of value in helping students become workforce ready is through mentoring and feedback. External internships are one method to ease students into the workforce and allow employers to test-out potential new fulltime hires. However not all students can find a suitable internship putting these individuals at a significant disadvantage. But this problem can be solved by looking within the academic organization to uncover substantive opportunities that can be resourced with a faculty mentor and students. Having a faculty member serve as the mentor in a university project bridges the gap between classroom knowledge and real-world experience connecting the dots between concepts and performance. This paper discusses the benefits of engaging faculty in university projects as they mentor students to become workforce ready through continuous feedback and scaffolding of tasks.

1 Introduction

According to the US Bureau of Labor Statistics employment of computer and information technology professionals is projected to grow 12 percent from 2018 to 2028, much faster than the average for all occupations. These occupations are projected to add about 546,200 new jobs. Increased demand for these workers is based upon greater emphasis on cloud computing, the collection and storage of big data, and information security. The bureau reported that the median annual wage for computer and information technology occupations was $88,240 in May 2019, which was higher than the median annual wage for all other occupations of $39,810 [10]. However, even though there is great demand, 22% of the recent computer science/information technology major graduates are underemployed in the United States according to a 2018 report by the Federal Reserve Bank of New York [11]. This suggests that current IT graduates may not be prepared for the workforce and recommends that academic institutions rethink how to make students workforce ready.

Academic institutions have developed an array of programs designed to teach students concepts and skills, however it's not until students apply their knowledge in real-world situations and reflect upon the experience does the "making meaning" of learning occur [2]. Experiential learning, as proposed by David Kolb has long been considered a successful approach to help students gain practical experience by applying classroom knowledge to real-world situations [7]. Workforce readiness activities such as practicums and internships have long been a component of the educational curriculum for healthcare professionals such as nursing and medicine. These successful programs have prompted other educational majors to adopt these models by requiring students to complete an internship before graduating.

2 Background

Summer internship programs have become the norm with students even in high school aggressively seek opportunities to gain real-world experience. This competitive environment has become a way for employer to identify top talent and build a pipeline for the next generation of employees [14]. With students everywhere seeking internships, students face new pressures to excel in the internship often without the benefit of mentors to provide feedback and guidance for improved performance.

Employers want students who can jump into the work environment and be productive with minimal coaching. Students come to the internship hoping to learn but often they report that mentors assigned from the employer were too busy to guide them and the internship mentor struggled to find tasks to

give the intern that didn't require much explaining [17]. This mismatch of expectations leads to poor internship experiences and dissatisfaction from the employer. According to a study of employers hiring interns, those interns who were viewed most successful possessed several common characteristics: strong communication skills, personal drive and critical thinking [17]. Given the importance of external work opportunities for students to contextualize classroom learnings and apply knowledge, it is imperative to build strong foundational programs to support students in this process.

Universities also are under pressure to adequately prepare students for the workforce as successful alumni are more likely to donate back to their school. Endowments and scholarship funds require alumni to be gainfully employed in order for them to donate back to their school. Studies have revealed that students who have strong marketable internship experiences are more competitive in the job market and reap the benefit of higher starting salaries [19]. Employers value work experience in new graduates as it reduces the start-up time for a new graduate to become productive[7].

Research from students who completed internships revealed that students sometimes felt unsure about how to approach problems regarding work assignments or company practices. Some students reported feeling isolated with minimal guidance from the organization's management [17]. Preparing students for external internships will help students have greater confidence in their internship experience and lead to greater work success.

Learning experts suggest that students need a place where they can make mistakes without fear of being fired and receive feedback in a non-threatening environment [1]. Given the importance of external internships, universities have an opportunity to look inward as a way to expand workforce readiness opportunities for students and engage faculty in the process especially in the area of Information Technology (IT). Universities, like most businesses require core IT infrastructure with systems such as: Finance, Human Resources, Payroll, Inventory to operate. These IT functions, rolled into an Enterprise Resource Planning (ERP) system are generally developed by software vendors and marketed across industries. ERP implementations expose students to state-of-the-art technology and provide students highly marketable skills which are immediately transferable to the marketplace. Since these system implementations are highly complex, faculty mentoring is essential for students to connect classroom concepts with implementation tasks. Thus, providing students opportunities to work on internal IT projects mentored by a knowledgeable faculty member afford students an opportunity to be employed in a safe real-world setting while receiving feedback for improved performance on a continual basis.

Internal IT projects mentored by faculty members is a win not only for the students but also for the university. Students can perform tasks that may have

been done by an employee or consultant, thus saving the university money while gaining marketable skills. In addition, it is important to remember that some students struggle to find an external internship due to transportation, housing, funding, visas or location therefore having meaningful opportunities available to students on campus demonstrates a workforce readiness path for all students.

This paper will discuss a work experiment conducted at a small university in which students were integrated as an essential component of an ERP implementation in the Information Technology department. It will discuss benefits to the university, faculty and students and also provide a framework for developing meaningful internal IT internships from real campus IT projects.

3 Feedback and Mentoring

Experts agree that students are more likely to be successful in their internship if they are exposed to models for imitation and feedback on performance [1] Educators have long known that feedback is essential for student learning but feedback is also critical for employees to receive feedback on their work performance . According to the industry experts, regular feedback will keep the intern engaged and provide opportunities to ask questions and clear-up misconceptions. Utilizing the "sandwich approach" to providing feedback suggests that corrective feedback be layered within other types of feedback such as praise or motivation [13].

Researchers agree that feedback is essential for helping students learn and improve performance. Feedback is essential for not just for educational purposes but for growth and improvement in the work environment. Experts have broken feedback in to 10 categories: (1) Administrative feedback provides information about due dates, technology formats, and other housekeeping issues [5] . (2) Affective feedback acknowledges the individual's participation and offers support [3]. (3) Corrective feedback provides the individual with information about personal performance and aims to redirect learning when errors are present [9]. (4) Praise feedback provides the individual positive reinforcement about performance on a task [4]. (5) Informative feedback comments on a individual's performance – could be a neutral comment [3]. (6) Socratic feedback poses reflective questions to the individual in order to stimulate deeper learning [3]. (7) Motivational feedback consists of comments or actions that focus on the unique needs of the individual with the intent to keep the person engaged, as opposed to corrective feedback, which focuses on content [15]; (8) Social feedback provides information to the individual about the accuracy of his or her response from peers, mentors, and observers [18]. (9) Future-focused feedback provides individuals with feedback that will assist the person in ap-

plying new concepts to future actions [5]; (10) Self-regulating feedback provides prompts and comments to help the individual stay on task [18].

Mentoring students by providing feedback can significantly help students learn complex tasks and apply their classroom knowledge to the job assignment without fear of failure. These personalized touchpoints have been shown to build confidence in the student's perceptions about their abilities and workforce-readiness . According to a study about what types of feedback students want, the results showed that meaningful feedback had to be timely, detailed and consistent [12]. These findings align with findings from career advisors where workers also want timely and detailed feedback especially for performance in need of improvement. Thus, creating meaningful work experiences that utilize feedback tools from academia will help to prepare students for employment outside the university and build student confidence and skills.

4 Experiential workforce rediness model

Experimental learning combines previous work experience, perceptions, cognition and behavior to create learning events [8]. Whereas behavioral theories link learning to stimuli and responses and social learning explain learning through models, experiential learning takes a more holistic approach to reinforcing concepts and beliefs through actual experiences. This paper suggests a model designed to guide faculty in implementing an experiential on-campus internship.

The model called the Experiential Workforce Readiness Model is a 7-step process to create and sustain a meaningful campus internship experience. The following describes the steps in this model: Step (1) Identify a project and project stakeholder who will be patient and open to working with students. Identifying the correct stakeholders along with selecting appropriate projects is pivotal in the workforce readiness experience. The experience should be current and relevant for future employability of the student. Faculty mentors must be knowledgeable about the project in order to guide the team. Step (2) Once the project is identified, it is essential to carve out specific tasks that can be completed by students. The tasks must be broken into smaller components such that students can roll on and off tasks with relative ease once their internship time has ended. The roles must be clearly defined with responsibilities and deliverables. Examples and artifacts are very useful to help students know what is expected of them. Step (3) Identify the correct student for the correct role. Just as a hiring manager evaluates candidates based upon existing skills and experiences, students will be selected for roles based upon their interests and class experiences. For example, a student who dislikes coding would not be a good choice for developing data conversion programs or APIs. Once students

are selected for the team, the students collaboratively identify the norms for their team. Step (4) Identify tasks, deliverables and deadlines for the students. The student must realize that their assigned tasks will impact the entire project so adherence to deadlines is team standard for all team members. Step (5) In this step the faculty member helps the students connect project deliverables with prior learning. Students discuss the theories learned through coursework and apply it to the tasks being performed. Step (6) Critique, provide feedback and work-product revision are essential components to the learning experience. Students have a chance to critique the work of peers and the faculty mentor provides feedback to help the individual student(s) improve performance. Step (7) is the final element in the learning experience. It is here that the student does self -discovery of his/her own learning and where "mental models" are created to support deep learning [6]. The student connects work performed with classroom concepts for deep learning. Figure 1 describes the steps to develop an Experiential Workforce Readiness opportunity on campus.

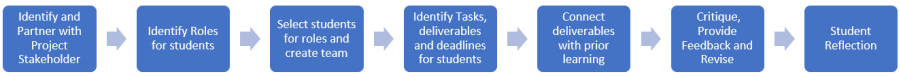


Figure 1: Experiential Workforce Readiness Model

4.1 The Project

The project was created in response to a resource need in the IT department for individuals to help with training, testing and user outreach for the university's new ERP – Human Resource/Payroll implementation. As an experienced ERP practice manager, I was asked to lead this initiative and given a minimal budget for implementation. Since it was the summer, I hired students who had taken IT classes and had availability to work with me. I was very surprised at the number of students wishing to work with one of their professors on a "real" project. The students ranged from freshmen to graduate students and a total of 5 were hired. One was given a graduate assistantship and the other four were hired at the student work pay rate. The students worked full-time for the summer and part-time through the fall semester when the system went live.

Students were given real assignments with detailed deliverables and deadlines. They provided input as to how the team would achieve tasks and open discussions happened daily to talk about how the team would divide tasks and meet deadlines. Students received feedback on a continual basis from stakeholders, full-time IT professionals, ERP external consultants and faculty to best meet the system requirements. Students also relied on peers to provide

feedback about ideas and work products. There was a level of independence for the students as they completed tasks but all projects were discussed and critiqued. The students developed a desire to exceed the expectations of the key users and demonstrate their skills and competencies. Administrators praised the initiative and quality of products created and the students were viewed as essential members of the team.

Information about the project was freely shared with the student team and they were able to apply not just their technical knowledge, but their knowledge about organizational behavior, and change. They soon began to understand how large integrated teams of developers, consultants, system administrators, users and university administrators worked together to make decisions and implement systems. The students created artifacts and had their work product reviewed by key stakeholders who provided suggestions for improvement. These improvements were incorporated into the final product that was implemented into production. An iterative feedback process became embedded in the process such that students went from novice to experience within a few iterations of review. In addition, they learned to improved their communication by observing experts and coaching. Less senior members of the student team learned from the more senior members who served as mentors and guides. The students formed a community of practice where they worked together to meet specific goals and helped each other learn throughout the process [16].

4.2 Conclusion

This experimental project helped students apply classroom knowledge in a real-world setting by reinforcing skills learned through coursework and applying them during their tenure on the project. While this project was created on-the-fly, upon reflection of my experience as the mentor on this project, I suggest a more intentional approach to identifying potential projects on the campus and mapping concrete outcomes from work on the project similar to what an academic course would expect. One aspect I would add to this experiential learning would be to incorporate student reflection into the experience. Work activities were often happening so quickly that students did not have the opportunity to reflect upon how their work activities were backed by theories and practices which they had learned in their courses. Research has shown that utilizing a structured method for reflection will help students retain their learning and reinforce the concepts through self-discovery [16].

References

[1] H. Aldewereld and E. van der Stappen. Programming, research and... coffee? an analysis of workplace activities by computing interns. In *Proceedings of the 8th Computer Science Education Research Conference*, pages 39–49, 2019. https://doi.org/10.1145/3375258.3375264.

[2] S. L. Ash and P. H. Clayton. Generating, deepening, and documenting learning: The power of critical reflection in applied learning. *Journal of Applied Learning in Higher Education*, 1:25–48, 2009.

[3] S. Blignaut and S. Trollip. Developing a taxonomy of faculty participation in asynchronous learning environments: An exploratory investigation. *Computers & Education*, 41:149–172, 2003. https://doi.org/10.1016/S0360-1315(03)00033-2.

[4] Cheryl Campanella Bracken, Leo W. Jeffres, and Kimberly A. Neuendorf. Criticism or praise? The impact of verbal versus text-only computer feedback on social presence, intrinsic motivation, and recall. *Cyberpsychology & Behavior: The Impact of the Internet, Multimedia and Virtual Reality on Behavior and Society*, June 2004. https://doi.org/10.1089/1094931041291358.

[5] S. S. Conrad and N. Dabbagh. *Student and Instructors Perceptions of Helpful Feedback for Asynchronous Online Learning: What Students Want From Instructor Feedback [Chapter]. Innovative Applications of Online Pedagogy and Course Design.* IGI Global, 2018. DOI:10.4018/978-1-5225-5466-0.ch009.

[6] P. N. Johnson-Laird. Mental models in cognitive science. *Cognitive Science*, 4(1):71–115, 1980. https://doi.org/10.1016/S0364-0213(81)80005-5.

[7] A. Kapoor and C. Gardner-McCune. Understanding cs undergraduate students' professional development through the lens of internship experiences. In *Proceedings of the 50th ACM Technical Symposium on Computer Science Education*, pages 852–858, 2019. https://doi.org/10.1145/3287324.3287408.

[8] A. Kolb and D. Kolb. Learning styles and learning spaces: A review of the multidisciplinary application of experiential learning theory in higher education. *Learning Styles and Learning: A Key to Meeting the Accountability Demands in Education*, pages 45–91, 2006.

[9] E. H. Mory. The use of informational feedback in instruction: Implications for future research. *Educational Technology Research and Development*, 40(3):5–20, 1992. https://doi.org/10.1007/BF02296839.

[10] U.S. Bureau of Labor Statistics. (n.d.). Computer and information technology occupations: Occupational outlook handbook. Retrieved July 19, 2020, from https://www.bls.gov/ooh/computer-and-information-technology/home.htm.

[11] Federal Reserve Bank of New York. (n.d.). The labor market for recent college graduates. Retrieved July 19, 2020, from https://www.newyorkfed.org/research/college-labor-market/college-labor-market_compare-majors.html.

[12] Lih-Bin Oh. Goal setting and self-regulated experiential learning in a paired internship program. In *Proceedings of the ACM Conference on Global Computing Education*, 2019. https://doi.org/10.1145/3300115.3312516.

[13] Jesse Proctor and Pedro Galicia-Almanza. What are some best practices in assessing employee performance without using performance reviews?, October 2017. Retrieved from: https://digitalcommons.ilr.cornell.edu/student/165.

[14] K. Rockwood. Should you create an internship program?, February 2020. SHRM: https://www.shrm.org/hr-today/news/hr-magazine/spring2020/pages/benefits-of-creating-an-internship-program.aspx.

[15] Patricia L. Smith and Tillman J. Ragan. An essay on experience, information, and instruction, 1995. https://eric.ed.gov/?id=ED383339.

[16] E. Wenger. *Communities of Practice: Learning, Meaning, and Identity*. Cambridge University Press, 1999.

[17] Sarah M. Zehr and Russell Korte. Student internship experiences: Learning about the workplace. *Education + Training*, 62(3):311–314, 2020. https://doi.org/10.1108/ET-11-2018-0236.

[18] B. J. Zimmerman and A. Kitsantas. Acquiring writing revision and self-regulatory skill through observation and emulation. *Journal of Educational Psychology*, 94(4):660–668, 2002. https://doi.org/10.1037/0022-0663.94.4.660.

[19] A. Zuckerman. 52 internship statistics: 2019/2020 data, trends & predictions, May 2020. CompareCamp.com: https://comparecamp.com/internship-statistics/.

Course Content as a Tool of Inclusivity for Black/African-American Women in Computing*

Edward Dillon[1], *Krystal L. Williams*[2]
[1]*Computer Science Department*
Morgan State University
Baltimore, MD 21251
{edward.dillon}@morgan.edu
[2]*Educational Leadership, Policy, Technology Studies Department*
The University of Alabama
Tuscaloosa, AL 35487
{krystal.l.williams}@ua.edu

Abstract

Promoting inclusion and increasing the representation of women in the field of Computer Science (CS) has been an ongoing initiative. When it comes to Black/African-American (AA) women, their under-representation in CS is even more disproportionate. CS is ever-evolving, but its ability to be perceived as a field with a breadth of spaces that reflects its potential to be inclusive-to-all has been a continuing challenge.

Using course content to provide spaces for under-represented groups, like Black/AA women, to express their personal interests as they develop their computational competencies can help in addressing such potential in CS. This article discusses a two-year study conducted on 51 Black-/AA female students enrolled in an introductory and/or intermediate CS course at a historically black university in the mid-Atlantic United States. A final project was administered in both courses to allow these

students to choose their own original problem to solve while showcasing their learned computational knowledge and developed programming competencies. The results revealed that 93% of the project topics chosen by these students exhibited people-centered orientations in nature. Furthermore, these outcomes reflect the potential nature for CS to be a field that can provide a breadth of spaces that reflect one's interest while also promoting inclusion.

1 Introduction

Increasing diversity and inclusion within the field of Computer Science (CS) has been an important and ongoing initiative for some time. One reason for this initiative is due to perceptions that CS is a male-centered field [6, 17], and that it is primarily suitable for individuals of Caucasian and Asian backgrounds [18]. It has been argued that such perceptions and relative stereotypes about CS can have an influential impact on overall representation [1]. When exploring the various spaces within CS, such noted perceptions and stigmas can be furthest from the truth. Moreover, the breadth of areas that can be studied, explored, and pursued in CS are continuously growing.

From a pedagogical standpoint, it is generally known that CS by nature can be challenging, especially to incoming majors. One common reason is that incoming majors tend to struggle developing ideal mental models comprised of foundational problem-solving/critical thinking skills as they learn to program [21, 4]. From an inclusion standpoint, another challenge has been the ability to showcase CS as a field with a variety of opportunities and spaces for everyone, regardless of one's personal interest, background, ethnicity, or gender.

When focusing on gender specifically, women are drastically underrepresented in CS [22]. In the case of Black/African-American women, their specific representation in CS is even lower [22]. While there has been an increased emphasis on diversity and inclusion in CS, there are a variety of critical barriers that must be addressed to help improve representation across genders with one notable barrier being unfavorable perceptions/influences [14, 17]. The nature of this article places a focus on inclusivity challenges within CS. Certain aspects of CS courses can help to create spaces for women, specifically Black/African-American women, to express their interests, and increase engagement, while showcasing their computational competencies.

2 Motivation

The motivation for this article is based on: 1) the potential for using CS course content to promote untapped spaces for increasing engagement in the

field, especially for under-represented groups, and 2) finding ways to alleviate barriers that negatively impact women, in particular Black/African-American women, in computing.

It has been noted that the type of course content used to introduce CS concepts can influence the interests of student participation in the field [7]. When it comes to under-represented groups, there have been efforts to introduce unique course content [11, 20] along with establishing complementary in-class interventions/initiatives [13, 8] to increase engagement of these particular populations as they learn CS concepts.

When placing a direct focus on women in particular, research has found that women tend to have a general interest in subject matters or fields that are people-oriented by nature [19, 5], and likewise pursue careers that exhibit more social interaction and interpersonal qualities [10, 16]. This observed trend could reflect the notion that the field of computing needs to showcase more of its interpersonal and social capabilities as a way to attract more women into the practice.

Another factor that has negatively impacted the representation of women in computing is the prevalence of gender disparities. Popular press, for instance, continuously highlights such disparities in professional settings regarding the mistreatment of women in tech companies' and failed attempts to improve working conditions for female employees [9, 12]. In addition, research has shown that in scientific fields employers are twice more likely to hire a man than a woman when equipped only with details about the candidates' physical appearance [15].

While the overall experiences and representation of women in computing and other scientific fields are critical, the specific experiences and representation of Black/AA women are also important. It has been noted that collegiate Black/AA women in computing sciences report feelings of cultural isolation and subordination while grappling with their gendered and raced identities in White-male dominated spaces [3]. In spite of current research efforts [3, 14] that address their specific under-representation in the field, more work can be done to promote their inclusion. The study that is discussed in this article seeks to better understand the interests exhibited by Black/AA women as CS majors. Guided by existing studies concerning women and occupational preferences, this study also examines the degree to which Black/AA women exhibit people-oriented interests while also showcasing their computational competencies within the context of undergraduate training. This study can inform research and practice by illustrating how purposeful pedagogical decisions around teaching and learning can help to create an environment of inclusion where under-represented groups are allowed opportunities to bridge existing interests with CS practices.

3 Methods

This study employs qualitative methods to examine how the development of CS course content in a manner that allows Black/AA female students to explore their interest and broader implication for society may help to create spaces of inclusion in computing. This study was conducted at an historically Black institution in the mid-Atlantic region of the United States during a two-year period spanning from Spring 2017 to Spring 2019.

The courses used to conduct this two-year study included *a CS2 programming course* and *an advanced programming course*. Using Python, the CS2 programming course exposed students to programming using data structures such as *file input/output and exceptions, advanced string usage, classes/OOP concepts, inheritance, polymorphism, recursion,* and *GUI programming*. The advanced programming course taught C++ programming using data structures such as *functions, arrays, structured data, file input/output, classes/OOP concepts, inheritance, linked-lists, stacks,* and *queues*.

3.1 Final Project Synopsis

The *final project* was one formal evaluation piece that was used to assess the students' ability to demonstrate proficiency in the areas outlined by the learning objectives in both courses, respectively.

Students were allowed to select an original topic to complete their final project for the course. To ensure that their topic met this criterion, they were expected to develop and defend a prospectus that centers around three questions:

- *What problem are you planning to solve, and how does this problem impact society?*
- *What data structures are you proposing to use to solve this problem?*
- *What real-world/societal impact will this proposed topic exhibit?*

During the latter semesters of the aforementioned courses (the Spring 2018 - Spring 2019 semesters), students were placed into groups for their final project in an effort to be exposed to group-based and project-based learning experiences and skills. During these particular semesters, there was an additional question that the students had to address during their prospectus regarding individual accountability:

- *How will the components of this project be divided amongst each group member?*

Based on the various data structures covered in each respective course during a semester, the students were required to show their proficiency in understanding this material and the competency in applying them effectively to solve an original problem. In order to evaluate this effectively, each student or team was required to schedule a project defense with the course instructor during the week of Finals in order to demo the outcomes of their projects, discuss the computational code developed that reflect the data structures learned in these courses, and defend their understanding and reasoning for applying the used data structures in their projects.

3.2 Data Collection & Analysis Procedures

Basic qualitative research approaches were used to analyze data from the final projects of each course. Specifically, document analysis [2] was employed to examine the type of topics that Black/AA women in these courses selected for their final project. Using existing studies concerning women and occupational preferences, this analysis also allowed the examination of whether these selected topics exhibited people-centered orientations in nature. Data from a total of 51 Black/AA women were analyzed. Aforementioned, the final project transitioned from being individual-based to group-based during the Spring 2018 semester and beyond. Therefore, some of the data and findings will also reflect a portion of these 51 students who worked either in all-female groups or mixed gender groups during the latter semesters of this study. The classifications for these women during their enrollment in these respective courses ranged from freshmen to seniors. In many cases, students who were taking these courses as seniors were non-CS majors; overall, 92% of the studied participants were CS majors. A total of 36 projects were evaluated.

For each of the Black/AA female students, their final projects were examined to determine the overall topical area and to better understand the general focus of these projects. As each of the projects were examined, small pieces of information concerning the topical area were labeled to begin to create codes that explain the overarching focus of the projects. After each of the projects was coded, similar codes were combined into categories that better represent the content areas for the various projects considered. This categorization process ultimately helped the authors to identify overarching themes that exist across the data and to develop insights about the degree of whether these projects also encompassed a person-centered focus.

4 Results

Table 1 provides insights about the themes emerging from the data concerning Black/AA women topical selection for their final projects. Based on the

document analysis, the centralized themes included the following: *Education, Entertainment, Financial, Food, Health, Sports, Travel,* and *Other*. Amongst these themes, projects concerning Education and Health were most commonly selected by the participants in this study. The next most common theme was Financial related projects. There were a few projects that were classified as Other due to their uniqueness in comparison to other topics selected. These particular projects were based on subjects such as grocery shopping, calendar planning, event prioritizing, and hair line products.

There were also 7 projects that exhibit characteristics which allowed them to be classified using multiple themes. These multi-themes are noted in each of the appropriate classifications included in Table 1 (which increased the total frequency of the project themes from 36 to 43). These projects were classified as: *Education/Financial, Education/Health, Financial/Health,* or *Financial/-Travel,* respectively.

Table 1: Centralized Project Themes and their frequencies of occurrence

Themes	**# of Projects (N=43)**
Education	12
Entertainment	1
Financial	9
Food	2
Health	12
Sports	1
Travel	2
Other	4

Based on the aforementioned research that suggests women to have a proclivity towards people-oriented environments that allow them to help others, it was important that we examined this dynamic amongst the participants to see if such an inclination manifests in topics that they selected. The document analysis revealed that 93% of the projects were people-centered in nature. When focusing on the centralized themes, subjects regarding Education, Health, and Financial were found to exhibit the most occurrences of projects that were people-centered in nature. In addition, there were relative projects that were categorized as Food, Travel, and Other. Table 2 illustrates (by theme) the project topics that were found to exhibit people-centered (People-Centered) orientations in nature.

Table 2: Occurrence of Projects Topics found to be People-Centered

Theme	**# of Projects (N=43)**	**People-Centered**
Education	12	12
Entertainment	1	0
Financial	9	9
Food	2	1
Health	12	12
Sports	1	0
Travel	2	2
Other	4	4

5 Discussion

This study revealed notable observations that could help in addressing the current interests and representation of Black/AA women, and other under-represented groups, in CS. When it comes to women's occupational interest in people-centered environments and opportunities, one explicit observation is that most of the selected projects were found to be such in nature. A more general observation relates to how Black/AA women in this study were able to connect their interests with the CS courses and the computational learning process. Another observation speaks to the ability for a CS course (or even a CS curriculum) to provide spaces via selected course content to help foster the interest of under-represented groups like Black/AA women, which also promotes innovation and inclusion.

These findings also highlight the potential to emphasize cross-disciplinary aspects of CS and how it may intersect with other fields such as business/-finance, education, health, and psychology. From a scalability standpoint, decisions employed in the noted comprehensive final projects of these courses highlight an approach that other educators in CS and related STEM fields may employ in efforts to create an inclusive classroom that allows students to explore their interests in authentic ways. In the case of women CS majors, such an approach may also be a fruitful way for them to explore the ways in which CS can be understood as a field with spaces that focuses on working with people and helping others.

5.1 Limitations

Even though the findings in this study revealed notable observations about the interests of the Black/AA women participants, there were some limitations that could have influenced its outcomes. One limitation relates to our sample

size (N=51). Since this study only reflects a small subset of Black/AA females who have ever taken CS courses, this study cannot be generalized to reflect general tendencies about Black/AA women in CS as a whole. In addition, this study was conducted at only one of many institutions who enroll Black/AA women as CS majors.

Another limitation is the mixed representation of female students who conducted a final project individually and those who did so in a group. Aforementioned, during the Spring 2018 semester and afterwards, students enrolled in these courses conducted their final projects in groups. There were some groups comprised of a mixture of both male and female students. Even though 93% of all of the projects in this study were found to be people-centered in nature, it is not entirely absolute that the female students in the mixed gender groups directly influenced the decision for their teams' selected project topics.

A final limitation relates to one of the key questions the students had to address as they developed their final project topic. As noted in the Final Project Synopsis subsection (3.1), one of these questions required them to focus on the real-world and societal aspects of their project. It is possible that this particular question could have influenced their decision to naturally select a project that was more people-centered in nature to reflect a real-world problem or scenario.

6 Conclusion

Creating effective initiatives that promote inclusion and increased representation of women in CS is a definite necessity. Using course content for such purposes can serve as one impactful initiative. Employing relative interventions towards Black/AA women in CS, likewise, could aid in their retention and enable their potential contributions to an ever-evolving field. This article explored the experience of Black/AA women CS majors and the types of interests they can exhibit as they showcase their computational skill sets in given CS courses. Such in-classroom practices via course content can provide immediate spaces to promote and engage interest from this particular group and other under-represented groups alike. Moreover, this type of initiative can serve as an interventive strategy for strengthening the correlations between CS and these groups' personal interests and potential contributions to the field.

References

[1] Sylvia Beyer. Why are women underrepresented in computer science? gender differences in stereotypes, self-efficacy, values, and interests and

predictors of future cs course-taking and grades. *Computer Science Education*, 24(2-3):153–192, 2014.

[2] Glenn A Bowen et al. Document analysis as a qualitative research method. *Qualitative research journal*, 9(2):27, 2009.

[3] L. J. Charleston, P. L. George, J. F. Jackson, J. Berhanu, and M. H. Amechi. Navigating underrepresented stem spaces: Experiences of black women in us computing science higher education programs who actualize success. *Journal of Diversity in Higher Education*, 7(3):166–176, 2014.

[4] Martha E Crosby and Jan Stelovsky. How do we read algorithms? a case study. *Computer*, 23(1):25–35, 1990.

[5] Jacquelynne S Eccles. *Where Are All the Women? Gender Differences in Participation in Physical Science and Engineering.* American Psychological Association, 2007.

[6] Jacquelynne S Eccles, Janis E Jacobs, and Rena D Harold. Gender role stereotypes, expectancy effects, and parents' socialization of gender differences. *Journal of social issues*, 46(2):183–201, 1990.

[7] Joseph V Elarde and Fatt-Fei Chong. Introductory computing course content: educator and student perspectives. In *Proceedings of the 2011 conference on Information technology education*, pages 55–60, 2011.

[8] Susan Horwitz, Susan H Rodger, Maureen Biggers, David Binkley, C Kolin Frantz, Dawn Gundermann, Susanne Hambrusch, Steven Huss-Lederman, Ethan Munson, Barbara Ryder, et al. Using peer-led team learning to increase participation and success of under-represented groups in introductory computer science. *ACM SIGCSE Bulletin*, 41(1):163–167, 2009.

[9] Sheelah Kolhatkar. The tech industry's gender-discrimination problem. *New Yorker*, 20, 2017.

[10] Richard Lippa. Gender-related individual differences and the structure of vocational interests: The importance of the people–things dimension. *Journal of personality and social psychology*, 74(4):996, 1998.

[11] Marlon Mejias, Ketly Jean-Pierre, Legand Burge, and Gloria Washington. Culturally relevant cs pedagogy-theory & practice. In *2018 Research on Equity and Sustained Participation in Engineering, Computing, and Technology (RESPECT)*, pages 1–5. IEEE, 2018.

[12] Liza Mundy. Why is silicon valley so awful to women. *The Atlantic*, 319:60–73, 2017.

[13] Tia Newhall, Lisa Meeden, Andrew Danner, Ameet Soni, Frances Ruiz, and Richard Wicentowski. A support program for introductory cs courses that improves student performance and retains students from underrepresented groups. In *Proceedings of the 45th ACM technical symposium on Computer science education*, pages 433–438, 2014.

[14] Yolanda A Rankin, Jakita O Thomas, and India Irish. Food for thought: Supporting african american women's computational algorithmic thinking in an intro cs course. In *Proceedings of the 50th ACM Technical Symposium on Computer Science Education*, pages 641–646, 2019.

[15] Ernesto Reuben, Paola Sapienza, and Luigi Zingales. How stereotypes impair women's careers in science. *Proceedings of the National Academy of Sciences*, 111(12):4403–4408, 2014.

[16] Els Rommes, Geertjan Overbeek, Ron Scholte, Rutger Engels, and Raymond De Kemp. 'i'm not interested in computers': Gender-based occupational choices of adolescents. *Information, Community and Society*, 10(3):299–319, 2007.

[17] Myra Sadker and David Sadker. *Failing at fairness: How America's schools cheat girls*. Simon and Schuster, 2010.

[18] Jocelyn Steinke, Maria Knight Lapinski, Nikki Crocker, Aletta Zietsman-Thomas, Yaschica Williams, Stephanie Higdon Evergreen, and Sarvani Kuchibhotla. Assessing media influences on middle school–aged children's perceptions of women in science using the draw-a-scientist test (dast). *Science Communication*, 29(1):35–64, 2007.

[19] Rong Su and James Rounds. All stem fields are not created equal: People and things interests explain gender disparities across stem fields. *Frontiers in psychology*, 6:189, 2015.

[20] Briana Lowe Wellman, James Davis, and Monica Anderson. Alice and robotics in introductory cs courses. In *the Fifth Richard Tapia celebration of diversity in computing conference: intellect, initiatives, insight, and innovations*, pages 98–102, 2009.

[21] Susan Wiedenbeck, Vennila Ramalingam, Suseela Sarasamma, and Cynthia L Corritore. A comparison of the comprehension of object-oriented and procedural programs by novice programmers. *Interacting with Computers*, 11(3):255–282, 1999.

[22] S Zweben, B Bizot, Computing Research Association, et al. Taulbee survey: Undergrad enrollment continues upward; doctoral degree production declines but doctoral enrollment rises (washington, dc, 2018), 2018.

Fileless Malware and Programmatic Method of Detection*

Student Paper Abstract

Pipop Nuangpookka, Zelalem Mengistu, Ghada Bafail
School of Business and Technology
Marymount University
Arlington, VA 22201
{pnuangpo, zwm26248, gab92599}@marymount.edu

Fileless malware lives unnoticeably in a computer system's main memory and executes its malicious processes freely. Although the anti-malware industry knows about this sophisticated form of attack and provides protective capabilities to detect such threats, Fileless Malware is violently relentless which makes the detection process so difficult, and yet challenging. The moment cybersecurity professionals came out with a robust and effective countermeasure, malware threat landscapes were also evolved to avoid detection and increasingly becoming more sophisticated. Furthermore, since fileless malwares does not use or reside on the file system, they could not be easily detected on any signature-aware antivirus detection system. Due to this fact, fileless malware attack vector is disastrous for any organization (government, private). It is reliance on an existing operating system and approved tools makes the attack too subtle. Despite the illusive nature of fileless malware, cybersecurity professionals use forensic tools to trace the attacker, which most, if not all of the time could be unsuccessful as the attackers might implement an anti-forensic tools to evade detection and or traces. This research work/ experiment aims at compromising a computer system by executing malicious scripts or payloads on a web browser remotely using JavaScript without requiring an installation of a file on the targeted computer system. The method of the attack is to take advantage of the feature of a Web Interface's Application Programming Interface (API), ActiveXObject, provided by Microsoft Corporation. Accordingly, the potential exploitation may only be possible to the users accessing websites by using Internet Explorer web browsers. Extensible + Apache + MariaDB + PHP + Perl (XAMPP) webserver were installed on a Microsoft Windows 8

machine to initiate fileless malware processes and a Java Development Kit and Java Runtime Environment 8 were the programming language for the implementation of a method of detection. Finally, we were also able to replicate the execution of legitimate CMD commands on a remote host through a malicious JavaScript. Hence, it only takes one mistake, or one click from an unaware user to get exploited, unintentionally. Fortunately, we were able to implement a monitoring process to detect such a threat using a Java code which executes the CMD command programmatically every three seconds using the technique of Multi-Thread Programming (i.e., Ability to control sequence of time for execution). In other words, when the ActiveXObject is active, it will detect the malicious CMD process. Such information would be valuable to an observer who should be able to react to this suspicious CMD activity promptly.

Maturity of the Malware Marketplace a Disturbing Trend using Probability Density Function*

Student Paper Abstract

Ana Valentin and Thomas Kim
School of Business and Technology
Marymount University
Arlington, VA 22201
ahv22211@marymount.edu

The centralization of malware variance signals an increase in the maturity of the malware marketplace and the operations that produce them. Since the early 1970s until today, malware has exponentially increased in volume and dispersion, but in early 2017 the trend showed a remarkable shift. For the first time in nearly 50 years AV-Test trend showed a decreased variance in malware file variance had declined. This paper examined whether the probability density function explain the diminishing of new malware specimens reported in malware trend. the researchers found that probability density function could predicted the likelihood of the diminishing of new malware specimens published by AV-Test in 2017. The Gamma-Poisson Distribution is the model used for describing randomly occurring of malware specimens reported and determine the probability of a number of malware specimens reported in 10-years period or finding the probability of waiting some time until the next number of malware specimens reported. The study concluded that the probability density function predicted for an expected likelihood of 0.25, 0.10 and 0.01 a descending trends of new malware specimens (in million) between 2007 and 2013 was closer to 0 in 2014.

Performance Analysis of the Lamp Stack Compared to Its Variants in a Single Page Web Application Environment*

Student Paper Abstract

Robert Kohlbus
Frostburg State University
Frostburg, MD 21532

For over two decades, the LAMP stack has been one of the most popular open-source application stacks for web development. As the web ecosystem has evolved and new open source software has been made available, is the LAMP stack still viable? This paper analyzes the performance of the classic LAMP stack when compared to its variants in a single page web application environment.

Credit Card Fraud Detection: An Evaluation of SMOTE Resampling and Machine Learning Model Performance*

Student Paper Abstract

Ran Xia and Faleh Alshameri
School of Business and Technology
Marymount University
Arlington, VA 22201
{r0x28181, falshame}@marymount.edu

Credit card fraud has been a noted security issue that requires financial organizations to improve their fraud detection system continuously. In most cases, a credit transaction dataset is expected to have a significantly larger number of normal transactions than fraud transactions. Therefore, the accuracy of a fraud detection system depends on building a model that can adequately handle such an imbalanced dataset. The purpose of this paper is to explore one of the techniques of dataset rebalancing, the Synthetic Minority Oversampling Technique (SMOTE). To evaluate the effects of this technique on model training, we selected four basic classification algorithms, Complement Naïve Bayes (CNB), K Nearest Neighbor (KNN), Random Forest, and Support Vector Machine (SVM). We then compared the performances of the four models trained on the rebalanced and original dataset using the Area Under Precision-Recall Curve (AUPRC) plots.

Supporting Underrepresented Groups in STEM During Uncertain Times: A Case for Transfer Students from Rural SW PA

Panel Discussion

Natalya Bromall, Karen Paullet,
Fred Kohun, Diane Igoche
Robert Morris University
{bromall, paullet, Kohun, igoche}@rmu.edu

The panelists have been awarded a National Science Foundation grant to broaden the participation of underrepresented student groups in STEM. The goal of the program is to attract students who have an aptitude for the STEM subjects, but for various reasons may not be able to complete their undergraduate degrees. They may face lack of support and mentorship from their families and peers, insufficient income, psychological barriers, and many other difficulties.

The panel discussion will focus on the efforts it has made to continuously provide support to Underrepresented Students during the onslaught of the COVID-19 pandemic and in the semesters after the 2020 nationwide lockdown. The panelists have received an award from the National Science Foundation to broaden the participation of underrepresented groups in STEM and they will be encouraging conversation around creating inclusive avenues for support for their students. The challenges colleges and universities are facing with navigating what might be a new normal cannot overshadow the gap that is created when students in this group are left to make meaning of their academic journeys as well as the hurdles that they will encounter in an ever changing job market.

Partitioned-Hill Cryptosystems: A STEM Lab for AP CSA*

Nifty Assignment

John Pais
Mathematics and Computer Science
Ladue Horton Watkins High School
pais.john@gmail.com

A Partitioned-Hill Cryptosystem uses a sequence of randomly generated matrix affine ciphers, AX + B, of different random dimensions, that are determined by a randomly generated partition of the plaintext string.

A Partitioned-Hill random sequence cipher is a symmetric cipher comprised of random blocks, which by design has good intrinsic security due to the difficulty of both searching all partitions of a plaintext string and searching all invertible (mod k) matrices of dimension n >= 3. In practice, the plaintext string is first partitioned into superblocks of a fixed size, e.g. m = 15, for which partition numbers are easily computed and used to select a random partition of each superblock. Finally, random matrix affine block ciphers are generated corresponding to the random partition of each superblock.

This work is an especially useful example of significant STEM integration that combines current and important subject matter content from linear algebra, number theory, computer science, cryptology, and cybersecurity. STEM students seem to respond well to computer science labs that provide a learning environment containing both relevant and challenging programming problems that motivate and necessitate the invention and use of complex data structures and algorithms.

Nifty Assignment in Computer Networking Laboratory*

Nifty Assignment

Wen-Jung Hsin
Computer Science and Information Systems
Park University
Parkville, MO 64152
wen.hsin@park.edu

Park University Faculty Center for Innovation obtained a grant through the Ewing Marion Kauffman Foundation, allowing the author of this proposal and 49 other instructors at the University to participate in a 32-week long course entitled Effective Teaching Practice. The course was offered through the Association of College and University Educators (ACUE) in conjunction with American Council on Education (ACE). One important concept that the author learned through the course is called "Transparent Assignment." The ACUE course module explains that "a transparent assignment is an assignment that aligns with the course learning outcomes." To develop a transparent assignment, one starts with a course learning outcome, goes through a 3-step process to create a teaching planner, and finally applies a transparent assignment template to create an actual assignment.

At the time of taking the ACUE course, the author was teaching a Computer Networking Laboratory course. So she used this networking course to practice and utilize the effective teaching practices she learned from the ACUE course. She created a transparent assignment for the Computer Networking Laboratory course and gave it to the students as a capstone project at the end of the semester.

This presentation will describe what a transparent assignment is, and show the steps of how to develop a transparent assignment using the Computer Networking Laboratory course as an example.

Containerizing CS Learning Environments*

Poster Abstract

Linh B. Ngo, Richard Burns, Si Chen
Computer Science Department
West Chester University of Pennsylvania
West Chester, PA 19383
{lngo, rburns, schen}@wcupa.edu

This work describes an attempt to create a containerized platform to support various computing resources for hands-on activities throughout the CS curriculum. Via containerization, students will have the opportunity to become engaged regardless of their personal computing devices. The baseline container images will be lightweight, and the number of images is minimized to reduce storage impacts on students taking multiple courses. Similarly, efforts will be made to ensure additional image layers can be reused across courses. Our platform must work with all three majors operating systems, be lightweight, and preferably be portable across personal and remote computing environments. It can also solve challenges in making sure hands-on lectures on various attacks remain up-to-date regardless of how the related system libraries are changed over time. The poster will have four major areas. The first area showcases existing courses with containerized components, broken down to individual image layers. It provides information on motivations and preliminary successes of our various courses that utilize containers in individual lectures, particularly with complex exercises. The second area, where a common platform is envisioned, is to create a merging common low-level container layers, with customized top-layers for the majority of junior and senior level courses at the authors' institution. The third area includes large bullet points identifying potential faculty's adoption challenges and students' usage challenges. The fourth area, connected to various courses in the second area, describes possible assessments on the effectiveness of containerization at different courses. We envision that this platform, shared among our CS courses, would greatly help instructors to share experience in incorporating and deploying new learning components and addressing common technical issues. Additional benefits of containerization also include the possibility of deploying large-scale computing environment for education, whether on local, federal, or industry cloud resources.

3D Printed Models for Teaching Data Structures*

Poster Abstract

Samah Senbel
School of Computer Science and Engineering
Sacred Heart University
Fairfield, CT 06825
senbels@sacredheart.edu

The data structures course is a fundamental course in Computer Science education. It describes and compares the different structures that can be used to store data in memory. The course typically covers arrays, stacks, queues, hash tables, trees, and graphs at least. The emphasis is on the different features of each and when it is most suitable to use it. And therefore, students need to visualize the data structures in memory and how they are used. This is traditionally done by showing diagrams and animations of the data structure being used. For example, a stack having data pushed and popped from it, or a binary tree changing shape with data insertion and removals.

We designed a set of 3D printed models that represents the different data structures. They enable students to better visualize them in memory, and to compare the structure of consecutive and non-consecutive memory allocation. This is particularly useful for explaining data structures with pointers such as linked lists and trees. Students were involved in the design of the 3D structures and the actual printing process. Four basic data structures were designed and printed during the spring 2020 semester: arrays, stacks, queues and linked lists. The data and memory addresses was represented as cubes with numbers printed on them that are inserted into the 3D frames representing the memory locations.

The students found the structures very useful and stated so in the course evaluations. The next challenge is the 3D design for a binary tree that can support the addition and removal of nodes. Several ideas were discussed using arrows, ropes and hooks, and the implementation of them will be built and compared in the fall 2020 semester. A complete data structure demo set will be published as an instruction aid for data structures courses in high schools and colleges.

Computational Thinking for Computer Science Majors: An Introduction to CS Education Career Pathways*

Poster Abstract

Alan C. Jamieson and Lindsay H. Jamieson
Department of Mathematics and Computer Science
St. Mary's College of Maryland
St. Mary's City, MD 20686
{acjamieson,lhjamieson}@smcm.edu

In general, computer science majors do not consider K-12 education as a career pathway. A potential reason is a lack of introduction to the career path during their undergraduate careers. In this project, supported through a grant from the Maryland Center for Computing Education (cs4md.org), we developed a pilot computational thinking course blending computer science majors and pre-service teachers. The course focused on describing and integrating computational thinking in a way that would translate to K-12 classrooms while utilizing non-programming based computational tools to demonstrate these ideas. Participants were able to put these ideas into practice in K-8 classrooms during the course and reflect on how the lessons were received. We discuss the implementation of the course, field placement, motivations, and our initial analysis of survey data, including impacts on attitudes on education as a career choice for computer science majors.

A Multi-Cloud Environment for Teaching Relational Database Services*

Poster Abstract

Weidong Liao
Department of Computer Science, Mathematics, and Engineering
College of STEM
Shepherd University
Shepherdstown, WV 25443
wliao@shepherd.edu

In today's business world, it has become standard practice to use cloud computing to host data and application alike. Instead of equipping with server rooms with expensive yet soon becoming outdated server computers, and a number of system and database administrators, now companies may "rent" the hosting services for database systems and applications. Cloud computing has its advantage being more cost-effective, just thinking about the average annual salary and benefits for database or system administrators. The other benefit of cloud computing is elastic. Companies can easily upgrade its cloud service level during peak business times, and downgrade the cloud service level afterwards.

On the other hand, cloud computing has also brought several concerns for business users. Security of data, reliability of application services, and availability of data and applications are among a few of these concerns. As cloud computing has infused into daily businesses, it has become an arguable point whether it is safer to store your business data compared to in-house data storage. People have been stating that "data is new currency". As a result, storing data in storage cloud would be like depositing cash into banks, and it should be safer than storing data onto in-house data storage. Nevertheless, the reliability and availability of data and application services over the cloud are still the major concern for companies who are exploring serverless computing environment for their business.

A multi-cloud database is a particular distributed database environment that makes use of multiple database storage engines over more than one cloud

computing platforms, aiming to providing reliable and efficient data access services to achieve an organization's datacenter goal. It combines data storage services over multiple, private or public, cloud computing platforms, which may augment benefits of data services from a single cloud computing platform.

Since cloud computing has become a dominant platform in providing database storage services for modern applications, college graduates equipped with cloud computing knowledge and skills gain competitive edge over job market. It is therefore critical for us as computer science educators to offer students opportunities to work on projects in this area to understand how modern computing platforms work and to stay current and front in this dynamic field of information technology.

In this poster, we present our investigation and findings over multi-cloud database systems, with focus on projects that undergraduate students could leverage to understand the fundamentals of cloud computing, its advantage and tradeoff, and its programming paradigm. An application framework for Java developers to access data from multi-cloud database systems is also described.

Introducing Computational Thinking to Pre-service Teachers*

Poster Abstract

Jiang Li, Paulette Shockey, Jennifer Cuddapah,
Christy Graybeal, Anthony Williams
Computer Science and Information Technology
Hood College
Frederick, MD 21701
lij@hood.edu

Computational, logic thinking and problem-solving skills are extremely important for students' success in the future. This poster describes a collaborative project that was conducted to promote K-8 Computer Science Education among in-service and pre-service teachers. More than 40 pre-service and in-service teachers participated in a learning experience designed to address the K-12 Computer Science Framework and Maryland's K-12 Computer Science Standards. The collaboration was designed to facilitate participants' learning about and application of foundational principles of computer science and computational thinking into K-8 STEM curriculum and teaching. Participants explored hard/software platforms and used open source sites such as Scratch, Code.org and Code Academy. Participants envisioned how activities apply to K-8 classrooms and worked in pairs or groups to design a problem-based project for students. Project evaluation included formative and summative assessments to examine changes in content and pedagogical knowledge.

Acknowledgement

Our professional development events were sponsored by Maryland Preservice Computer Science Teacher Education Program, we appreciate Maryland Center for Computing Education team for allowing us to provide such great opportunities to our pre-service teachers in CS education.

Low-Code/No-Code Software Development Platforms and their Uses in Computer Science and Information Technology Education*

Faculty Poster Abstract

Weidong Liao and Osman Guzide
Department of Computer Science, Mathematics, and Engineering
Shepherd University
Shepherdstown, WV 25443
wliao, oguzide@shepherd.edu

Low Code/No-Code (LCNC) software development has gained its share and popularity in today's software development market, especially in cloud-based application development. In addition to companies specializing in LCNC software development such as Appian, OutSystems and Mendix, major cloud service providers, such as Amazon, Microsoft and Google, have all released their LCNC development platforms.

In this poster, we describe our experience in using LCNC platforms in Computer Science (CS) and Information Technology (IT) classrooms, and our observations and lessons learned from integrating LCNC concepts and using LCNC platforms in teaching a variety of CS and IT concepts. It is evident that students must equip with solid fundamental concepts in order to use LCNC platforms effectively and efficiently. In addition, we believe there will always be need for traditional programmers. LCNC and traditional programming that has always been taught in CS/IT curriculum may very well complement each other in future software development.

Towards Understanding Privacy Trade-Off in an Epidemic*

Poster Abstract

Sajedul Talukder
Mathematics and Computer Science
Edinboro University, Edinboro, PA 16444
stalukder@edinboro.edu

Although the COVID-19 is still as complicated as ever, it is a significant job for private networks across the globe to gather and share data in the light of the battle against coronavirus. The scale and severity of the disease are not rare, but they seem to be near it. Consequently, drastic steps to remedy the situation seem to be the rule in a very short period of time. As the coronavirus pandemic spread across the world, countries developed massively controlled networks to track virus transmission and forced governments all over the world to take into account trade in health and protection for millions of people. In certain cases, the whole population has been intensively monitored and diagnostic records from those who are infected with the virus are usually distributed around organizations and nations. While in many countries innovative approaches have been introduced to counter this, privacy advocates are concerned that technology would eventually erode privacy, while regulators and supporters are concerned about the type of effect that this may have.

In our research, we found that the best way is to strike the right balance. Systematic study on different data security problems at different fronts during the pandemic shows that multiple privacy strategies, including privacy protection measures, aggregated anonymized data and collaborative data for software and device designers would help them reliably exploit and stay effective with data from third parties without violating user privacy or confidentiality. The cases related to the ability of public authorities to intervene with the fundamental right to privacy in the interests of national security or public safety have consistently shown the prospect of achieving a fair balance if the global scientific community, and government leaders make a concerted effort to standardize Privacy Enhancing Technologies (PETs), such as ZKProof, to encourage broader adoption.

Privacy and Security Vulnerabilities in Health Care Infrastructure Mobile Technology*

Poster Abstract

Sajedul Talukder
Mathematics and Computer Science
Edinboro University, Edinboro, PA 16444
stalukder@edinboro.edu

The intense penetration of mobile devices into our everyday lives has fundamentally changed how we interact with each other in any sector. It also refers to the healthcare sector, in which technology is becoming increasingly relevant in nearly all areas of the industry. The inventing of modern software applications and digital advances have now allowed patients to view medical records using mobile apps, check up on their main alerts, navigate and organize healthcare, and execute a large variety of activities more conveniently. Mobile apps and technology provide a great deal of benefits for consumers, while a range of problems in terms of health, usability, and risks are often known. Health care companies are gradually seeking to upgrade their aging technologies to meet these concerns by incorporating the use of mobile apps. The proliferation of mobile apps and technologies also provides the company with new challenges and dangers. We find Epic Rover as a test case, a smartphone application described as a potential alternative for the control of electronic medical devices.

In our research, we consider a safety team including CISO, the technology consultant, technology engineer, and chief enforcement officer to evaluate a suitable mobile application for a fictional health agency, as well as to assess whether or not its use is worth the risk to that organization. We discuss data that the security department is looking to determine and thoroughly assess the vulnerable circumstances and threats posed by mobile devices in the health sector prior to the launch of the latest technologies using Epic Rover mobile Software, an advanced Product from Epic Systems, a leading vendor of Health Care Technology, and propose risk management and mitigation strategies while ensuring compliance with regulatory requirements.

Benchmarking the Performance of RESTful Applications Implemented in Spring Boot Java and MS.Net Core*

Poster Abstract

Hardeep Kaur Dhalla
Computing and New Media Technologies
University of Wisconsin-Stevens Point
Stevens Point, WI 54481
hdhalla@uwsp.edu

RESTful is stateless, lightweight, and predominant architectural style to design and develop web services/applications. This paper aims to benchmark the performance of RESTful web applications implemented using two different technologies: Spring Boot Java and MS.Net Core. Both implementations were developed using the same business use case to provide the basic four Create, Read, Update, and Delete (CRUD) operations. Apache JMeter 5.2.1, an automated java-based performance testing tool, was used to create virtual users to load both the applications at regular intervals. The comprehensive experimental results are presented at the end of the paper.

Parsing Performance of Native JSON Libraries in Java, MS.Net Core, and Python: A Comparative Study*

Poster Abstract

Hardeep Kaur Dhalla
Computing and New Media Technologies
University of Wisconsin-Stevens Point
Stevens Point, WI 54481
hdhalla@uwsp.edu

The emergence of JSON as a popular data interchange format has motivated the software industry to provide support for JSON in their native environment. As a result, JSON is now supported in most of the software solutions. The performance of JSON parsing, however, varies with internal implementation of a JSON parser. In this research paper, native JSON parsers in 3 programming platforms Java, MS.Net core, and Python have been studied in order to compare their parsing performance. The innermost keyvalue pair was accessed from the JSON data with varying degree of depth as the criteria to evaluate and analyze the parsing efficiency of JSON parsers in terms of parsing speed. The experimental results are discussed and shared at the end of this research paper.

Lesson Plan: An Interdisciplinary Approach to Teaching Cyber Warfare Concepts*

Poster Abstract

Donna M. Schaeffer[1] and Patrick C. Olson[2]
[1]Marymount University
Arlington, VA 22207
donna.schaeffer@marymount.edu
[2]National University
Quantico, VA 22134
polson@nu.edu

This poster supports the idea that Cybersecurity, and thus cyber warfare are interdisciplinary topics. Based on the concepts that are covered from course descriptions on cyber warfare at various levels of study and from different disciplines, including computer science, criminal justice, and international relations the poster provides support for a single lesson plan on cyber warfare that includes learning competencies, recommended current materials, and easy to implement classroom activities. The lesson plan can be modified as needed for introducing the concepts of cyber warfare across disciplines.

Robotics-Based Creative Expression for Middle/High School Female Students*

Faculty Poster Abstract

Yanxia Jia[1], Teresa Ontiveros[1], Maya Sierra[1], Thach Phung[1], Lily Liang[2]
[1]Department of Computer Science and Mathematics
Arcadia University, Glenside, PA 19038
{jiay,tontiveros,msierra_02,tphung}@arcadia.edu
[2]Department of Computer Science and Information Technology
University of District of Columbia
4200 Connecticut Ave NW, Washington, DC, 20008
lLiang@udc.edu

Innovation in computing require talents from diverse backgrounds. Given the lack of female participation in computer science, we propose to integrate artistic practice, such as choreography design and video production, into educational Robotics. Our goal is to create computing experience that can relate to and engage female students, and to encourage them to use computer science as a tool for self-expression. Based on PLEN:bit, a unique robotics device, we created lesson plans based on this idea and offered teaching sessions at four schools with diverse ethnic populations. Our study shows that the proposed method is effective in helping students learn basic programming concepts. More importantly, it increases middle/high school female students' interest in computer science and leads to a positive change of female students' attitude toward programming.

Artificial Intelligence Operated Data Warehouse*

Student Poster Abstract

Joseph Cvetovich, Harrison Linn, Kaylea Daigle,
Phil Huddleston, ABM Rezbaul Islam
Department of Computer Science
Sam Houston State University, Huntsville, TX 77340
jnc037@shsu.edu

In this research, we developed a multilayered inventory management software which pledges to revolutionize how inventories are sorted and maintained. However, current inventory management software fails to analyze for optimal methods for how and where inventory stock should be stored. Instead, many inventories solely act as a tool by which users manually maintain and allocate parts. Thus, the heavy burden calling for frequent rearrangement of parts, reflected by supply and demand falls onto employees and management.

Systematically, our software utilizes sophisticated AI (Artificial Intelligence) which interprets part data such as type or dimensions. It then analyzes sales history and other data to optimize the inventory. This method is achievable due to various channels which simulate and virtually represent an existing storage arrangement within a department. By knowing the available space, the size of parts, the history of sales and other data, the software will be able to observe best practices for part arrangement. In addition, proposed software accounts for frequency, sale association, and part dimensions without requiring our clients to change their existing storage facilities. Nonetheless, it is crucial to strengthen the available tools for inventory management to drive the management process rather than simply supporting it. Utilization of this software will improve efficiency, specific to relay times in part retrieval, speed of stocking new parts, improvement of shelf utilization, and ultimately adapting seamlessly to growing or changing inventories. Through thoroughly understanding the principles and guidelines for data warehousing, this software effectively meets the demand within the market for data mining tools specific to achieving inventory management objectives

MyHealthChart Mobile App: Gives People Control and Access to Their Medical Records*

Student Poster Abstract

Jessica Byrd, Samuel McManus,
ABM Rezbaul Islam, N.Karpoor Shashidhar
Department of Computer Science
Sam Houston State University, Huntsville, TX 77340
jnb040@shsu.edu

MyHealthChart is a mobile application that is specifically designed for families, elderly people, caregivers, and people who are chronically ill or have longterm illnesses. With a focus on the seven top chronic diseases in America per the CDC, our goal is to give people a new option to store and access their medical information with the touch of a button in a secure digital format. To the best of our knowledge, this is a novel application of its kind. However, MyHealthChart is an organizational tool used to centralize all of a user's medical information in one place, complete with an interface designed to meet the needs of people with disabilities by including large buttons and a color-blind friendly design. Each user can store information about their doctors, appointments, prescriptions, and vaccines with multi-user capabilities to extend usage to the whole family. This mobile application has many more features, such as reminding users of upcoming appointments or when to take their medicine, and the ability to attach a picture, such as an X-ray or an injury, inside of a medical note. MyHealthChart can also produce a range of medical reports based on previously reported data and allows a user to generate daily health charts based on their specific needs. MyHealthChart eliminates the need for patients to pay for medical records from their doctor's office, recall medical details from memory, or search through a filing cabinet at home for specific medical information by replacing the traditional paper filing system. Lastly, we strongly believe, MyHealthChart will revolutionize personal medical record keeping and serve as a double-checking mechanism for medical information documented by doctors and caregivers.

An Interactive Mobile Application for Skin Clinic*

Student Poster Abstract

Khalid Noman, Mohamed Barodi, Ahmed Noman,
Carilyn Santisteban, ABM Rezbaul Islam
Department of Computer Science
Sam Houston State University, Huntsville, TX 77340
ksn007@shsu.edu

A mobile application is developed and implemented to streamline communication between a clinic and its customers. This application will allow the clinic to relay some vital information about the clinic, its services and will also give the customers a chance to interact with the clinic beyond operational hours. The primary objective of the application is to provide a communication service for a dermatology and laser clinic. It will enable communication with their customers over a medium that is not direct contact or social media based. This type of application is commonly used in the USA but other countries still need this type of application. This application is currently used by Skin Essentials Dermatology and Laser clinic, Saudi Arabia who was the client for this project.

In terms of service, the mobile application has the ability to provide functionality for the customers, doctors and hospital administrators. The customers can get information about the services provided, information regarding the doctors and their specialties. Besides, this application is also equipped with a registration process which will eliminate long waiting time upon arrival for both customers and respective doctors. It will be also helpful to reduce number of people gathering in a space, especially in a situation like ongoing COVID-19 pandemic. Nonetheless, the appointment request should eliminate the need to directly contact the clinic to request an appointment. It will exclude the possible waiting time and any technical difficulties that arises during an appointment scheduling process. The feedback function will give customers the capability to evaluate their satisfaction about the clinic or their visit. Currently, these services can be modified by the system administrator by editing

in the database, but a function in the computer application is currently being developed to give the clinic the simplest method of editing their services. The applications can help the clinic in relaying information to their customer base with a single application that the customers can use on their android mobile devices. The mobile application will allow the customers of the clinic to relay information back to the clinic. The mobile application is developed for the general customer that requires a simple interface and efficient usability. The customer and the client satisfactory feedback ensures the easy accessibility, usability and proficiency in error handling for this application.

Gear Shifting: Back to the Basics Phase 1*

Student Poster Abstract

Meghan Murphy
Frostburg State University
Frostburg, MD 21532
memurphy@shsu.edu

A gear shifting study to improve child biking was being conducted by Meghan Murphy, a sophomore at Frostburg State University. The problem is the difficulty of comprehending why gears are needed. Children do not understand the complexity of gear shifting. At young ages, children do not have enough biking experience and knowledge to transition between gears properly. According to the Journal of Sport & Exercise Psychology, children at younger ages of 5-7 years do not use a strategy while riding their bike compared to older children of 8-10 years of age. The second part of their study showed that all age groups made fewer errors when given a strategy for riding (Liu, T., & Jensen, J. L., 2007). Also, Cannoni stated that mechanical reasoning develops around the ages of 7 and 8. At an average age of 8 years, the child can state how a bike works (Cannoni, Elenora, et al., 2018). It could be reasoned that at these young ages, children are still trying to understand the bike and cannot fully comprehend gear shifting. Gear shifting comes with experiencing and understanding the terrain. Children are not developed enough to completely apprehend this topic, so they need help learning this fundamental bike skill.

The purpose of this study was to better understand the process of learning how to shift gears at a young age. A survey was administered to collect data. This study helped gained insight into useful aspects which will help develop a gear shifting app. For the following phase, the goal is to create an app that could help children ages 7 years old and older to know when to change gears. The final product will assist children in learning to shift gears properly and efficiently.

The poster presentation will include my abstract, purpose, example question of my survey, data gathered from the survey, results of the survey, and

future app creation plans. The poster will explain my reasoning to conduct this research followed by the survey that I produced to get feedback from bikers. The study was used to determine the importance of gear shifting and obtain information about average cadence and timing to switch gears. This information will be used to create the app. The poster will contain features of the planned app as well as a mockup. The poster presentation will give the viewer an overall idea of my research on gear shifting.

Homeostasis and Machine Learning in the Biology Classroom*

Student Poster Abstract

Judith Lucas-Odom
Drexel University
Philadelphia, PA 19104

Pediatric Cardiology and Machine Learning technologies are helping young patients understand their symptoms, while helping them to improve their condition. Understanding how some students of ethnic backgrounds can have pre-existing conditions that may progress to other diseases as they grow older (i.e. heart disease, high blood pressure, and diabetes) is a major concern. These students in some cases have poorer health standards due primarily to their economic status. The lack of nutritional choices in their diet correlates to higher risk factors and ultimately high blood pressure or diabetes. Students without these factors need to understand that maintaining a healthy lifestyle will decrease the likelihood of developing these diseases during their lifetime. The objective of this poster is first, to help students and young adults change their understanding of what causes heart disease and diabetes. Second, to introduce them to computer science concepts that will help them analyze risk factors that lead to heart disease and diabetes, while tracking changes that can be made once they are aware. Third, students will use machine learning to collect pertinent data to analyze and compare. This analysis will help them monitor their unhealthy lifestyle while encouraging healthier alternative options. The data that is collected includes their Body Mass Index, Blood Pressure, diet, and exercise amount. Using their coding skills, students will design projects to empower other young adults and children to make healthier lifestyle changes. Lastly, this tool will be used to help medical personnel better monitor their patient's health through the development of a website to collect and analyze the data collected.

Where Did the Time Go? An Android-Based Phone Time Management App*

Student Poster Abstract

John Viaud, Vitali Surmach, Bilal Abdulmajid, Yanxia Jia
Department of Computer Science and Mathematics
Arcadia University, Glenside, PA 19038
{jviaud,vsurmach,babdulmajid,jiay}@arcadia.edu

Recent data shows most people, on average, spend 3 hours and 15 minutes on our phones, and the top 20% of smartphone users have daily screen time in excess of 4.5 hours. During COVID-19, screen time surges due to lockdown and school closures, and that is taking a serious toll on our productivity and mental and physical well-being. Recent research shows that after 1 hour/day of use, more hours of daily screen time were associated with lower psychological well-being, including less curiosity, lower self-control, more distractibility, more difficulty making friends, less emotional stability, being more difficult to care for, and inability to finish tasks. Children are especially vulnerable to the negative effects of excessive screen time, and youth who spend the most time with screen media are most prone to depression, behavior problems, low self-esteem and poor physical fitness. Often people are unaware of the amount of their screen time. We created an Android app that is able to track and visualize phone time usage patterns and statistics to help user establish awareness of how much and in what patterns they use their Android devices. For example, the app tracks and visualize phone-level and app-level device usage, the number of times the user picks up the device, notifications from various apps, as well as the how the device usage is distributed in various categories of apps, etc. Compared with currently existing screen time apps, such as Apple's ScreenTime and Google's new Digital Wellbeing, our app is able to not only track mobile device usage, but also provide screen time goal management and mechanisms to encourage off-device time.

Resolving Dark Web Identities*

Student Poster Abstract

Babur Kohy
IT and Cyber Department
Marymount University
Arlington, VA 22201
bkohy@marymount.edu

Criminals continue to pose a significant threat on the dark web; a threat that will continue for decades. As cybersecurity professionals, we often hear and read about many forms of breached data for sale on the dark web yet we seldom learn how to properly resolve the identities of these criminals. Just having awareness of data breaches, sales, and/or knowledge of an uncontrolled release of information is not enough to adequately diagnose, fix, and prevent these problems from happening again. Therefore, it is important to quickly identify and expose these bad actors using the same hidden networks they use for exploitation and extortion. My research will attempt to offer a solution to these problems by investigating several multi-faceted strategies to identify, analyze, and expose criminal dark web identities.

A Template for Useful Proof of Work*

Student Poster Abstract

Riley Vaughn and Sajedul Talukder
Mathematics and Computer Science
Edinboro University
Edinboro, PA 16444
{rv161780,stalukder}@edinboro.edu

Cryptocurrencies and numerous other dispersed frameworks use agreement calculations so as to accomplish concession to information. The calculation utilized by Bitcoin and numerous different cryptographic forms of money for this design is known as Proof of Work (PoW). A PoW algorithm typically demands that a large amount of computing power be used to solve an easily verifiable problem. Current implementations of Proof of Work entail vast quantities of energy consumption, where the bulk of this energy is expended exclusively on consensus-building. Our aim is not to minimize energy consumption directly, but to make it possible for Proof of Work to produce more useful and pragmatic computation, so that energy is saved by not running these computational tasks separately. We are building a template for proof of work protocols in our study, such that if followed, a protocol with similar security guarantees can be assured as the proof of work found in Bitcoin. Secondarily, we also develop "useful" prototypes based on this template. Our approach is not to directly decrease energy consumption, but rather to make less waste from such consumption.