2019

To Heck With Ethics: Thinking About Public Issues With a Framework for CS Students

Michael Glass

The Journal of Computing Sciences in Colleges

Papers of the 28th Annual CCSC Rocky Mountain Conference

October 11th-12th, 2019
University of Sioux Falls
Sioux Falls, SD

Baochuan Lu, Editor
Southwest Baptist University

Pam Smallwood, Regional Editor
Regis University

Volume 35, Number 2

October 2019

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. Printed in the USA. POSTMASTER: Send address changes to Susan Dean, CCSC Membership Secretary, 89 Stockton Ave, Walton, NY 13856.

Table of Contents

The Consortium for Computing Sciences in Colleges Board of Directors

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

Foreword

The following five CCSC conferences will take place this fall.

Midwestern Conference	October 4-5, 2019
	Benedictine University in Lisle, IL
Northwestern Conference	October 4–5, 2019
	Pacific University, Forest Grove, OR
Rocky Mountain Conference	October 11-12, 2019
	University of Sioux Falls in Sioux Falls, SD
Eastern Conference	October 25-26, 2019
	Robert Morris University in Moon Township, PA
Southeastern Conference	October 25-26, 2019
	Auburn University in Auburn, AL

The papers and talks cover a wide variety of topics that are current, exciting, and relevant to us as computer science educators. We publish papers and abstracts from the conferences in our JCSC journal. You will get the links to the digital journals in your CCSC membership email. You can also find the journal issues in the ACM digital library and in print on Amazon.

Since this spring we have switched to Latex for final manuscript submission. The transition has been smooth. Authors and regional editors have worked hard to adapt to the change, which made my life a lot easier.

The CCSC board of directors have decided to deposit DOIs for all peer-reviewed papers we publish. With the DOIs others will be able to cite your work in the most accurate and reliable way.

Baochuan Lu
Southwest Baptist University
CCSC Publications Chair

Welcome to the 2019 CCSC Rocky Mountain Conference

Welcome to the 28th annual conference of the Rocky Mountain (RM) Region of the Consortium for Computing Sciences in Colleges hosted by the University of Sioux Falls, in Sioux Falls, South Dakota. The CCSC RM region board members are grateful for the authors, presenters, speakers, attendees, and student participating in this year's conference. We especially would like to thank the faculty, staff, and students at the University of Sioux Falls who have provided a great venue for the RM conference, and Shawn Chiappetta, the site chair, who worked diligently on behalf of the conference and CCSC to make all the arrangements for the conference.

This year we received 14 paper submissions on a variety of topics, of which 10 papers were accepted for presentation in the conference. Multiple reviewers, using a double-blind paper review process, reviewed all submitted papers for the conference. The review process resulted in an acceptance rate of 71%. In addition to the paper presentations, there will be two interesting tutorials/workshops out of two total tutorials/workshops submissions. We truly appreciate the time and effort put forth into the reviewing process by all the reviewers. A special thank you goes to co-Submission chair Karina Assister and co-Program chair Shawn Chiappetta who worked with co-chair of both areas, Mohamed Lotfy. Without their dedicated effort, none of this would be possible.

The CCSC RM region board would like to thank our national partners: TuringsCraft, Google for Education, GitHub, the National Science Foundation (NSF), Codio, zyBooks, the National Center for Women Information Technology (NCWIT), TERADATA University Network, Mercury Learning and Information, Mercy College, and the Association for Computing Machinery in-cooperation with SIGCSE. We hope you enjoy the conference and take the opportunity to interact with your colleagues and we hope you leave enthused and motivated. As you plan your scholarly work for the coming year, we invite you to submit a paper, workshop, tutorial, or panel for a future CCSC RM region conference or to serve as a reviewer or on the CCSC RM region board. Please encourage your colleagues and students to participate in future CCSC RM region conferences.

Jun Zheng
New Mexico Institute of Mining and Technology
(New Mexico Tech)
Conference Chair

2019 CCSC Rocky Mountain Conference Steering Committee

Jun Zheng, Conference Chair	New Mexico Tech, NM
Shawn Chiappetta, Site Co-chair	University of Sioux Falls, SD
Matthew Rieck, Site Co-chair	University of Sioux Falls, SD
Karina Assiter, Submission Co-chair	Landmark College, VT
Mohamed Lotfy, Submission Co-chair	Regis University, CO
Kim Bartholomew, Publicity Chair	Utah Valley University, UT
Michael Leverington, Student Posters Chair	University of Nevada, Reno, NV
Mohamed Lotfy, Program Co-chair	Regis University, CO
Shawn Chiappetta, Program Co-chair	University of Sioux Falls, SD

Regional Board — 2019 CCSC Rocky Mountain Region

Mohamed Lotfy, Board Representative	Regis University, CO
Ed Lindoo, Treasurer	Regis University, CO
Pam Smallwood, Regional Editor	Regis University, CO
Durga Suresh, Registrar	Wentworth Institute of Technology, MA
Kim Bartholomew, Webmaster	Utah Valley University, UT

Reviewers — 2019 CCSC Rocky Mountain Conference

Amin, Mohammad	National University, San Diego, CA
Anthony, Barbara	Southwestern University, Georgetown, TX
Assiter, Karina	Landmark College, Putney, VT
Bandi, Ajay	Northwest Missouri State University, Maryville, MO
Bartholomew, Kimberly	Utah Valley University, Orem, UT
Bonakdarian, Esmail	Dominican University, River Forest, IL
Church, James	Austin Peay State University, Clarksville, TN
D'Antonio, Lawrence	Ramapo College of New Jersey, Mahwah, NJ
Datta, Soma	Univ of Houston Clear Lake, Houston, TX
Eisele, Victoria	Front Range Community College, Fort Collins, CO
Ferrer, Gabriel	Hendrix College, Conway, AR
Glass, Michael	Valparaiso University, Valparaiso, IN
Harris, Laurie	Southern Utah University, Cedar City, UT
Hemmes, Jeffrey	Regis University, Denver, CO
Lenth, Kathryn	Westminster College, Salt Lake City, UT
Li, Jiang	Austin Peay State University, Clarksville, TN
Liang, Jingsai	Westminster College, Salt Lake City, UT
Lindoo, Ed	Regis University, Denver, CO
Lotfy, Mohamed	Regis University, Denver, CO
Lulis, Evelyn	DePaul University, Chicago, IL
Mazumdar, Subhasish	New Mexico Tech, Socorro, NM
Ramyaa, Ramyaa	New Mexico Tech, Socorro, NM
Salinas Duron, Daniel	Westminster College, Salt Lake City, UT
Smallwood, Pam	Regis University, Denver, CO
Suresh, Durga	Wentworth Institute of Technology, Boston, MA
Teresco, James	Siena College, Loudonville, NY
Yu, Yingbing	Austin Peay State University, Clarksville, TN
Zheng, Jun	New Mexico Tech, Socorro, NM

What Will You Build?*

Keynote

Ryan Swanstrom
Data Scientist
Unify Consulting

Abstract

You wake up every morning and one the first things you probably do, is reach for your phone. It has the capability to connect you to nearly anyone on the planet or teach you almost anything. Yet, if we are honest, most of us spend our time scrolling through social media, watching cat videos, or playing games. I don't think that is good enough.

I believe we are still in the early stages of how the internet is going to impact our lives. I hope we all take advantage of the life and time we have been given. We are currently building the infrastructure and processes which will guide life for the next 100 years. As a computer scientist in 2019, you not only get to be amongst the first to use this technology, you have the opportunity to build it.

Bio

Since creating the first data science specific blog on the internet in 2012, Ryan now helps to educate people about using data to solve problems. Ryan has nearly 20 years of experience as a data scientist and software engineer with companies like Microsoft, Wells Fargo and SAIC. He has built software to: track nuclear weapons, process satellite images, ingest millions of dollars a day, and score software development. Ryan is currently a data scientist at Unify Consulting and he holds a PhD in Computational Science and Statistics from South Dakota State University. He lives in South Dakota with his bride and 5 children. Follow his Data Science 101 blog `http://101.datascience.community` or find him on most of your favorite social media sites `@ryanswanstrom`

*Copyright is held by the author/owner.

Introduction to Jetstream: A Research and Education Cloud[*]

Conference Tutorial

Sanjana Sudarshan and Jeremy Fischer
Research Technologies
Indiana University
Bloomington, IN 47401
`{ssudarsh, jeremy}@iu.edu`

1 Introduction

Jetstream is the first production cloud funded by the National Science Foundation (NSF) for conducting general-purpose science and engineering research as well as an easy-to-use platform for education activities. Unlike many high-performance computing systems, Jetstream uses the interactive Atmosphere graphical user interface developed as part of the iPlant (now CyVerse) project and focuses on interactive use on uni-processors or multiprocessors. This interface provides for a lower barrier of entry for use by educators, students, practicing scientists, and engineers. A key part of Jetstream's mission is to extend the reach of the NSF's eXtreme Digital (XD) program to a community of users who have not previously utilized NSF XD program resources, including those communities and institutions that traditionally lack significant cyberinfrastructure resources. One manner in which Jetstream eases this access is via virtual desktops facilitating use in education and research at small colleges and universities, including Historically Black Colleges and Universities (HB-CUs), Minority Serving Institutions (MSIs), Tribal colleges, and higher education institutions in states designated by the NSF as eligible for funding via the Established Program to Stimulate Competitive Research (EPSCoR).

While cloud resources won't replace traditional HPC environments for large research projects, there are many smaller research and education projects that would benefit from the highly customizable, highly configurable, programmable

[*]Copyright is held by the author/owner.

cyberinfrastructure afforded by cloud computing environments such as Jetstream. Jetstream is a Infrastructure-as-a-Service platform comprised of two geographically isolated clusters, each supporting hundreds of virtual machines and data volumes. The two cloud systems are integrated via a user-friendly web application that provides a user interface for common cloud computing operations, authentication to XSEDE via Globus, and an expressive set of web service APIs.

Jetstream enables on-demand access to interactive, user-configurable computing and analysis capability. It also seeks to democratize access to cloud capabilities and promote shareable, reproducible research. This event will describe Jetstream in greater detail, as well as how its unique combination of hardware, software, and user engagement support the "long tail of science." This tutorial will describe Jetstream in greater detail, as well as how its unique combination of hardware, software, and user engagement support the "long tail of science." Attendees will get a greater understanding of how Jetstream may enhance their education or research efforts via a hands-on approach to using Jetstream via the Atmosphere interface.

2 Tutorial Description

This tutorial requires two to three hours.

- Prerequisites: Basic Linux command line knowledge a plus (but not required)
- Required: Laptop, modern web browser (Chrome, Firefox, Safari)
- Targeting: Educators, Researchers, Campus Champions/ACI-Ref Facilitators, Campus research computing support staff

This tutorial will first give an overview of Jetstream and various aspects of the system. Then we will take attendees through the basics of using Jetstream via the Atmosphere web interface. This will include a guided walk-through of the interface itself, the features provided, the image catalog, launching and using virtual machines on Jetstream, using volume-based storage, and best practices.

We are targeting users of every experience level. Atmosphere is well-suited to both HPC novices and advanced users. This tutorial is generally aimed at those unfamiliar with cloud computing and generally doing computation on laptops or departmental server resources. While we will not cover advanced topics in this particular tutorial, we will touch on the available advanced capabilities during the initial overview.

3 Tutorial Program

This is a sample tutorial program. Time required for this tutorial is approximately 3 hours.

- What is Jetstream?
- Q & A and what brief hands-on overview
- Getting started with Jetstream, including VM launching
- Break
- Accessing your VM, creating and using volumes
- Customizing and saving images, DOIs
- Cleaning up
- Final Q & A

Developing Cloud-based Mobile Apps Using Google Firebase[*]

Conference Tutorial

Robert Sjodin and Mohamed Lotfy
Regis University
Denver, CO 80221
{rsjodin, mlotfy}@regis.edu

Cloud computing is an architectural form of distributed enterprise computing that has experienced increased popularity over the last decade. Enterprise solutions that use cloud computing such as Backend as a Service (BaaS) involve both front-end and back-end considerations. Front-end considerations address the environment of the client device; namely, whether the client app resides on a desktop and/or a mobile device, and whether the client app runs natively or as a web application. Back-end considerations address the host platform (e.g., Google, Amazon, Microsoft) and a suite of tools for deploying and managing the host services.

Cloud computing service models are used to describe "full-stack" business solutions. That is, they include all aspects of an enterprise app: user interface, business layer, data layer, etc. But what about mobile applications, where the "front-end" user interface is resident on the mobile device. To address this need, two additional service models have evolved for mobile apps: BaaS and Database as a Service (DaaS).

- Backend as a Service (BaaS)
  The BaaS service model focuses on the business logic of mobile applications, thereby freeing the app from needing to perform long-running computational tasks. BaaS providers include Google Firebase, AWS, Parse Server.

[*]Copyright ©2019 by the Consortium for Computing Sciences in Colleges. Permission to copy without fee all or part of this material is granted provided that the copies are not made or distributed for direct commercial advantage, the CCSC copyright notice and the title of the publication and its date appear, and notice is given that copying is by permission of the Consortium for Computing Sciences in Colleges. To copy otherwise, or to republish, requires a fee and/or specific permission.

- Database as a Service (DaaS)
  The DaaS service model focuses on the data used by mobile applications, by providing database storage technology; e.g., SQL relational and/or NoSQL implementations.

The Cloud has become an important factor in enterprise computing and will remain so in the foreseeable future. According to [2], 83% of enterprise workloads will be in the Cloud by 2020. Hence, there's both a need and an opportunity to provide educational training to equip students with the necessary skills to compete in this high-demand market. Teaching Cloud computing concepts covering the full-stack business solution requires introducing students to the different full-stack layers.

In addition, the ACM/IEEE Computer Science Curricula 2013 (CS2013) added platform-based development (PBD) as a new knowledge area to the Computer Science body of knowledge. CS2013 recommended adding web application development and applying it over a wide range of ecosystems as part of the PBD knowledge area. CS2013 acknowledged the "increasing use of platform-specific programming environments, both at the introductory level and in upper-level electives" [1].

Tutorial Description

In this hands-on tutorial we will provide a mobile app working example that illustrates the use of cloud computing in the context of Database as a Service (DaaS). In particular, the tutorial example involves the following architectural components:

1. An Android mobile app that provides the front-end user interface
2. A Cloud NoSQL real-time database, hosted in Google Firebase

The mobile app allows the user to manage a list of items (i.e., domain objects) that illustrates the use of Google Firebase in an instructional setting.

Tutorial program

Step-by-step implementation and testing of a DaaS scenario (using Google Firebase as the cloud provider).

Expected outcomes

Attendees will exit the tutorial with a working example of a DaaS mobile application.

Target audience

Any faculty who desires to incorporate cloud computing in their courses.

Prerequisites

Attendees should be familiar with OO programming; e.g. Java, C++, Python, etc. It is highly recommended that attendees bring their own laptops with Android Studio installed.

References

[1] ACM/IEEE. Computer science curricula 2013 - curriculum guidelines for undergraduate degree programs in computer science. *The Joint Task Force on Computing Curricula, Association for Computing Machinery (ACM) IEEE Computer Society*, 2013.

[2] Louis Columbus. 83% of enterprise workloads will be in the cloud by 2020, 2018. `https://www.forbes.com/sites/louiscolumbus/2018/01/07/83-of-enterprise-workloads-will-be-in-the-cloud-by-2020/#1f71f2a86261`.

Learning Language Equations and Regular Languages using Alternating Finite Automata[*]

Aziz Fellah and Ajay Bandi
School of Computer Science and Information Systems
Northwest Missouri State University
Maryville, MO 64468
{afellah,ajay}@nwmissouri.edu

Abstract

Learning regular languages using several classes of finite state machines and different learning framework has been one of the objective in the theory of computation and computability courses. Alternating finite automata (AFA) are an appealing abstraction and a key ingredient in modeling many software systems and parallel computations. Despite the fact that AFA accept regular languages, they have several interesting properties due to their transition structures such as their mode of accepting is quite different from that of deterministic finite automata (DFA) and nondeterministic finite automata (NFA); and importantly, AFA are double-exponentially more succinct than DFA. This paper presents a new paradigm to learning regular languages and solving language equations using AFA. Such models can be described naturally as a set of equations that parallels the solutions of algebraic equations. Moreover, the solution of such systems of equations is the class of regular languages.

1 Introduction

Despite the rapid pace of the technology which has significantly altered many aspects of the CS programs, the theory of computation and its cluster of related disciplines continue to play a foundational role in the field of computer

[*]Copyright ©2019 by the Consortium for Computing Sciences in Colleges. Permission to copy without fee all or part of this material is granted provided that the copies are not made or distributed for direct commercial advantage, the CCSC copyright notice and the title of the publication and its date appear, and notice is given that copying is by permission of the Consortium for Computing Sciences in Colleges. To copy otherwise, or to republish, requires a fee and/or specific permission.

science. Most undergraduate CS programs offer a course in theory of computation, alongside with formal languages, automata, and computability. Such a course expose students to different types of abstract models and the foundations of computation. Regular languages is an important class of formal languages in the Chomsky's hierarchy and are the first concepts to be taught in the theory of computation and computability courses. A regular language can be expressed symbolically with a regular expression, a regular grammar, and conventional models, namely deterministic finite automata (DFA) and nondeterministic finite automata (NFA). A vast amount of literature is devoted to learning formal and regular languages [10] which are used in parsing and designing programming languages. The concept of alternation [5, 9, 7] extends the notion of nondeterminism by including the universal quantifiers in addition to the existential quantifiers. A formalization of this concept yields the definition of alternating finite automata (AFA) which form a powerful abstraction layer beyond nondeterminism. AFA, a generalization of NFA, become an influential model that has received significant interests inside and outside the classroom, from instructors and researchers [11, 6, 9, 7, 8, 3, 1]. Similarly to DFA and NFA, AFA are equally expressive as the class of regular languages. Aside from the applications in modeling many systems and phenomena, AFA models capture different level of abstractions that are used in software system designs, model checking, and various areas of computer science [11, 6, 8]. While all these formalisms, DFA, NFA and AFA, are equivalent in expressive power terms of language recognition, they differ in other terms. Importantly, NFA are exponentially more succinct than DFA but AFA are double-exponentially more succinct than DFA. Language equations are equations over language operations where both the constants and variables are languages. Usually, they are formalized through various classes of finite state machines such as DFA and NFA using two operations, union and concatenation. The relevant properties of language equations, such that existence and uniqueness of their solutions have been established in the literature [7, 4].

An *alphabet* is a finite, nonempty set. The elements of an alphabet are called *symbols* or *letters*. A *word* (string) over an alphabet Σ is a finite sequence consisting of zero or more letters of Σ. The word consisting of zero letters is called the *empty word*, denoted by ϵ. By definition, $|\epsilon| = 0$. The set of all words (respectively all nonempty words) over an alphabet Σ is denoted by Σ^* (Σ^+). Given a language L over an alphabet Σ, the Kleene closure (Kleene star "*") of L is the set $L^* = \bigcup_{i=0}^{\infty} L^i$ and the positive Kleene closure (Kleene plus "+") of L is $L^+ = \bigcup_{i=1}^{\infty} L^i$. The language $\overline{L} = \Sigma^* \setminus L$ is the complement of L. The *concatenation* of two words w_1 and w_2 is the word consisting of the symbols of w_1 followed by the symbols of w_2, denoted $w_1 \cdot w_2$. The length of a word w, denoted by $|w|$, is the number of letters in w. We denote by the symbol $\mathcal{B}$ the Boolean semiring, $\mathcal{B} = \{0, 1\}$. Let Q be a set. Then $\mathcal{B}^Q$ is the set of all mappings of Q into $\mathcal{B}$. Note that $u \in \mathcal{B}^Q$ can also be considered as a Q-vector over $\mathcal{B}$.

The remaining of the paper is organized as follows. In section 2, we describe alternating finite automata and state some definitions and results that can be used in subsequent sections. Section 3 introduces systems of equations to represent AFA. These equations, and not the transition diagrams, are in essence the underlying algebraic framework that makes AFA appealing from the point of view of representation and flexibility. In section 4, we exhibit constructions for the Boolean operations on AFA. Section 5, shows such systems of equations parallel the solution of algebraic equations. In particular, the solution of such systems of equations is the class of regular languages. We conclude the paper with some potential discussions in section 6. This paper is based on our original work on AFA but summarized in a compact form given the limitation on the conference page number.

2 Alternating Finite Automata

NFA are a generalization of DFA by allowing a state to have multiple outgoing transitions labeled with the same symbol or a ϵ-transition. A string is accepted by an NFA if there exists some path that leads to an accepting state. In a nondeterministic computation all configurations are *existential* in the sense that there exists at least one successful path that leads to acceptance. An alternating finite automaton (AFA) may have also *universal* configurations from which the computation branches into a number of parallel computations that must all lead to acceptance. We represent existential and universal choices by a Boolean formula. Formally, let Q be a set, we use $\mathcal{B}^Q$ to be the set of all Boolean formulas over Q. That is, $\mathcal{B}^Q$ is built from the elements $q \in Q$, 1, and 0 using the binary operations *and* ($\vee$), *or* ($\wedge$), and *not* ($^-$). If in a given state q the AFA reads an input symbol a, it will activate all states of the AFA to work on the remaining part of the input in parallel. Once the states have completed their tasks, q will evaluate their results using a Boolean function and pass on the resulting value to the state by which it was activated. A word w is accepted if the starting state computes the value of 1. Otherwise, it is rejected. We now formalize this idea.

Definition 1 *An alternating finite automaton* (*AFA*) *is a quintuple* $A = (Q, \Sigma, s, F, g)$ *where* (a) *Q is a finite set, the set of states;* (b) *Σ is an alphabet, the input alphabet;* (c) *$s \in Q$ is the starting state;* (d) *$F \subseteq Q$ is the set of final states;* (e) *g is a mapping of Q into the set of all mappings of $\Sigma \times \mathcal{B}^Q$ into $\mathcal{B}$.*

We turn to defining the sequential behavior of an AFA. For $q \in Q$ and $a \in \Sigma$, let $g_q(a)$ be the Boolean function defined as:

$$g_q(a, u) : \Sigma \times \mathcal{B}^Q \rightarrow \mathcal{B}$$

where $a \in \Sigma$ and $u \in \mathcal{B}^Q$. Also, for $a \in \Sigma$, $q \in Q$, and $u \in \mathcal{B}^Q$, $g_q(a, u) = g_q(a)(u)$, is equals to either 0 or 1. Now define $f \in \mathcal{B}^Q$ by the condition

$$f_q = 1 \iff q \in F.$$

f is called the *characteristic vector* of F. We extend g to a mapping of Q into the set of all mappings of $\Sigma^* \times \mathcal{B}^Q$ into $\mathcal{B}$ as follows:

$$g_q(w, u) = \begin{cases} u_q & \text{if } w = \epsilon, \\ g_q(a, g(v, u)) & \text{if } w = av \text{ with } a \in \Sigma \text{ and } v \in \Sigma^* \end{cases}$$

where $w \in \Sigma^*$ and $u \in \mathcal{B}^Q$.

Definition 2 *Let $A = (Q, \Sigma, s, F, g)$ be an AFA. A word $w \in \Sigma^*$ is accepted by A if and only if $g_s(w, f) = 1$. The language accepted by A is the set $L(A) = \{w \mid w \in \Sigma^* \wedge g_s(w, f) = 1\}$.*

Example 1 Consider the following AFA $A = (Q, \Sigma, s, F, g)$ where $Q = \{q_0, q_1, q_2\}$, $\Sigma = \{a, b\}$, $s = \{q_0\}$, $F = q_2$, and g is given by the following table. The example shows a run of the AFA on the input *bab*.

State	a	b
q_0	$q_0 \wedge q_1$	$q_1 \vee \overline{q_2}$
q_1	q_0	$\overline{q_0} \wedge q_2$
q_2	$q_0 \vee \overline{q_1}$	1

In the same example of AFA, we have drawn the existential states as $\vee$ and the universal states as $\wedge$. The AFA can have multiple runs on a given input where both of these choices coexist. Notice that the run branches in parallel to two states, $\overline{q}_0$ and q_2 on the second input symbol b from q_1.

Example 2 Let $w = bab$ be a string. We will check whether w is accepted by the NFA A.

$$\begin{aligned}
& g_{q_0}(bab, f) \\
= \;& g_{q_1}(ab, f) \vee \overline{g_{q_2}(ab, f)} \\
= \;& g_{q_0}(b, f) \vee \overline{(g_{q_0}(b, f) \vee \overline{g_{q_1}(b, f)})} \\
= \;& (g_{q_1}(\epsilon, f) \vee \overline{g_{q_2}(\epsilon, f)}) \vee \overline{((g_{q_1}(\epsilon, f) \vee \overline{g_{q_2}(\epsilon, f)}) \vee \overline{(\overline{g_{q_0}(\epsilon, f)} \wedge g_{q_2}(\epsilon, f))})} \\
= \;& (0 \vee \overline{1}) \vee \overline{((0 \vee \overline{1}) \vee \overline{(\overline{0} \wedge 1)})} = (0 \vee 0) \vee \overline{((0 \vee 0) \vee \overline{(1 \wedge 1)})} \\
= \;& (0) \vee \overline{((0) \vee \overline{(1)})} = 0 \vee \overline{(0 \vee 0)} = 0 \vee \overline{(0)} = 0 \vee 1 \\
= \;& 1
\end{aligned}$$

Therefore the string *bab* is accepted.

Proposition 1 [9, 4] *AFA are equivalent in term of language recognition power to NFA and DFA. That is, they can only accept regular languages.*

Proposition 2 *AFA are double-exponentially more succinct that DFA.*

Let $A = (Q, \Sigma, S, F, g)$ be an AFA to simulate an equivalent DFA $A' = (Q', \Sigma, S', F', g')$, Q' needs $2^{2^{|Q|}}$ Boolean functions on Q. For instance, an NFA may be exponentially smaller than the minimal DFA, and an AFA may be doubly-exponentially smaller than the minimal DFA.

3 Representing AFA by Systems of Language Equations

Language equations are equations defined over languages where both the constants and variables are formal languages. It is well-known that regular languages can be described as the solutions of systems of one-sided linear equations in an appropriate semiring [7, 4, 2]. We show that AFA can be readily represented by systems of equations. However, the systems of equations to be considered involve Boolean expressions over a finite set X of variables and the symbols of an alphabet Σ. The main result of this section is that the solutions of such systems of equations are precisely the regular languages and that, indeed, there is a natural correspondence between AFA and such systems of equations. In the sequel we associate with each AFA a system of equations such that the languages accepted by the AFA with various start states constitute the unique fixpoint of the system of equations.

Let $A = (Q, \Sigma, s, F, g)$ be an AFA. For $q \in Q$, we use x_q to denote a boolean variable associated with the state q and $\overline{x}_q$ to denote its negation. Let $X_Q = \{x_q \mid q \in Q\}$. Then the following system $L(A)$ of equations can be used to describe A:

$$L(A) = \left\{ X_q = \sum_{a \in \Sigma} a \cdot g_q(a, X) + \epsilon(f_q) \right\}_{q \in Q} \tag{1}$$

$$\epsilon(f_q) = \begin{cases} \epsilon & \text{if } f_q = 1 \\ \emptyset & \text{otherwise} \end{cases}$$

In the system $L(A)$ of equations, the Boolean function $g_q(a, X)$ is considered as being given by a Boolean expression in $\mathcal{B}(X_Q)$. Any system of language equations of the above form has a unique solution for each X_Q, $q \in Q$. Furthermore, the solution for each X_Q is regular.

4 Boolean Operations on AFA

Of course, as every AFA language is regular, the class of AFA languages is closed under the Boolean operations. However, this is only a "surface" result. Rather than this type of existence result one would like to have concrete constructions for AFA.

Let $A = (Q, \Sigma, s, F, g)$ be an AFA. First we construct the complement of A

$$\overline{A} = (Q, \Sigma, s, F', g')$$

such that $L(\overline{A}) = \Sigma^* \setminus L(A)$. The set F' of final states is defined by the condition

$$q \in F' \iff \begin{cases} q \in F, & \text{if } q \neq s, \\ q \notin F, & \text{if } q = s, \end{cases}$$

for $q \in Q$. For $u \in \mathcal{B}^Q$, let u' be the mapping given by

$$u'_q = \begin{cases} u_q, & \text{if } q \neq s, \\ \overline{u}_q, & \text{if } q = s, \end{cases}$$

for $q \in Q$. The function g' is given by

$$g'_q(a, u) = \begin{cases} g_q(a, u'), & \text{if } q \neq s, \\ \overline{g_q(a, u)}, & \text{if } q = s, \end{cases}$$

where $q \in Q$, $a \in \Sigma$, and $u \in \mathcal{B}^Q$.

Theorem 1 $L(\overline{A}) = \Sigma^* \setminus L(A)$.

Proof: We prove that $g'(w, f') = g(w, f)'$ for all $w \in \Sigma^*$. This is obviously true for $w = \epsilon$. Now assume that the statement holds for $v \in \Sigma^*$ and consider $w = av$ with $a \in \Sigma$. For $q \in Q$, one has

$$g'_q(w, f') = g'_q(a, g(v, f')') = g'_q(a, g(v, f)') = g_q(w, f)'.$$

Thus, $g'_s(w, f') = \overline{g_s(w, f)}$, that is, $w \in L(\overline{A}) \iff w \notin L(A)$. $\square$

Example 3 Let the AFA $A = (Q, \Sigma, s, F, g)$, where $Q = \{1, 2, 3, 4\}$, $\Sigma = \{a, b\}$, $s = 1$, $F = \{1, 4\}$ and g is given as follows. A is represented by $L(A)$. We construct the complement of A, $\overline{A}$, that accepts $\overline{L(A)}$.

$$\begin{array}{llcl} L(A): & x_1 & = & a \cdot (\overline{x}_2 \vee \overline{x}_3) + b \cdot (\overline{x}_2 \vee \overline{x}_3 \vee \overline{x}_4) + \epsilon \\ & x_2 & = & a \cdot (x_2 \vee x_3 \wedge x_4) + b \cdot (x_2 \wedge x_3 \wedge x_4) \\ & x_3 & = & a \cdot (x_2 \wedge x_3 \vee x_3 \wedge \overline{x}_4 \vee \overline{x}_3 \wedge x_4) + b \cdot (x_2 \wedge x_3 \wedge x_4) \\ & x_4 & = & a \cdot (x_2 \wedge x_3 \vee x_2 \wedge \overline{x}_4 \vee x_3 \wedge \overline{x}_4) + b \cdot (x_2 \wedge x_3 \wedge x_4) + \epsilon \end{array}$$

$$\begin{array}{llcl}
\overline{L(A)}: & x_1 & = & a\cdot(x_2\wedge x_3)+b\cdot(x_2\wedge x_3\wedge\overline{x}_4)\\
& x_2 & = & a\cdot(x_2\vee x_3\wedge\overline{x}_4)+b\cdot(x_2\wedge x_3\wedge\overline{x}_4)\\
& x_3 & = & a\cdot(x_2\wedge x_3\vee x_3\wedge x_4\vee\overline{x}_3\wedge\overline{x}_4)+b\cdot(x_2\wedge x_3\wedge\overline{x}_4)\\
& x_4 & = & a\cdot(x_2\wedge x_3\vee x_2\wedge x_4\vee x_3\wedge x_4)+b\cdot(x_2\wedge x_3\wedge\overline{x}_4)+\epsilon
\end{array}$$

Our next construction is for the union of languages accepted by AFA. For $i=1,2$ let $A^{(i)}=(Q^{(i)},\Sigma,s^{(i)},F^{(i)},g^{(i)})$ be two AFA with disjoint state sets. We construct an AFA

$$A=A^{(1)}\vee A^{(2)}=(Q,\Sigma,s,F,g)$$

such that $L(A^{(1)}\vee A^{(2)})=L(A^{(1)})\cup L(A^{(2)})$. Let s be a new state symbol, $s\notin Q^{(1)}\cup Q^{(2)}$, let

$$Q=Q^{(1)}\cup Q^{(2)}\cup\{s\},$$

$$F=\begin{cases}F^{(1)}\cup F^{(2)}, & \text{if } s^{(1)}\notin F^{(1)} \text{ and } s^{(2)}\notin F^{(2)},\\ F^{(1)}\cup F^{(2)}\cup\{s\}, & \text{otherwise.}\end{cases}$$

The function g is given as follows

$$g_q(a,u)=\begin{cases}g_q^{(i)}(a,u\big|_{Q^{(i)}}), & \text{if } q\in Q^{(i)} \text{ with } i\in\{1,2\},\\ g_{s^{(1)}}^{(1)}(a,u\big|_{Q^{(1)}})\vee g_{s^{(2)}}^{(2)}(a,u\big|_{Q^{(2)}}), & \text{if } q=s.\end{cases}$$

where $q\in Q$, $a\in\Sigma$, and $u\in B^Q$.

Theorem 2 $L(A^{(1)}\vee A^{(2)})=L(A^{(1)})\cup L(A^{(2)})$.

Proof: By induction on $|w|$ one verifies that

$$g_q(w,f)=\begin{cases}g_{s^{(1)}}^{(1)}(w,f^{(1)})\vee g_{s^{(2)}}^{(2)}(w,f^{(2)}), & \text{if } q=s,\\ \\ g_q^{(i)}(w,f^{(i)}), & \text{if } q\in Q^{(i)} \text{ with } i\in\{1,2\},\end{cases}$$

for $q\in Q$ and $w\in\Sigma^*$. □

The construction of an AFA

$$A=A^{(1)}\wedge A^{(2)}=(Q,\Sigma,s,F,g)$$

such that $L(A^{(1)}\wedge A^{(2)})=L(A^{(1)})\cap L(A^{(2)})$ is similar. With Q as above, one defines

$$F=\begin{cases}F^{(1)}\cup F^{(2)}, & \text{if } s^{(1)}\notin F^{(1)} \text{ or } s^{(2)}\notin F^{(2)},\\ F^{(1)}\cup F^{(2)}\cup\{s\}, & \text{otherwise.}\end{cases}$$

and

$$g_q(a,u) = \begin{cases} g_q^{(i)}(a, u|_{Q^{(i)}}), & \text{if } q \in Q^{(i)} \text{ with } i \in \{1,2\}, \\ \\ g^{(1)}_{s^{(1)}}(a, u|_{Q^{(1)}}) \wedge g^{(2)}_{s^{(2)}}(a, u|_{Q^{(2)}}), & \text{if } q = s. \end{cases}$$

where $q \in Q$, $a \in \Sigma$, and $u \in B^Q$. One then proves that $L(A)$ is indeed the intersection of the two languages.

Theorem 3 $L(A^{(1)} \wedge A^{(2)}) = L(A^{(1)}) \cap L(A^{(2)})$.

Example 4 Let the AFA $A^{(1)} = (Q^{(1)}, \Sigma, s^{(1)}, F^{(1)}, g^{(1)})$ where $Q^{(1)} = \{1,2,3,4\}$, $\Sigma = \{a,b\}$, $s^{(1)} = \{1\}$, $F^{(1)} = \{1,4\}$, and $g^{(1)}$ is given as follows.

$$\begin{array}{rrcl} L(A^{(1)}): & x_1 & = & a \cdot (\overline{x}_2 \vee \overline{x}_3) + b \cdot (\overline{x}_2 \vee \overline{x}_3 \vee \overline{x}_4) + \epsilon \\ & x_2 & = & a \cdot (x_2 \vee x_3 \wedge x_4) + b \cdot (x_2 \wedge x_3 \wedge x_4) \\ & x_3 & = & a \cdot (x_2 \wedge x_3 \vee x_3 \wedge \overline{x}_4 \vee \overline{x}_3 \wedge x_4) + b \cdot (x_2 \wedge x_3 \wedge x_4) \\ & x_4 & = & a \cdot (x_2 \wedge x_3 \vee x_2 \wedge \overline{x}_4 \vee x_3 \wedge \overline{x}_4) + b \cdot (x_2 \wedge x_3 \wedge x_4) + \epsilon \end{array}$$

Let the AFA $A^{(2)} = (Q^{(2)}, \Sigma, s^{(2)}, F^{(2)}, g^{(2)})$ where $Q^{(2)} = \{5,6,7\}$, $\Sigma = \{a,b\}$, $s^{(2)} = \{5\}$, $F^{(2)} = \{5,7\}$ and $g^{(2)}$ is given as follows.

$$\begin{array}{rrcl} L(A^{(2)}): & x_5 & = & a \cdot (x_5) + b \cdot (x_5 \wedge \overline{x}_6 \vee x_5 \wedge \overline{x}_7 \vee \overline{x}_5 \wedge x_6 \wedge x_7) + \epsilon \\ & x_6 & = & a \cdot (x_6) + b \cdot (x_6 \wedge \overline{x}_7 + \overline{x}_6 \wedge x_7) \\ & x_7 & = & a \cdot (x_7) + b \cdot (\overline{x}_7 \wedge x_5 \wedge x_6) + \epsilon \end{array}$$

$A^{(1)} \wedge A^{(2)} = (Q, \Sigma, s, F, g)$ where $Q = \{1,2,3,4,5,6,7\}$, $\Sigma = \{a,b\}$, $s = \{0\}$, $F = \{1,4,5,7\}$, $L(A^{(1)} \wedge A^{(2)}) = L(A^{(1)}) \cap L(A^{(2)})$, and g as follows.

$$\begin{array}{rcl} x_0 & = & a \cdot ((\overline{x}_2 \vee \overline{x}_3) \wedge x_5) + b \cdot ((\overline{x}_2 \vee \overline{x}_3 \vee \overline{x}_4) \wedge \\ & & (x_5 \wedge \overline{x}_6 \vee x_5 \wedge \overline{x}_7 \vee \overline{x}_5 \wedge x_6 \wedge x_7)) + \epsilon \\ x_1 & = & a \cdot (\overline{x}_2 \vee \overline{x}_3) + b \cdot (\overline{x}_2 \vee \overline{x}_3 \vee \overline{x}_4) + \epsilon \\ x_2 & = & a \cdot (x_2 \vee x_3 \wedge x_4) + b \cdot (x_2 \wedge x_3 \wedge x_4) \\ x_3 & = & a \cdot (x_2 \wedge x_3 \vee x_3 \wedge \overline{x}_4 \vee \overline{x}_3 \wedge x_4) + +b \cdot (x_2 \wedge x_3 \wedge x_4) \\ x_4 & = & a \cdot (x_2 \wedge x_3 \vee x_2 \wedge \overline{x}_4 \vee x_3 \wedge \overline{x}_4) + b \cdot (x_2 \wedge x_3 \wedge x_4) + \epsilon \\ x_5 & = & a \cdot (x_5) + b \cdot (x_5 \wedge \overline{x}_6 \vee x_5 \wedge \overline{x}_7 \vee \overline{x}_5 \wedge x_6 \wedge x_7) + \epsilon \\ x_6 & = & a \cdot (x_6) + b \cdot (x_6 \wedge \overline{x}_7 + \overline{x}_6 \wedge x_7) \\ x_7 & = & a \cdot (x_7) + b \cdot (\overline{x}_7 \wedge x_5 \wedge x_6) + \epsilon \end{array}$$

Other proofs are quite similar and omitted due to lack of space. A summary is as follows.

Proposition 3 *Language Equations are closed under concatenation, union, complement, intersection, Kleene star operations. That is,* $L(A^{(1)} \cdot A^{(2)}) = L(A^{(1)}) \cdot L(A^{(2)})$, $L(A^{(1)} \vee A^{(2)}) = L(A^{(1)}) \cup L(A^{(2)})$, $\overline{L(A)}$, $L(A^{(1)} \wedge A^{(2)}) = L(A^{(1)}) \cap L(A^{(2)})$, *and* $(L(A))^* = \bigcup_{i=0}^{\infty} (L(A))^i$.

5 Equations over Languages

Finite automata are represented as systems of language equations with two operations, union and concatenation. The relevant properties of these equations, such that existence and uniqueness of their solutions have been established in the literature. For instance, the equation $X = \alpha X + \beta$, where α and β are fixed languages, and it is well-known by Arden's Lemma that the equation has $\alpha^*\beta$ as solution and if the empty word $\epsilon \notin \alpha$, then it is the only solution.

Theorem 4 [7] *Let* A *be the equational representation of* A. *Let* s *be the starting state of* A. *Then the solution for* X_s *is exactly the language accepted by* A. *Furthermore, the system of equations of the form (1) has a unique solution for each* $X_q \in Q$ *and the solution for each* X_q *is regular.*

Example 5 Given the AFA $A = (Q, \Sigma, s, F, g)$, where $Q = \{1, 2, 3\}$, $\Sigma = \{a, b\}$, $s = 1$, $F = \{3\}$ and g is given by the following system of language equations. The regular language $L(A)$ generated by A is obtained by solving the following system of language equations. The subscript operators $*$ and $+$ indicate the Kleene star and Kleene plus operators, respectively.

$$\begin{aligned} x_1 &= a \cdot (x_1 \vee x_2) + b \cdot x_3 \\ x_2 &= a \cdot x_1 + b \cdot (x_1 \vee x_3) \\ x_3 &= \epsilon \end{aligned}$$

The regular language accepted by the AFA A is obtained by solving the above system of language equations. That is, $x_0 = ab^*a$.

6 Conclusion

Regular expressions have been always centered around NFA and DFA covered in theoretical CS courses. With considerably fewer states than DFA and even NFA, AFA are a generalization of NFA and provide a powerful abstraction layer beyond nondeterminism. Moreover, they are a key ingredient in modeling many software systems, parallel computations, and characterizing model checking algorithms. Language equations have always been far from being well understood in the context of NFA and DFA, but AFA equations, and not the transition diagrams, are in essence the underlying algebraic framework that makes AFA appealing from the point of view of representation and flexibility. However, the systems of equations to be considered involve Boolean expressions over a finite set X of variables and the symbols of an alphabet Σ. The solutions of such systems of equations are precisely the class of regular languages. These language equations may not always be easy to solve algebraically despite being attractive from a theoretical point of view.

References

[1] D. Angluin, S. Eisenstat, and D. Fisman. Learning regular languages via alternating automata. In *Proceedings of the 24th Int. Joint Conference on Artificial Intelligence*, pages 3308–3314. IAAA Press, 2017.

[2] F. Baader and A. Okhotin. On language equations with one-sided concatenation. *Fundamental Informaticae*, 126:1–34, 2013.

[3] S. Berndt, Lutter M. Liskiewicz, M., and R. Reischuk. Learning residual alternating automata. In *Proceedings of the 31st AAAI conference on artificial intelligence*, pages 1749–1755. IAAA Press, 2017.

[4] J. A. Brzozowski and E. L. Leiss. On equations for regular languages, finite automata, and sequential networks. *Theoret. Comput. Sci.*, 10:19–35, 1980.

[5] A.K. Chandra, D.C. Kozen, and L.J. Stockmeyer. Alternation. *J. ACM*, 28:114–133, 1981.

[6] L. D'Antoni, Z. Kincaid, and F. Wang. A symbolic decision procedure for symbolic alternating finite automata. *Electronic Notes in Theoret. Comput. Sci.*, 336:79–99, 2018.

[7] A. Fellah. Equations and regular-like expressions for afa. *Int. J. of Comput. Math*, 51(3-4):157–172, 1994.

[8] A. Fellah. Real-time languages, timed alternating automata, and timed temporal logics: Relationships and specifications. *J. Procedia Computer*, 62:47–54, 2015.

[9] A. Fellah, H. Jürgensen, and S. Yu. Constructions for alternating finite automata. *Int. J. of Comput. Math.*, 35:117–132, 1992.

[10] J.E. Hopcroft, R. Motwani, and Ullman J.D. *Introduction to automata theory, languages, and computation.* Addison-Wesley, Boston, 2001.

[11] J. Kavitha, L. Jeganathan, and G. Sethuraman. Descriptional complexity of alternating finite automta. In *Procedings of the Int. Workshop on Descriptional Complexity of Formal Systems*, pages 188—198, 2016.

Integrating Cloud Computing into the Curriculum[*]

Robert Sjodin and Mohamed Lotfy
Regis University
Denver, CO 80221
`{rsjodin, mlotfy}@regis.edu`

Abstract

In this paper we provide a strategy for integrating cloud-based computing in a university/college setting. We address front-end considerations, back-end considerations, and the practical aspects of teaching these concepts in a Computer Science curriculum. The material provided can be taught in a dedicated course or as supplemental topics in an existing course. An end-to-end working example is provided to illustrate the concepts.

1 Introduction

Cloud computing is an architectural form of distributed enterprise computing that has experienced increased popularity over the last decade. Enterprise solutions that use cloud computing as Backend as a Service (BaaS) involve both front-end and back-end considerations. Front-end considerations address the environment of the client device; namely, whether the client app resides on a desktop and/or a mobile device, and whether the client app runs natively or as a web application. Back-end considerations address the host platform (e.g., Google, Amazon, Microsoft) and a suite of tools for deploying and managing the host services. Advances in cloud computing technologies allowed for the creation of commercial grade cloud offerings by various corporations, as highlighted in Figure 1.

[*]Copyright ©2019 by the Consortium for Computing Sciences in Colleges. Permission to copy without fee all or part of this material is granted provided that the copies are not made or distributed for direct commercial advantage, the CCSC copyright notice and the title of the publication and its date appear, and notice is given that copying is by permission of the Consortium for Computing Sciences in Colleges. To copy otherwise, or to republish, requires a fee and/or specific permission.

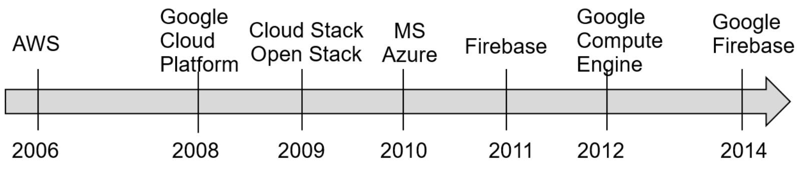


Figure 1: Timeline of Cloud Computing Offerings Models.

According to Columbus [3], 83% of enterprise workloads will be in the Cloud by 2020. There are various reasons for this trend, but chief among them are the following benefits of cloud computing: enhanced organizational visibility, easier collaboration, rapid development, consistency across platforms, enhanced security, lower risk, and last but not least, reduced cost. The Cloud has become an important factor in enterprise computing and will remain so in the foreseeable future. Hence, there's both a need and an opportunity to provide educational training to equip students with the necessary cloud computing skills to compete in this high-demand market. To make this happen, we first identify the various aspects of cloud computing and their supporting architectural components. Thereafter, we suggest the various ways cloud computing can be incorporated into a university/college curriculum. Finally, we provide an end-to-end example as a starting point for teaching cloud computing concepts in the curriculum.

2 An Overview of Cloud Computing

Currently, cloud computing recognizes five service models and three deployment models. The service models identify the functionality offered by a provider, whereas the deployment models identify the public/private nature of the service (i.e., open to the public, or private to a company and/or organization). Figure 2 below shows the relationships among the three cloud computing service models and the level of functionality offered by the cloud providers.

The service models depicted in Figure 2 are used to describe "full-stack" business solutions. That is, they include all aspects of an enterprise app: user interface, business layer, data layer, etc. But what about mobile applications, where the "front-end" user interface is resident on the mobile device. To address this need, two additional service models have evolved for mobile apps: BaaS and Database as a Service (DaaS). The BaaS and DaaS service models are depicted below in Figure 3.

- Backend as a Service (BaaS)

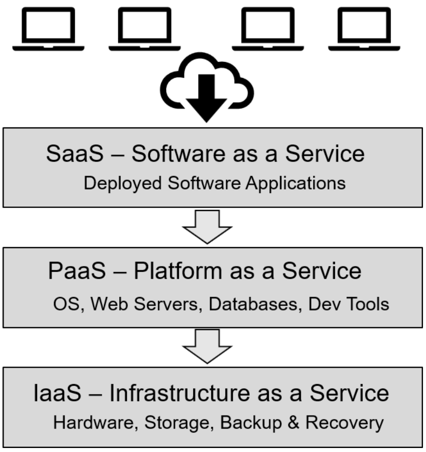

SaaS – Software as a Service
Deployed Software Applications
PaaS – Platform as a Service
OS, Web Servers, Databases, Dev Tools
IaaS – Infrastructure as a Service
Hardware, Storage, Backup & Recovery

Figure 2: The Relationships of SaaS, PaaS, and IaaS Service Models.

The BaaS service model focuses on the business logic of mobile applications, thereby freeing the app from needing to perform long-running computational tasks. BaaS providers include Google Firebase, AWS, Parse Server.

- Database as a Service (DaaS)
  The DaaS service model focuses on the data used by mobile applications, by providing database storage technology; e.g., SQL relational and/or NoSQL implementations.

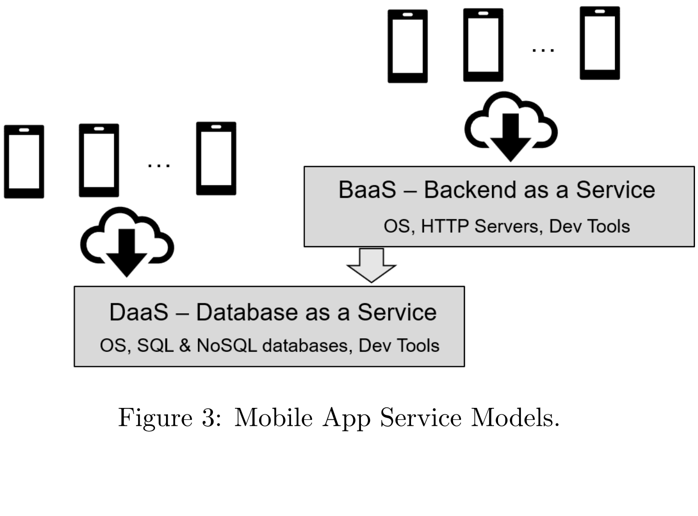

…
…
BaaS – Backend as a Service
OS, HTTP Servers, Dev Tools
DaaS – Database as a Service
OS, SQL & NoSQL databases, Dev Tools

Figure 3: Mobile App Service Models.

Teaching cloud computing concepts covering the full-stack business solution requires introducing students to the different full-stack layers. Vijayalakshmi et al. [7] provided an integrated course project approach where a common android application project is used in the mobile application development, cloud computing and information security courses. Guo and Koufakou [5] provided the structure of a hands-on cloud computing course were students had to develop eleven labs using AWS to master the different cloud computing services. Deb, Fuad and Irwin [4] provided an example of teaching the different cloud services (cloud architecture, cloud analytics, and database management) using three integrated modules in three undergraduate courses. In Chen et al. [2], teaching cloud services was integrated into different exercises in seven undergraduate courses. Rehman et al. [6] provided the structure of a cloud Computing course were student learned the different cloud services using four AWS projects.

In addition, the ACM/IEEE Computer Science Curricula 2013 (CS2013) added platformbased development (PBD) as a new knowledge area to the Computer Science body of knowledge. CS2013 recommended adding web and mobile application development and applying it over a wide range of ecosystems as part of the PBD knowledge area. CS2013 acknowledged the "increasing use of platform-specific programming environments, both at the introductory level and in upper-level electives" [1].

3 An Instructional Example

In this section, we provide an end-to-end, working example that illustrates the use of cloud computing in the context of DaaS (Database as a Service). In particular, the example involves the following architectural components:

- An Android mobile app that provides the front-end user interface
- A cloud NoSQL real-time database, hosted in Google Firebase

The above configuration is taught at Regis University in an undergraduate course entitled "CS338: Mobile and Enterprise Computing". The course has eight topics, with the last three dedicated to Google Firebase. The example illustrated below is taught in the first Google Firebase topic.

The example is an app that allows the user to manage a list of items (i.e., domain objects). The site map, Figure 4 below, identifies the functionality of the user interface and the supported navigation.

To illustrate the use of Google Firebase in an instructional setting, we can proceed in a number of ways:

- persist the "list of items" in Google Firebase,
- persist the "Contact Us" information in Google Firebase,
- or, persist user "Notes" in Google Firebase.

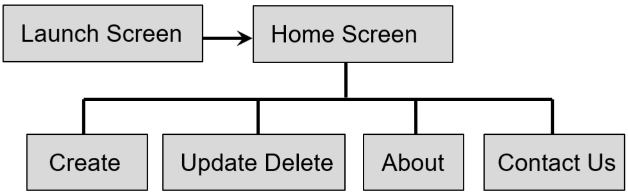


Figure 4: Site Map for "My List" app.

Any of the above candidate scenarios will work, but we chose the last option: User Notes, simply because the notion of "user note" can be applied to any mobile app. The concept is depicted in Figure 5.

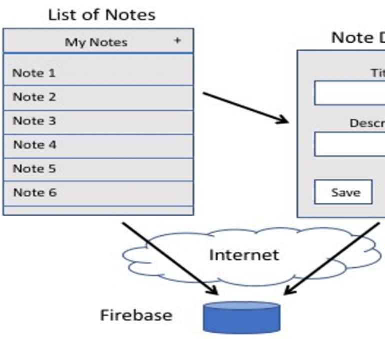


Figure 5: User notes stored in Google Firebase.

To make this scenario happen, we need to perform the following tasks:

1. Configure Google Firebase.
2. Configure the Android project.
   Before using Google Firebase in the app, the Android Studio project has to be configured with the following tasks:
   - Updating the Android Manifest
   - Adding Google Firebase to the Android App
   - Referencing the core Google Firebase library in the app's Gradle file
   - Referencing additional Google Firebase libraries in the app's Gradle file
3. Develop the app.

The app provides a capability to create, read/retrieve, update and delete user-supplied notes in the Google Firebase Realtime Database. In industry, these operations are collectively known as "CRUD" operations. The UI for CRUD functionality is typically implemented in a Master List / Details design pattern. To implement the pattern, we need to create two Android activities:

- The NoteListActivity – displays all the user notes in a List View
- The NoteDetailActivity – displays the details of a particular user note

The layout of the two activities, as displayed in Android Studio, are shown in Figure 6 below.

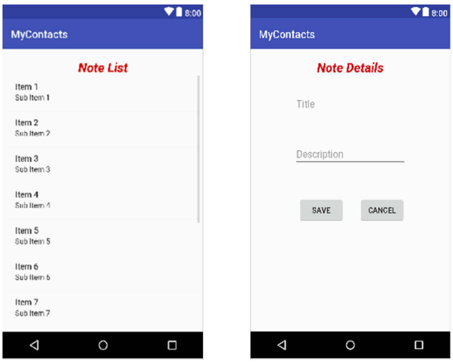


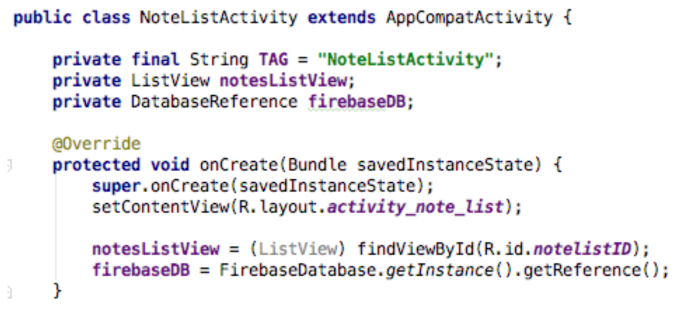
Figure 6: User notes App Interface.

The List and Detail activities will perform CRUD operations on "user notes". Hence, to facilitate those functions, we create a Note class and place it in a "domain" package of our project. The Note class will have three data members:

- uid, a unique identifier that's auto generated by the Google Firebase real-time database
- title, supplied by the user
- description, supplied by the user.

We now turn our attention to the two activities: NoteListActivity and NoteDetailActivity. The NoteListActivity has two data members:

- noteListView - a reference to the ListView that displays the list of user notes
- firebaseDB - a reference to the remote Google Firebase database



```
public class NoteListActivity extends AppCompatActivity {

    private final String TAG = "NoteListActivity";
    private ListView notesListView;
    private DatabaseReference firebaseDB;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_note_list);

        notesListView = (ListView) findViewById(R.id.notelistID);
        firebaseDB = FirebaseDatabase.getInstance().getReference();
    }
```

- OnCreate - The “onCreate” method loads the layout and initializes the two data members.
- OnResume - The onResume method does the bulk of the work for the NoteListActivity. In particular, it adds a “ValueEventListener” to the Google Firebase DB, that contains two methods:
  - onDataChange
    - initially called with a snapshot of the database.
    - subsequently called with a snapshot whenever a change is made to the database.
  - onCancelled
    - called when a database error is detected on the serve.r

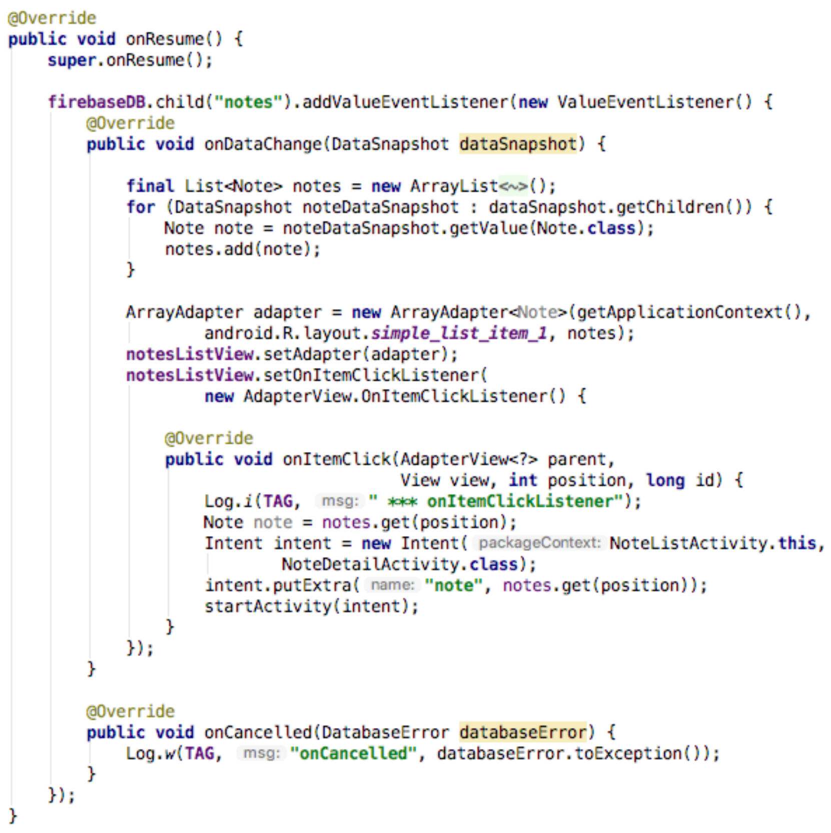
Notice that “onDataChange” performs the following tasks:

- Creates an ArrayList of notes a and displays them in the ListView
- Creates an OnItemClickListener to respond to clicks on notes in the ListView and transitions the user to the NoteDetailActivity

Once the above tasks are completed, we can run the app to create, retrieve, update and delete user notes in Google Firebase. We can test the note functionality using an Android device or an Android Studio emulator. Figure 7 shows a sequence of screenshots depicting the mobile app in action with Google Firebase.

4 Results

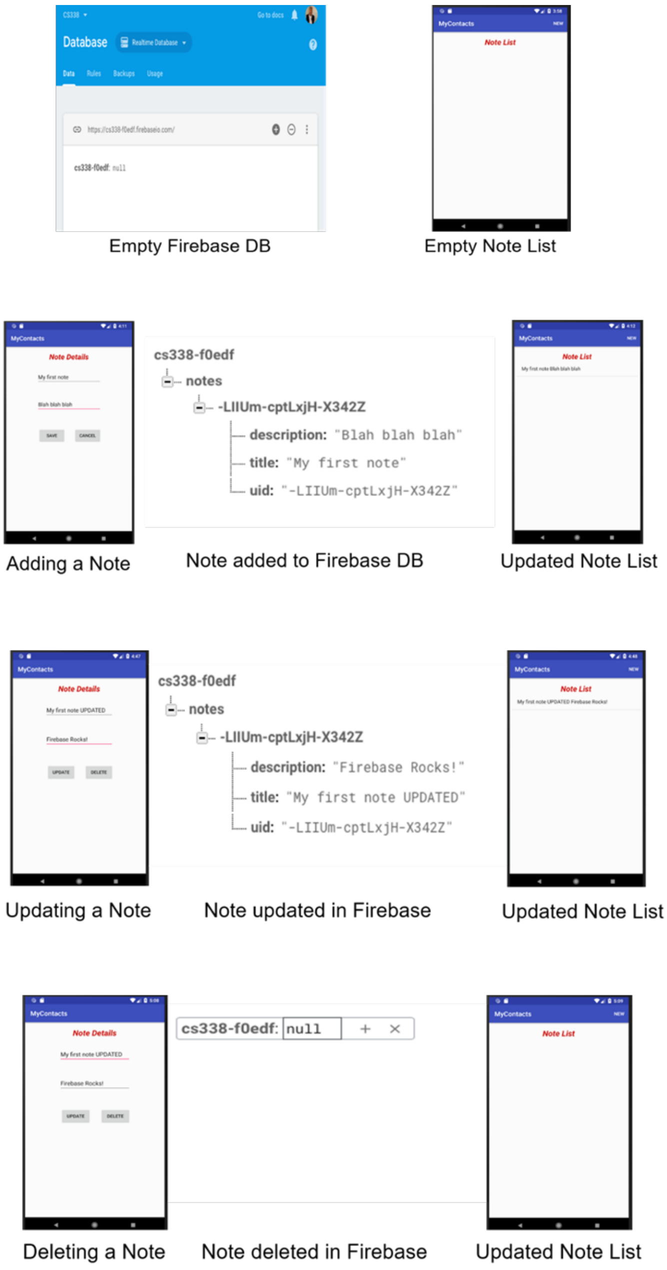
The DaaS Notes example, as illustrated in the previous section, has been taught in four course offerings in the 2018/2019 academic year. Three of the courses were online and one of the courses was classroom-based. In all four courses the material was well received and successfully executed. The students gained hands-on experience in using cloud-based storage and were able to articulate the advantages/disadvantages of cloud vs. local storage.



```
@Override
public void onResume() {
    super.onResume();

    firebaseDB.child("notes").addValueEventListener(new ValueEventListener() {
        @Override
        public void onDataChange(DataSnapshot dataSnapshot) {

            final List<Note> notes = new ArrayList<~>();
            for (DataSnapshot noteDataSnapshot : dataSnapshot.getChildren()) {
                Note note = noteDataSnapshot.getValue(Note.class);
                notes.add(note);
            }

            ArrayAdapter adapter = new ArrayAdapter<Note>(getApplicationContext(),
                    android.R.layout.simple_list_item_1, notes);
            notesListView.setAdapter(adapter);
            notesListView.setOnItemClickListener(
                    new AdapterView.OnItemClickListener() {

                @Override
                public void onItemClick(AdapterView<?> parent,
                                        View view, int position, long id) {
                    Log.i(TAG, msg: " *** onItemClickListener");
                    Note note = notes.get(position);
                    Intent intent = new Intent( packageContext: NoteListActivity.this,
                            NoteDetailActivity.class);
                    intent.putExtra( name: "note", notes.get(position));
                    startActivity(intent);
                }
            });
        }

        @Override
        public void onCancelled(DatabaseError databaseError) {
            Log.w(TAG, msg: "onCancelled", databaseError.toException());
        }
    });
}
```

5 Conclusions

Cloud computing is a "next generation" evolution of distributed enterprise computing that has been steadily gaining popularity over the last ten years. Current market trends indicate that the adoption of cloud techniques will continue to grow and be a skill set that will always be in high demand. Educational curriculum needs to address this market trend. In this paper we identified a starting point for incorporating cloud computing in the classroom, provided an end-to-end illustrative example, and an evolutionary path for enhancing its delivery. Failure to meet this need will be a missed opportunity and a disservice to aspiring CS students.



```
cs338-f0edf: null
```

```
cs338-f0edf
  notes
    -LlIUm-cptLxjH-X342Z
      description: "Blah blah blah"
      title: "My first note"
      uid: "-LIIUm-cptLxjH-X342Z"
```

```
cs338-f0edf
  notes
    -LlIUm-cptLxjH-X342Z
      description: "Firebase Rocks!"
      title: "My first note UPDATED"
      uid: "-LIIUm-cptLxjH-X342Z"
```

```
cs338-f0edf: null
```

Figure 7: User notes mobile app in action.

References

[1] ACM/IEEE. Computer science curricula 2013 - curriculum guidelines for undergraduate degree programs in computer science. *The Joint Task Force on Computing Curricula, Association for Computing Machinery (ACM) IEEE Computer Society*, 2013.

[2] Ling Chen, Yang Liu, Marcus Gallagher, Bernard Pailthorpe, Shazia Sadiq, Heng Tao Shen, and Xue Li. Introducing cloud computing topics in curricula. *Journal of Information Systems Education*, 23(3):315, 2012.

[3] Louis Columbus. 83% of enterprise workloads will be in the cloud by 2020, 2018. `https://www.forbes.com/sites/louiscolumbus/2018/01/07/83-of-enterprise-workloads-will-be-in-the-cloud-by-2020/#1f71f2a86261`.

[4] Debzani Deb, Muztaba Fuad, and Keith Irwin. A module-based approach to teaching big data and cloud computing topics at cs undergraduate level. In *Proceedings of the 50th ACM Technical Symposium on Computer Science Education*, pages 2–8. ACM, 2019.

[5] Dahai Guo and Anna Koufakou. A comprehensive and hands-on undergraduate course on cloud computing. In *Proceedings of the ASEE Southeastern Section Conference*, 2018.

[6] M Suhail Rehman, Jason Boles, Mohammad Hammoud, and Majd F Sakr. A cloud computing course: from systems to services. In *Proceedings of the 46th ACM Technical Symposium on Computer Science Education*, pages 338–343. ACM, 2015.

[7] M Vijayalakshmi, Mahesh S Patil, Aruna S Nayak, Vidya S Handur, and GS Hanchinamani. Enhancing students learning skills through integrated course project design model (icpdm). In *2018 IEEE 18th International Conference on Advanced Learning Technologies (ICALT)*, pages 30–33. IEEE, 2018.

Improving Abstraction through Python Programming in Undergraduate Computer Science and Math Classes*

Jay L. Jackson[1], *Cynthia L. Stenger*[1],
James A. Jerkins[2], *Mark G. Terwilliger*[2]
[1]*Mathematics Department*
[2]*Computer Science and Information Systems*
University of North Alabama
Florence, AL 35630
{jljackson3,clstenger,jajerkins,mterwilliger}@una.edu

Abstract

Stenger et al. [8] developed an instructional model that can be applied in any STEM classroom to use computer programming to explore the essential characteristics of a math or science concept and to push learners to advance in levels of abstraction. After promising results in a study with high school teachers (N=25), the research team increased the scale and scope of the project to include undergraduate students (N=144) in seven lower-division computer science and math classes including a control group. Results showed that most of the students began the class with a pre-action knowledge level. Of the students in lessons using computer programming activities, 38% improved to a process level or higher, while only 15% of students in the control group reached a process level or higher. The data demonstrated how computer programming activities helped students form mental images and pushed them to higher levels of abstraction.

1 Introduction

There is widespread agreement among computer science educators that abstraction and generalization are essential for success in computer science. For

*Copyright ©2019 by the Consortium for Computing Sciences in Colleges. Permission to copy without fee all or part of this material is granted provided that the copies are not made or distributed for direct commercial advantage, the CCSC copyright notice and the title of the publication and its date appear, and notice is given that copying is by permission of the Consortium for Computing Sciences in Colleges. To copy otherwise, or to republish, requires a fee and/or specific permission.

example, the ability to abstract is essential when designing, implementing, and using object oriented programming. Another example is decomposing specific tasks into generalized behaviors in computational thinking. However, generalization and abstraction are not often explicitly addressed in either the P-12 curriculum or in introductory computer science courses. Students are generally expected to acquire these mental habits of mind on their own.

Other STEM disciplines also require students to develop these critical cognitive tools for success. The historical setting for learning these habits is in the mathematics classroom. Many researchers have proposed methods for explicitly inducing students to develop generalization in the context of mathematical explorations [3, 9]. However, a recent trend towards emphasizing mathematics as a purely computational task in many P-12 settings has robbed students of the opportunity to develop these habits. This has led to what is often called a "lack of mathematical maturity" in first year college students and an increase in the number of students placed into remedial math courses.

Many education researchers believe that utilizing computer programming activities designed to parallel the construction of an underlying mathematical process can act as a catalyst to developing the associated construction [3]. In 2011, Stenger et al. developed an instructional model that uses computer programming activities to promote mathematical reasoning and explicitly teach generalization and abstraction [7, 5, 4]. The computer programming exercises are designed to stimulate the development of mental frameworks that support the transfer of learning in a mathematics context, specifically abstraction and generalization. The approach is modeled on constructivist pedagogy and uses computer programming as a laboratory for exploring the behavior of a specific concept under investigation.

2 Instructional Model

Over six years of a Math/Science Partnership (MSP) grant, Stenger et al. designed and delivered lessons using the model on a variety of mathematical concepts. The objectives of the MSP grant were to improve STEM teaching in Alabama through professional development (PD) for practicing teachers and integrate authentic computer science activities into math classrooms. Math, computer science, and science teachers from 5th to 12th grade participated in PD. Results from these PD sessions demonstrated that teachers and their students benefited from the approach [6].

Using this instructional model, the researchers described the effectiveness of a geometry lesson with a robotics activity for a high school teachers' PD [10]. Feedback and comments from this report included questions about efficacy of the instructional model with college undergraduates. Several small sample studies have been conducted with college undergraduates [8]; however, a large study with a control group has not been undertaken. This study addresses the gap by utilizing a rigorous design with a control group and a large sample size to demonstrate the effectiveness of the instructional model with college undergraduates.

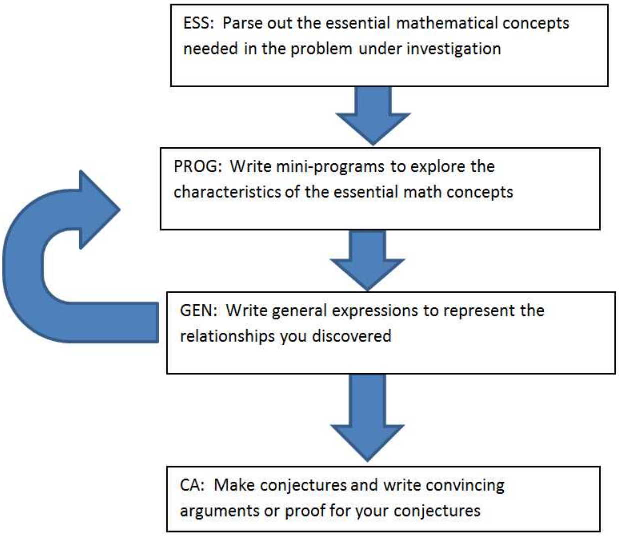

ESS: Parse out the essential mathematical concepts needed in the problem under investigation
PROG: Write mini-programs to explore the characteristics of the essential math concepts
GEN: Write general expressions to represent the relationships you discovered
CA: Make conjectures and write convincing arguments or proof for your conjectures

Figure 1: The Instructional Model

As shown in Figure 1, the instructional model is composed of four primary phases: documenting essential characteristics (ESS), designing programming explorations (PROG), discovering general expressions (GEN), and making conjectures and convincing arguments (CA). Designing a lesson begins with an initial decomposition of the concept to be learned. This decomposition focuses on what the learner needs to understand in order to begin building a conceptual model of the topic. For example, if the topic is even/odd numbers, learners need to understand the concepts of integer, consecutive integer, and the modular operation. The next stage of lesson development is where computer programming activities are created that will guide the learner towards abstract reasoning and generalization about the concept under study. In addition to the programming activities, motivating questions are created for participants to answer during their exploration. Special attention is paid to the form of statements in the programming activities so that the general expressions are a natural outcome of the exercise. Finally, in the lesson development, conjectures and proofs are created using the general expressions. Learners will be guided into making conjectures during the lesson and, using the general expression they have discovered, making convincing arguments about their conjectures.

Once developed, the format of the delivered lesson is as follows. First a pre-test over the concept is taken by all learners. Next, an introduction to using the programming tool is presented by the lesson facilitator. Typically Python and IDLE are used; however, C++ has also been used with undergraduates [4]. Relevant background and definitions are presented by the facilitator along with the initial programming activity. Next, the programming activities and motivating questions are performed, guided by the facilitator, until the necessary general expressions are uncovered by the participants. The PROG phase is repeated in order to discover as many relationships and general expressions as possible. Finally, a conjecture is made by the facilitator along with a convincing argument using a general expression from the programming activities. Learners are then asked to make their own conjectures and convincing arguments. At the conclusion of the lesson, participants take a post-test. In addition to the pre- and post-tests, student responses are collected throughout the PROG, GEN, and CA phases of the lesson. All of the collected data is analyzed using a framework based on APOS [1].

In APOS theory the mental structures are Action, Process, Object, and Schema. APOS theory was developed as a way to describe the mental constructions held by learners about a concept [1, 2]. Actions are interiorized as Processes, and Processes are encapsulated to mental Objects. APOS practitioners do not believe the progression of a learner is linear through the constructs, rather learners may move back and forth as well as hold positions between levels. In our study we classified participants' conception of the topic under study as pre-Action, Action, Process, or Object with a scale of zero to three. Using this framework, we created a genetic decomposition of the lesson topic — angles of regular polygons — that was used as a rubric to evaluate responses.

3 Experiment

The genetic decomposition, shown in Table 1, motivated the design of this research study and the analysis of the results. It was also the basis for the computer activities in the lessons that were developed.

The researchers applied the instructional treatment to the concept of interior and exterior angles in polygons to an experimental group consisting of CS0 students (N=68) and a geometry content class for elementary education majors (N=29). A control group (N=47) of "Mathematical Reasoning for the Arts" students covered the same material in a traditional manner with instructional

Table 1: Genetic decomposition for interior and exterior angles of polygons

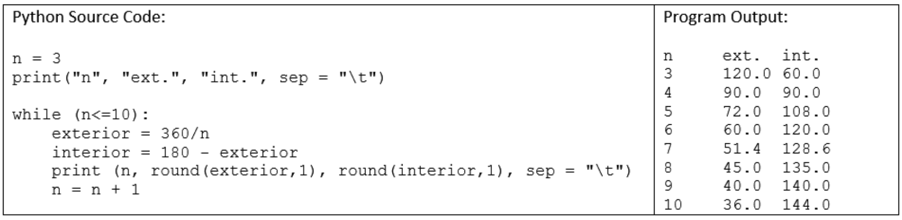
Pre-Action	Unfamiliar with polygons / interior angles / exterior angles
Action	Can find measures of interior/exterior angles with a formula; may not see relationship between interior / exterior angles
Process	Can visualize what happens to interior / exterior angles as the number of sides in a polygon increases, allowing them to generalize the behavior of how the relationship comes to be
Object	Might imagine interior / exterior angles as a linear pair, recognizing them as one entity; thus, if one angle has a measure of α, then they know that the other has a measure of $(180-\alpha)$. They might see the process (of increasing side numbers) as a totality, imagining the sides moving toward an infinite number (and even completing the process), where the polygon is now a "circle".

lectures, a classroom activity, and worksheets. The elementary education major students were the only ones who had a prerequisite mathematics class, either Pre-Calculus Algebra or Finite Math. For the experimental groups, the instructional treatment covered three 75-minute class periods; the control group covered the material in two 75-minute periods with the post-test given in the next class meeting. All participants completed a pre-test before any instruction and a post-test once the instruction was complete. The experimental groups also completed response sheets at various points during their instruction.

In all groups, a brief lecture on the definition and characteristics of a polygon began the instructional process. Examples of polygons were shown and on each, the interior and exterior angles were highlighted and defined. An emphasis was then placed on exterior angles, and each was highlighted on an image of a hexagon. At this point, the instruction for the experimental and control groups diverged. For the experimental groups, the action of each exterior angle being moved together to form a circle was described; the control group was given a handout of the picture and told to cut out each exterior angle and put them together, allowing them to "discover" that they formed a circle.

The control group instruction continued with a traditional lecture about curves and polygons in the plane including curves and regions, convex and concave figures, polygonal curves, triangles, quadrilaterals, and regular polygons. A worksheet was given after the instruction which emphasized the classification of triangles, polygon identification, the measure of individual interior and exterior angles in a polygon, and the sum of the measures of all interior and exterior angles in a regular polygon.

For the experimental groups, the instructional treatment began with a brief introduction to Python programming. Learners wrote programs to produce tables depicting the measure of exterior and interior angles in polygons, iterating the number of sides, ultimately resulting in the following code and output shown in Figure 2.



Python Source Code:

```
n = 3
print("n", "ext.", "int.", sep = "\t")

while (n<=10):
    exterior = 360/n
    interior = 180 - exterior
    print (n, round(exterior,1), round(interior,1), sep = "\t")
    n = n + 1
```

Program Output:

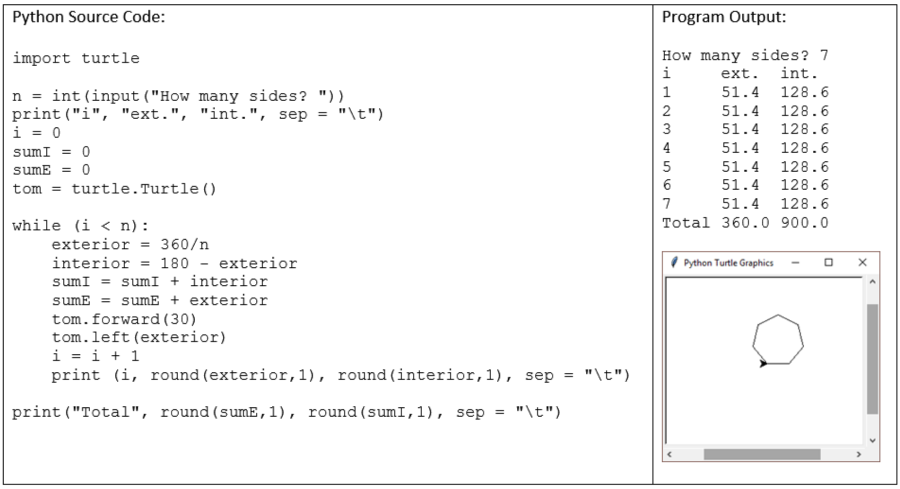
```
n       ext.  int.
3       120.0 60.0
4       90.0  90.0
5       72.0  108.0
6       60.0  120.0
7       51.4  128.6
8       45.0  135.0
9       40.0  140.0
10      36.0  144.0
```

Figure 2: Computer program to generate polygon angles

Participants were asked to add rows to their programs to show what the measure of a regular polygon's exterior would be as the number of sides increased. Learners were encouraged to experiment with their computer programs and make observations about any relationships. Once the initial table was constructed, the participants were ushered through a series of program modifications and written responses, including writing a convincing argument (i.e., proof) that as the number of sides of a polygon increases, the measure of each of its exterior angles decreases. Participants wrote responses to questions and reflections on their observations on response sheets, including generalizations of behavior observed in their programs. Observations on the relationship between the number of sides of a polygon and the measure of its exterior angles were solicited, and participants were taught how to denote the general expressions in mathematical language.

Next, the students were instructed to tinker with their programs in order to produce a third column, one that depicted the measure of interior angles in a regular n-gon. Learners wrote programs to show those measures and were then encouraged to experiment with their programs and make observations about any relationships. Once again, response sheets solicited general expressions in the code that showed the relationship between the number of sides in a polygon and the measure of its interior angles. Students were then shown a convincing argument using those general expressions, that as the number of sides in a regular polygon increases the measure of its interior angles increases. The instructional treatment was designed so that repetition with various program modifications would stimulate the desire to generalize the observed behavior and make conjectures about the mathematical construct.

The next stage of the lesson involved using the Python Turtle module to draw various polygons by utilizing turns equal to the degree of the exterior angles. The code developed and the corresponding output is shown in Figure 3.


Python Source Code:

```
import turtle

n = int(input("How many sides? "))
print("i", "ext.", "int.", sep = "\t")
i = 0
sumI = 0
sumE = 0
tom = turtle.Turtle()

while (i < n):
    exterior = 360/n
    interior = 180 - exterior
    sumI = sumI + interior
    sumE = sumE + exterior
    tom.forward(30)
    tom.left(exterior)
    i = i + 1
    print (i, round(exterior,1), round(interior,1), sep = "\t")

print("Total", round(sumE,1), round(sumI,1), sep = "\t")
```

Program Output:

```
How many sides? 7
i      ext.  int.
1      51.4  128.6
2      51.4  128.6
3      51.4  128.6
4      51.4  128.6
5      51.4  128.6
6      51.4  128.6
7      51.4  128.6
Total  360.0 900.0
```

Figure 3: Summing polygon angles and drawing shapes using Turtle graphics

Students in the study were enthusiastic as they input larger numbers of sides for their polygon. Several enjoyed the challenge of trying to input the largest number of sides that would still print the entire polygon on their display. Almost all students thought it was “cool” and “interesting” that by inputting very large numbers of sides, the image produced by the Turtle program looked exactly like a circle. The “Total” columns in the program emphasized for the learners the point that the sum of the exterior angles in a regular polygon is always 360 degrees while the sum of the interior angles continues to increase relative to the number of sides.

All of the participant’s responses were collected on response sheets during the lesson. All of the collected data from pre-tests and post-tests was reviewed and scored using APOS theory as a guide. A ranked set of scores was devised to denote pre-Action (0), Action (1), Process (2), and Object (3) levels based on the genetic decomposition and recorded for each subject’s submissions. Three scorers ranked each participant. In the event that authors disagreed, a discussion and further analysis of the data was used to reach consensus.

4 Results

From the total pool of students who participated in the lessons (N=144), 23 were removed due to missing a pre- or post-test. Data in Table 2 summarizes the results for the remaining students (N=121).

Table 2: Pre-test and post-test results

	Control Group (n=40)		Experimental Group (n=81)	
APOS Level	PRE	POST	PRE	POST
Pre-Action	40	19	76	25
Action	0	15	4	25
Process	0	6	0	26
Object	0	0	1	5

Before instruction, the control and experimental groups showed similar mastery of the geometry concept. Only one out of 81 (1.2%) students in the Experimental Group was at Process level or above and none (0%) of the Control group was at the Process level or above. After instruction, 38.3% of the Experimental group were at the Process level or above and 15% of the undergraduates in the Control Group scored at the Process level or above.

These results indicate that even though the control group was similar in capability, it did not make the same improvements as the experimental group. This suggests that the instructional model did indeed improve students' ability to generalize and abstract over the concept, and supports the pursuit of future randomized experimentation.

5 Conclusion

Students who participated in the Python and Turtle programming activities showed higher levels of abstraction after instruction over the targeted geometry concept than the students who were presented a traditional lecture format lesson. The iterative nature of the programming motivated students to see the general expressions and influenced their ability to employ abstraction in their mental images. Students who started at the pre-action level were able to transition to a process conception. As in the previous study with P-12 teachers, the programming activities influenced undergraduate students and served as a catalyst to move from purely English descriptions of their conceptions to using mathematical symbols, and the activities put them into a mindset of generalizing. The results of this study can facilitate further analysis of using computer programming to overcome cognitive obstacles in students' understanding of geometry.

References

[1] Mark Asiala, Anne Brown, David J DeVries, Ed Dubinsky, David Mathews, and Karen Thomas. A framework for research and curriculum development in undergraduate mathematics education. *Maa Notes*, pages 37–54, 1997.

[2] Mark Asiala, Anne Brown, David J. Devries, David Mathews, and Karen Thomas. A framework for research and curriculum development in undergraduate mathematics education. In *Research in Collegiate Mathematics Education*, pages 1–32. American Mathematical Society, 1996.

[3] Ed Dubinsky and David Tall. *Advanced Mathematical Thinking and the Computer*. Springer, Dordrecht, 2002.

[4] Janet T. Jenkins, James A. Jerkins, and Cynthia L. Stenger. A plan for immediate immersion of computational thinking into the high school math classroom through a partnership with the alabama math, science, and technology initiative. In *Proceedings of the 50th Annual Southeast Regional Conference*, ACM-SE '12, pages 148–152, New York, NY, USA, March 2012. ACM.

[5] Cynthia Stenger, Janet Jenkins, James A. Jerkins, and Jessica Stovall. An Explicit Method for Teaching Generalization to Pre-Service Teachers Using Computer Programming. In *Proceedings of the 20th Annual Conference on Research in Undergraduate Mathematics Education (MAA SIGMAA on RUME 2017)*, February 2017.

[6] Cynthia Stenger and James A. Jerkins. Using Computer Programming to Improve Mathematical Thinking: A Preliminary Report. In *Proceedings of the 2017 Joint Mathematics Meeting*, January 2017.

[7] Cynthia Stenger, James A. Jerkins, Janet Jenkins, and Jessica Stovall. Using Computer Programming to teach Mathematical Generalization and Proof. In *Proceedings of the 13th International Congress on Mathematical Education (ICME-13)*. International Commission on Mathematical Instruction, July 2016.

[8] Cynthia Stenger, James A. Jerkins, Jessica Stovall, and Janet Jenkins. An APOS Study on Undergraduates' Understanding of Direct Variation: Mental Constructions and the Influence of Computer Programming. In *Proceedings of the 21st Annual Conference on Research in Undergraduate Mathematics Education (MAA SIGMAA on RUME 2018)*, February 2018.

[9] D. Tall. *Advanced Mathematical Thinking*. Mathematics Education Library. Springer Netherlands, 2006.

[10] Mark G Terwilliger, Jay L Jackson, Cynthia L Stenger, and James A Jerkins. Using computer programming activities and robots to teach generalization of a geometry concept. *Journal of Computing Sciences in Colleges*, 34(3):82–90, January 2019.

Teaching Relational Database Normalization in an Innovative Way*

Mohammad Amin[1], Gordon W. Romney[2]
Pradip Dey[1] and Bhaskar Sinha[1]
[1] Department of Engineering and Computing
National University
San Diego, CA 92123
[2] Center for Cyber Security Engineering and Technology
University of San Diego
San Diego, CA 92110
{mamin,3pdey,4bsinha}@nu.edu, 2gromney@sandiego.edu

Abstract

Teaching relational database normalization is a challenging and critical task. The authors of this paper have taught database design for many years and have identified reasons why students find normalization to be difficult to understand and why they fail to see the systematic, rule-based normalization process. Recently, the authors have designed and developed an improved functional dependency diagram where two new innovative items have been added: the Anti Partial Dependency (APD), and the Double Headed Arrow ($\updownarrow$). Traditional functional dependency diagrams can show Partial Dependencies and Transitive Dependencies, but dependencies related with Boyce Codd Normal Form and Fourth Normal Form have no graphical representation and are only explained in words. The new improved functional dependency diagram has the capability to graphically show all the dependencies up to Fourth Normal Form. One of the authors used these new functional dependency descriptive features to teach normalization in several classes and observed remarkably positive outcomes on teaching and learning. This paper describes this new innovative teaching approach for normalization and presents evidence of improved student learning outcomes.

*Copyright ©2019 by the Consortium for Computing Sciences in Colleges. Permission to copy without fee all or part of this material is granted provided that the copies are not made or distributed for direct commercial advantage, the CCSC copyright notice and the title of the publication and its date appear, and notice is given that copying is by permission of the Consortium for Computing Sciences in Colleges. To copy otherwise, or to republish, requires a fee and/or specific permission.

Introduction

Designing and building a good Relational Database Management System (RDBMS) can be very challenging but the satisfaction and reward are definitely worth the effort. From an information management perspective, possibly the most destructive problems are introduced by uncontrolled data redundancy, that create data integrity problems. A database can be structured in a better way if it is first analyzed using the data dependency diagram and then properly normalized. In the case of a large table, the normalization process may be used to separate redundant data into different groups and produce new, smaller tables.

A recent, published article addressed some pertinent issues that contribute to database complexity and can be used to estimate the extent to which a database is complex [11]. In other literature, rudimentary visual teaching tools for database analysis were introduced [1, 12]. Some efforts towards automated normalization were also described in the literature [9, 6]. This paper addresses some related issues on teaching normalization of relational databases and provides an innovative method of teaching normalization. Normalization has become a standard in the relational database area for measuring the database quality and the approach is specified by others [2, 3, 4, 7, 8, 10, 5]. Database designers use this technique during the design phase for eliminating data redundancy and reorganizing data for improvement. The level (degree) of normalization of a database indicates how properly a given database is designed [3, 4, 7]. In relational databases, the functional dependency diagrams (FDD) help to understand the level of data redundancies and anomalies. Normalization produces accurate information with increased data integrity by minimizing data redundancy and by eliminating anomalies.

The word "anomaly" in a database means abnormality of data. If a database contains abnormal data then it has at least one data anomaly. Redundant data generates the most easily identified abnormal data in a database. There are three major types of data anomalies: a) Modification Anomaly, b) Insertion Anomaly, and c) Deletion Anomaly. The differences and details of these are found in literature [3, 4, 7]. Data integrity and data anomaly problems are tightly coupled with each other. Data integrity indicates the consistency and accuracy of data in a database. There are three types of data integrity: Entity integrity, Domain integrity, and Referential integrity [3, 4, 7]. Data integrity, according to Codd, assures that a normalized database operated on by standard-compliant SQL operators will always produce a normalized product. Simply stated, "If Humpty-Dumpty is taken apart, Humpty-Dumpty can be re-assembled." If, however, multiple instances of data are saved in different tables then, most likely, a data anomaly will occur in the database [3, 4, 7]. It is most important that a database meet all the data integrity conditions.

RDBMSs use formal mathematical set theory and relational algebra operations for data manipulation, extraction and union to produce meaningful information. Codd's relational model is based on Abelian commutative group theory which makes it possible to maintain the integrity of the virtual "Humpty-Dumpty". There are eight basic relational operators that can be used in a relational database: 1) SELECT, 2) PROJECT, 3) JOIN, 4) INTERSECT, 5) UNION, 6) DIFFERENCE, 7) PRODUCT, and 8) DIVIDE [3, 4, 7]. Most of the RDBMSs support the first three operators (SELECT, PROJECT and JOIN), but only standards-compliant systems support all eight operators. When a relational operator acts upon a relation it produces a new relation and this property is known as a closure property. In a relational database, the rows are treated as sets for data manipulation or processing.

In a relational database, the concept of functional dependencies (FD) is used for normalization purposes. There are many different types of functional dependencies as mentioned in the literature. In this paper, the following four are discussed: 1) Partial Dependency (PD), 2) Transitive Dependency (TD), 3) Anti-Partial Dependency (APD) (this a new term used by the authors), and 4) Multi-valued Dependency (↟MVD). This ↟ is a symbol used for multiple values of data in an attribute.

Partial Dependency (PD): If a database contains a composite primary key then it has one or more partial dependencies. When a nonprime or nonkey attribute is identified by a part of the composite primary key then this dependency is known as a partial dependency.

Transitive Dependency (TD): When a nonprime attribute determines another nonprime attribute then that dependency is known as a transitive dependency.

Anti-Partial Dependency (APD): The term "Anti-Partial Dependency" is an innovative term that has been introduced in this paper. This kind of dependency exists in a database only when a nonprime (nonkey) attribute determines part of the key attribute.

Multi-valued Dependency (↟ MVD): If an attribute has more than one value then it has multi-valued dependency. The double headed arrow (↟) in the FDD indicates that the attribute has multiple values.

In a relational database, some or all of these dependencies may exist. In some cases, some other types of dependency, also, may exist. In general, most business databases are designed to comply with 3NF ("3-Normal-Form") or 4NF ("4-Normal-Form). A database normalized to a higher normal form automatically meets all the conditions of lower normal forms. For example, a 4NF database also meets all the conditions of BCNF, 3NF. 2NF and 1NF. Relations should be normalized to the highest possible normal form but a designer must consider the associated cost implications. The ultimate purpose of data modeling is to create a well-structured, efficient, accurate and easy-to-use database. The five commonly used rules found in database normalization processes are [3, 4, 7]:

1. Elimination of repeating groups: make a separate table for each set of related attributes and give each table a primary key.
2. Elimination of redundant data (duplicate data): if an attribute depends on only part of a multi-valued key, remove it to a separate table.
3. Elimination of columns not dependent on the key attribute: if attributes do not contribute to a description of the key, remove them to a separate table.
4. Isolation of independent multiple relationships: no table may contain two or more 1:n or n:m relationships that are not directly related.
5. Isolation of semantically related multiple relationships: there may be practical constraints on information that justify separating logically related many-to-many relationships.

An Example with Various Dependencies

The Functional Dependency Diagram (FDD) represents the header of a relation (table) and all the boxes in the FDD with labels represent the attributes (columns) of that relation. Figure 1 is an example of the FDD of a Student-Record Table in a database called Academic. The primary key of this table is a composite key and consists of two attributes: SEMAIL and CLASS ID (underlined). This table satisfies all eight characteristics of a relation and it satisfies the condition of First Normal Form (1NF). But this FDD does not contain any data dependency.

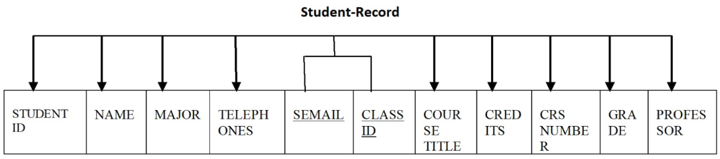


Figure 1: Student-Record

Step-1: Inspection of data types shows that this FDD can be refined with identifications of all types of data dependencies. Figure 2 is the new improved FDD. This relation has two PDs, one TD, one APD and one MVD. This needs to be normalized.
Step-2: When these PDs are removed by normalization (decomposing table), a set of new relations (tables) is created. The new database tables, shown in Figure 3, is considered in 2NF.

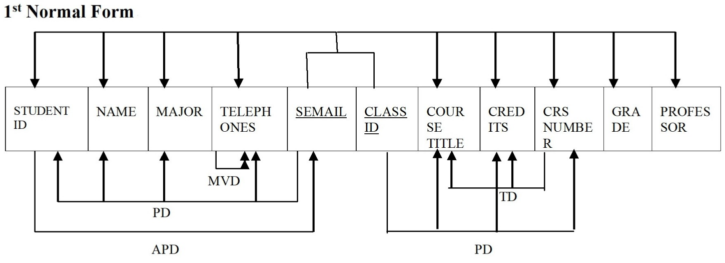


Figure 2: 1NF Academic Database

Step-3: The database can be upgraded to 3NF if TD is removed by decomposing the Class table. Figure 4 shows the new set of tables after removal of TD from the Class table.

Step-4: Anti-Partial Dependency (APD) is a new innovative term introduced by the authors. This kind of dependency exists only when a nonprime (non-key) attribute determines a key attribute. All textbooks and literature explain this dependency with a sentence or paragraph. In most cases, the beginners of database design have difficulty understanding the meaning of this dependency. The authors identified the learners' challenges and provide a visual symbol labeled as APD in the FDD. This innovative method has helped enhance students' learning of this dependency. This 3NF database has an APD in the Student table and it can be removed. The new database can be refined to BCNF by removing this APD. Figure 5 shows the new database which meets BCNF rules.

Step-5: Multi-Valued Dependencies (MVD): A relation may contain one or more Multi-valued attributes. In this example, the "Telephones" attribute has several values (cell, home, work). When a relation has a multi-valued attribute then it has multi-valued dependency. Usually, this type of dependency is not shown pictorially in the FDD. These authors started using a new technique to indicate this MVD in the FDD. In Figure 5, the Student table has an MVD and has been labeled with a double headed arrow. This double headed arrow is a new approach to label the MVD attribute. If this MVD is removed from a database then the new database satisfies the Fourth Normal Form (4NF). An MVD can be removed from a table in two different ways: 1) create more attributes (columns) in the table and 2) create a new table for this MVD attribute. In this example, the MVD has been removed by creating three columns in the Student table. Figure 6 shows the new database which satisfies the 4NF condition.

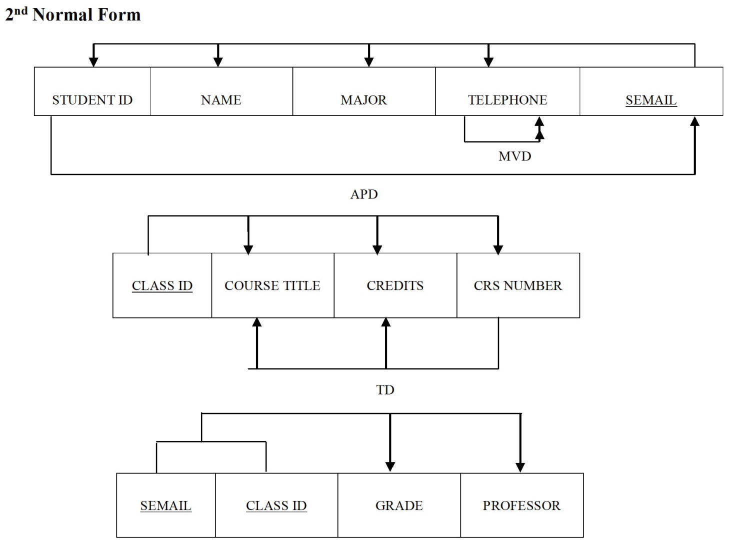


Figure 3: 2NF Academic Database

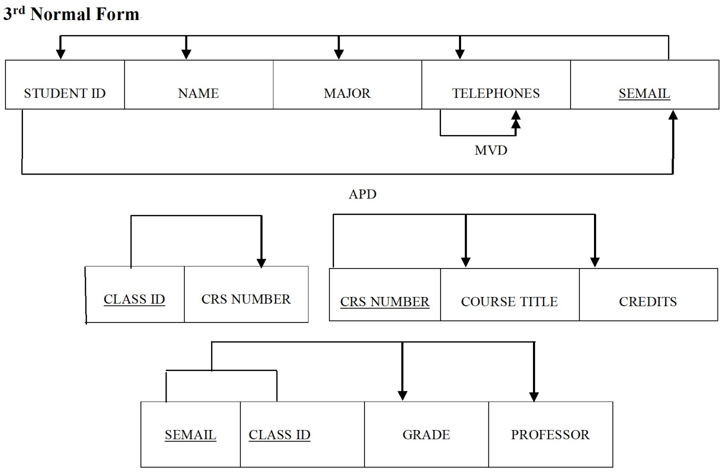


Figure 4: 3NF Academic Database

1 Discussions

One of authors of this paper, has taught database classes several times using these innovative constructs: Anti Partial Dependency (APD) and symbol of

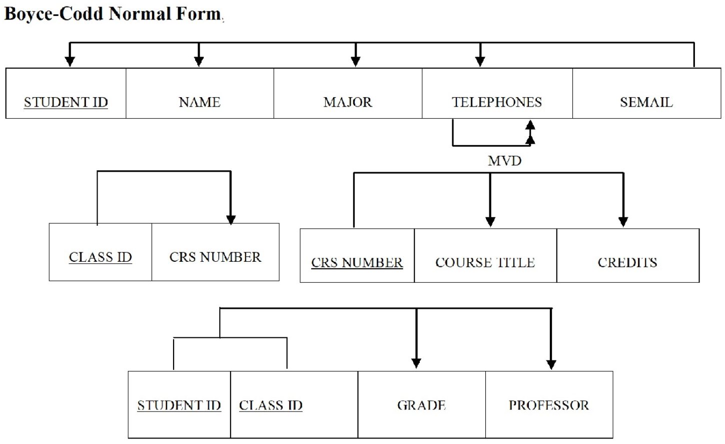


Figure 5: BCNF Academic Database

double headed arrow (↨) for Multi-Valued Dependency representation. The impact of using this new method of teaching normalization was found to be very positive. Students appreciated this innovative teaching method and expressed their positive experiences in learning at the end of course evaluations, especially in 2016 as the method matured and other aspects of teaching remained basically the same. Following are some end of course evaluations and students' comments on teaching copied from the university records.

1) "Dr. X is a very good as a teacher and also as an advisor he taught me many things which will be helpful for my career and life. His way of teaching is different and can be easily grasped. I want to thank national university for providing such a knowledgeable person." 2)"The method of teaching was good and good live examples which made me to learn all concepts" 3)"I really enjoyed this class by learning new things. Instructor is awesome in explanation. His way of teaching is too good for example he teaches from general examples which helps students who not even know about the subject. By this me learn more and more subject in an efficient way." 4) "Hello. This is a wonderful experience learning from this professor the way of teaching the students by real time examples taught me more knowledge about the databases the practical was very good. I gained real experience in database programming. I wish to work with him in future. Thank you" 5) "The professor's method of teaching was excellent. He teaches with examples. The lectures has clearly helped me to know about the subject significantly." 6) "The instructor is very knowledgeable and helps students in any way possible. He is very inspiring and a great asset to National University. The course itself is too advanced for students

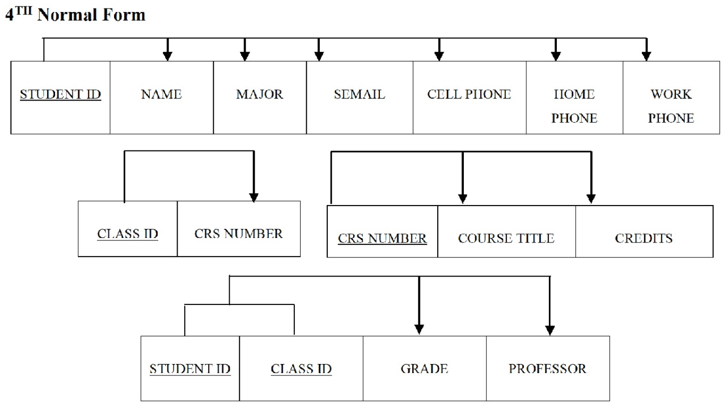


Figure 6: 4NF Academic Database

Course (NU Catalog)	**Student self-assessment of Learning out of 5.00**	**Student Assessment on Teaching out 5.00**	**Month/Term**
DAT604	4.90	4.92	December 2016
DAT604	4.50	4.84	December 2015
DAT604	4.60	4.65	April 2015
DAT604	4.25	4.47	December 2014

Figure 7: End of Course Evaluation Scores (Note: all courses at National University are offered in a one-month format www.nu.edu)

with no database knowledge. I just making this class an elective instead of a requirement for MIS majors."

2 Conclusion

The authors are concluding that using this innovative method of teaching has great impacts on students' perception of learning. By repeating this method, the instructor also gained more experience and ratings of both teaching and learning were gradually increased with higher appreciations.

3 Acknowledgment

Authors are grateful to anonymous conference reviewers for their constructive suggestions.

References

[1] M. Amin, G. Romney, P. Dey, B. Sinha, and D. Bowen. Teaching and learning of database concepts using multimode teaching methodologies. pages 24–31. `http://www.asee.org/papers-and-publications/papers/zone-proceedings/zone4`.

[2] Edgar F Codd. A relational model of data for large shared data banks. *Communications of the ACM*, 13(6):377–387, 1970.

[3] Carlos Coronel and Steven Morris. *Database systems: design, implementation, & management*. Cengage Learning, 2016.

[4] Christopher John Date. *An introduction to database systems, 8th Ed.* Pearson, 2003.

[5] Moussa Demba. Algorithm for relational database normalization up to 3nf. *International Journal of Database Management Systems*, 5(3):39, 2013.

[6] YV Dongare, PS Dhabe, and SV Deshmukh. Rdbnorma:-a semi-automated tool for relational database schema normalization up to third normal form. *arXiv preprint arXiv:1103.0633*, 2011.

[7] Ramez Elmasri and Sham Navathe. *Fundamentals of database systems*. Pearson, 2017.

[8] Marin Fotache. Why normalization failed to become the ultimate guide for database designers? *Available at SSRN 905060*, 2006.

[9] Amir Hassan Bahmani, M Naghibzadeh, and Behnam Bahmani. Automatic database normalization and primary key generation. pages 000011 – 000016, 06 2008.

[10] William Kent. A simple guide to five normal forms in relational database theory. *Commun. ACM*, 26(2):120–125, 1983.

[11] Bhaskar R Sinha, Gordon W Romney, Pradip P Dey, and Mohammad N Amin. Estimation of database complexity from modeling schemas. *Journal of Computing Sciences in Colleges*, 30(2):95–104, 2014.

[12] Lu Zhang, Mudasser Wyne, Alireza Farahani, Bhaskar Sinha, and Mohammad Amin. Visual learning tool for teaching entity relationship mapping rules. 04 2015.

Taking Active Learning to an Online Environment[*]

Linda DuHadway
School of Computing
Weber State University
Ogden, UT 84408
lindaduhadway@sbuniv.edu

Abstract

Teaching online is becoming more and more prevalent. When I was assigned to teach two courses online there were many decisions to be made. At the forefront of those decisions was how it would work with my understanding of the effectiveness of student engaged learning. Over several years I have studied much of the research about active and student engaged learning and have implemented many of the techniques in the classroom. Through this study and personal experience, I have seen the effectiveness of providing a curriculum that encourages students to actively engage in the instructional process. The dilemma was how to do this in an online environment. In this article I share the experience of building two online courses while keeping my focus on active, engaged learning strategies. The result provides clear evidence that it is possible. And most valuable of all was the response from students which indicated the approach resonated with them.

1 Introduction

When I was assigned to teach a class online I wondered how others had approached the online teaching environment. One common approach I learned about is to make videos of the lectures from a face-to-face class and publish them for the online students. No other change is required. Everything else can stay the same.

[*]Copyright ©2019 by the Consortium for Computing Sciences in Colleges. Permission to copy without fee all or part of this material is granted provided that the copies are not made or distributed for direct commercial advantage, the CCSC copyright notice and the title of the publication and its date appear, and notice is given that copying is by permission of the Consortium for Computing Sciences in Colleges. To copy otherwise, or to republish, requires a fee and/or specific permission.

The problem with this approach is that I rarely lecture in my face-to-face classes. I have been a proponent of active learning in the classroom for many years. Over the last ten years a variety of active learning strategies have moved into my classroom and almost entirely extinguished the lecture.

Putting a lecture on video does offer some benefits to the traditional lecture. The student can watch the video at a time and place of their choosing. They can fast forward, pause, and rewind as needed. But somehow this does not seem active. Having more control is not the same as engaging with the material. I was not willing to return to lectures. This left me with the dilemma of how to create active learning experiences in an online setting. This paper describes one solution to the dilemma.

2 Active Learning meets Online Education

This is an experiment in course design and is specifically focused on the design and implementation of active learning strategies in an online course.

One of the problems is defining exactly what active learning is [1, 6, 7, 15], Definitions vary dramatically in the literature, ranging from interaction between fellow students and between student and teacher [2] to the responsibility of learning being placed on the student [4]. My perspective of active learning is captured by this question: How is the student engaged in the material during instruction?

Active learning approaches range from polling systems used to inject one or two questions for audience response into a traditional lecture to problem solving activities that span the entire class period. There are many articles that describe a wide variety of activities considered active learning [2, 3, 5, 7, 9, 10, 11, 12, 16, 18].

Spending many years studying, implementing, and observing student engagement in the classroom has provided me with an understanding of what it means to engage in active learning. But all this work has been done in face-to-face classrooms. Now the setting is changing. My assignment is to teach online.

Online courses have become a significant part of student learning and the number of students taking online courses is growing. Enrollments increased for the fourteenth straight year, growing faster than they have for the past several years. As of Fall 2016, there were 6,359,121 students taking at least one distance education course, comprising 31.6% of all higher education enrollments [17]. This increases the demand for instructors teaching online courses.

As an instructor familiar with the advantages of active learning in the classroom, how am I going to use that experience to design an online course? The challenge is to use my understanding and training in active learning to create effective learning experiences for students in online courses.

3 The Approach

The approach implemented is to break each unit down into three instructional stages:

- learn
- practice
- implement

3.1 Learn

For the learning stage I prepare a series of videos built around a learning activity that introduces the material. I work through the activity in the videos showing my work. As I do the work, I provide detailed explanations of what I am doing and how it works. The student is expected to follow along and complete the same activity. This is the active part. They are actually doing the activity as they watch the video. This gets the students actively engaged in the material during the instruction phase. They are learning by doing.

Students are expected to turn in the solution to the learning activity. This verifies that they have actually done the work. Without this requirement, students would likely not complete the activity. The submission is graded for completeness and correctness. The work is shown throughout the video and a document is provided that shows what the final results of the activity are to look like. This provides the information students need to confirm the work they have done meets the requirements.

3.2 Practice

The next stage is practicing. For this stage there is a different activity that covers the same material. For this activity the instructions are written. Each step is outlined and less explanation is given. More explanation is provided when something is covered that was not included in the learning activity.

This stage is also active and introduces repetition and generalization. The students are repeating the concepts learned in the initial learning activity. They are applying those concepts to a novel problem. For this stage the goal is for students to use what they have just learned and apply that knowledge to a new problem. In this phase they are expected to figure out more on their own. The challenge is to balance how much to provide, offering some support but not too much.

Students are expected to turn in a solution to the practice activity. Each solution is graded for completeness and correctness. Again, I provide sample results that students can use to verify they have completed the work correctly.

3.3 Implement

The final step is the implementation phase. In this phase the students are to use what they have learned and practiced to complete an assignment. These are more open ended. They include a list of requirements and provide a rubric of how the requirements will be graded.

Students are expected to turn in a solution. Each solution is graded for completeness and correctness. The final result is not provided. This step is a traditional homework assignment and I approach it the same as homework assignments in my face-to-face classes.

4 The Evidence

Two questions arose from this experience. Each is answered in the affirmative.

The question: Can I use what I know about active learning and create an online course that implements active learning strategies?

The answer: Yes.

The evidence: Two online courses were designed using active learning methods. These courses were delivered Summer and Fall 2018. The expected topics were covered and students produced results similar to their face-to-face counterparts.

The question: Would this approach work for students?

The answer: Yes

The evidence: A high percentage of positive feedback from students.

I do midterm evaluations. The evaluation consists of two simple questions.

- List one or two things that are working well for you in this course.
- List one or two ideas of how things could be improved in this course.

I use the information I gather from these midterm reviews to learn how the class is working for students and what I might do to improve the course. Over the years I have gathered much information that has been used to improve my teaching. What came as a surprise is how many comments are directly related to the specific design of these courses. This is a surprise for a few reasons.

1. There is rarely a single comment that comes up more than just a few times. But in this case there are several.
2. Active learning approaches do not always get good reviews. By definition active learning requires more from the student and they often find this something to complain about.
3. It was unsolicited. Since the evaluation questions do not say anything about the delivery method of the course, specific comments about the learn, practice, implement approach were unsolicited. Students resonated with the strategies enough to mention them without prompting.

The data from two semesters of using this approach are displayed in Table 1. Notice how the numbers do not total the number of student responses. It is possible for a student to include a comment that is positive about the structure of the course and another comment that is negative about it.

Table 1: Student responses on midterm evaluation that include comments about the learn, practice, implement structure of the course

course	number	N2+	N1	neutral	P1	P2+
Web F18	11	0	1	2	6	3
Web S18	17	1	0	2	9	6
DB F18	27	1	2	5	10	9
DB S18	36	4	2	12	7	14
Totals	91	6	5	21	32	32
		7%	5%	23%	35%	35%

N2+: number of negative responses related to two or more stages
N1: number of negative responses related to one of the stages
neutral: no mention any of the learn, practice, implement stages
P1: number of positive responses related to one of the stages
P2+: positive responses related to two or more stages

These responses are from two courses over two semesters: CS 2350 Client Side Web Programming (Web) and CS 3550 Advanced Databases (DB) during Summer 2018 (S18) and Fall 2018 (F18).

70% of the students included a positive comment specifically related to the learn, practice, implement course design. This is significantly higher than expected. There is something about this approach that resonates with many students.

The comments that are about a single part of the course are mostly about the first stage, the learning activity. This is the initial learning part of the unit and is highly active. The student is to follow along with a video that demonstrates and explains new concepts. The student participates in the instruction by completing the activity being demonstrated.

Following are sample comments. These are representative of the comments submitted by students. I have included a proportional number of comments per category as are included in the evaluations.

Positive multiple stages (P2+):

- I really like the setup of having a learn section that introduces me to a new topic, a practice where I really buckle down and learn and apply the concepts and a homework. It helps me to repeat the same topics in different ways through out the week. The videos are also so wonderful. I go back to the videos often as a resource.
- I really like the format of this class. I think the first two assignments do a great job preparing us for the homework
- I enjoy the way the LA, PA, and HW are set up. It helps keep the material fresh and keeps everything in order. I really enjoy learning this way.
- The teaching model. I am really enjoying first learning, practicing, and then testing our knowledge with homework.
- I really enjoy the practice, learn, and then homework. If I don't understand something in practice it's ok because I have time to learn the information before the homework

Positive one stage (P1):

- The follow along activities. I work better when I get to see how something works first beforehand. I can see the steps and exactly what I need to do.
- Having videos showing how the LA assignments are done step-by-step.
- The videos really help me understand how to make the web pages
- I really like the videos and how they break the martial into parts. It helps me focus better on the material when it teaches little parts at a time.
- I love the videos. They are well done and very helpful! I love that I can access them at any time and go back to them as needed. I really appreciate how you do the work in the videos and I can follow along. Too many teachers, even in Computer Science, just lecture and don't show examples. I am a visual learner and I really like the videos you do.

Negative one stage (N1):

- The practice is repetitive. There should just be a learning activity and then the homework. The practice assignment just feels unnecessary and busy work.

Negative multiple stages (N+2):

- Three assignments a week can be overwhelming when you work full time and go to school

5 More Details

The overall design focuses on a learn, practice, implement structure. But there are additional elements that impact the effectiveness of the course.

5.1 Transfer of Knowledge

In each stage the student is actively engaged in the material and in the learning process. The ownership of and responsibility for the knowledge gradually shifts from the instructor to the student. In each unit, the student engages with the topics in three different projects letting them see how the techniques can be used in a variety of ways.

5.2 Discussions

To provide an open forum for interaction between students and the instructor there are ongoing discussions. The course starts with a general discussion that introduces discussions as a way to ask and answer questions, share ideas, and have interaction that applies to current activities. This general discussion remains active throughout the semester and is used for information about general topics.

There is a discussion that goes with each unit. These discussions start with a brief introduction of the unit. In these unit specific discussions, we share information about the current unit. Students use these discussions to ask questions and share information about what they are currently working on. Some students are uncomfortable asking questions in a public forum. To include these students, everyone is also encouraged to ask questions in a more private way by using email. When I receive questions by email that might be of interest to other students, I will add the question and the answer to the discussion. This provides a private way for questions to be asked and still provides the benefit of the answer being available to all the students.

There is no requirement that students participate in the discussion.

5.3 Weighting the Stages

The points associated with the learning and practice activities are significantly less than the homework assignment. For a standard unit the learning activity is 15%, the practice activity is 15%, and the homework is 70% of the points. This lets students learn and practice at a low cost. By the time they are in the implementation stage they have had time to see what works and what they are still struggling with. There is plenty of time to ask questions and get clarification on how to do something.

5.4 Time Expectations

The time to complete the activities is also proportional. The learning and practice activities can be completed in 1-2 hours while the homework takes 4-6 hours. This is accomplished by providing starting resources for the learning and practice activities. The activity is designed to focus on the new topics presented in this unit. Providing the parts that have been learned in previous units reduces the amount of time for completion and focuses on the material to be learned.

In the implementation phase, students are required to complete assignments that include the newly introduced topics as well as topics learned previously. This helps to reinforce what has been previously learned and see how to integrate topics across units. The new topics are still the primary focus of the assignment.

5.5 Timely Feedback

There are advantages to providing timely feedback to each student on the work they have submitted. Providing the expected results for the learning and practice activities is a way to provide general feedback quickly. Then individual feedback is provided as quickly as the assignment can be graded.

5.6 Engaging with the Material Often

Units last a week on average. With three assignments for each unit, this keeps students interacting with the material often. This uses the advantage of distributed practice [8] and spaced repetition. And if students have other courses it also encourages interleaved practice [8].

5.7 Supporting materials

A range of supporting materials are provided for each unit. These supporting materials are recommended. There is no required submission directly related to them.

- Readings. These are suggested readings about the topics in the unit.
- Individual topic videos. These videos cover many topics covered in the learning activity video. They provide a second resource for students that want more instruction. They also cover related topics that are not specifically covered in the learning activity video. This is a way to include additional topics.
- Links to online resources. These are resources related to the topics for that unit.
- Meeting with instructor. If a student wants to meet I set up a Google Hangout that we both can join. This supports audio discussions and video sharing. The option for the student to share their screen informs the discussion and helps me provide detailed feedback.

6 Conclusion

The experience of taking active learning to an online course has been successful. The topics required of the course were all covered. Students responded positively to the experience and many explicitly commented on the value of the learn, practice, implement design.

It is clear that it is possible for someone new to the online environment to find ways to incorporate active learning strategies in online course design. There are many approaches to using active learning in face-to-face classes and many have shared their experiences in education research. There are some examples of using active learning online [13, 14], but there is a need for many more. As more instructors already committed to active learning move to teaching online, there will be many more discoveries about how to incorporate active learning. This will also happen as current online educators see the value of active learning and realize it is possible in an online setting.

References

[1] T. M. Andrews, M. J. Leonard, C. A. Colgrove, and S. T. Kalinowski. Active learning not associated with student learning in a random sample of college biology courses. *CBE—Life Sciences Education*, 10(4):394–405, 2011.

[2] L. Blasco-Arcas, I. Buil, B. Hernández-Ortega, and F. J. Sese. Using clickers in class. the role of interactivity, active collaborative learning and engagement in learning performance. *Computers Education*, 62:102–110, 2013.

[3] P. S. D. Chen, A. D. Lambert, and K. R. Guidry. Engaging online learners: The impact of web-based learning technology on college student engagement. *Computers Education*, 54(4):1222–1232, 2010.

[4] J. R. Drake. A critical analysis of active learning and an alternative pedagogical framework for introductory information systems courses. *Journal of Information Technology Education*, 11:39–52, 2012.

[5] J. R. Drake, M. O'Hara, and E. Seeman. Five principles for mooc design: With a case study. journal of information technology education: Innovations in practice. *Journal of Information Technology Education: Innovations in Practice*, 14(14):125–143, 2015.

[6] V. Drew and L. Mackie. Extending the constructs of active learning: implications for teachers' pedagogy and practice. *Curriculum Journal*, 22(4):451–467, 2011.

[7] L. P. DuHadway. *Course transformation: Content, structure and effectiveness analysis [Thesis]*. The University of Utah, 2016.

[8] J. Dunlosky, K. A. Rawson, E. J. Marsh, M. J. Nathan, and D. T. Willingham. Improving students' learning with effective learning techniques: Promising directions from cognitive and educational psychology. *Psychological Science in the Public Interest*, 14(1):4–58, 2013.

[9] J. Gardner and B. R. Belland. A conceptual framework for organizing active learning experiences in biology instruction. *Journal of Science Education and Technology*, 21(4):465–475, 2012.

[10] H. H. Hu and T. D. Shepherd. Teaching cs 1 with pogil activities and roles. In *Proceedings of the 45th ACM technical symposium on Computer science education*, pages 127–132. ACM, 2014.

[11] S. Kotru, S. L. Burkett, and D. J. Jackson. Active and collaborative learning in an introductory electrical and computer engineering course. *The journal of general education*, 59(4):264–272, 2010.

[12] C. B. Lee, S. Garcia, and L. Porter. Can peer instruction be effective in upper-division computer science courses? *ACM Transactions on Computing Education (TOCE)*, 13(3):12, 2013.

[13] B. Mehlenbacher, C. R. Miller, D. Covington, and J. S. Larsen. Active and interactive learning online: A comparison of web-based and conventional writing classes. *IEEE Transactions on Professional Communication*, 43(2):166–184, 2000.

[14] J. M. Phillips. Strategies for active learning in online continuing education. *The Journal of Continuing Education in Nursing*, 36(2):77–83, 2005.

[15] M. Prince. Does active learning work? a review of the research. *Journal of engineering education*, 93(3):223–231, 2014.

[16] G. Salmon. *E-tivities: The key to active online learning*. Kogan Page, 2013.

[17] J. E. Seaman, I. E. Allen, and J. Seaman. Grade increase: Tracking distance education in the united states, 2018. `http://www.onlinelearningsurvey.com/highered.html`.

[18] Z. Zayapragassarazan and S. Kumar. Active learning methods. *NTTC Bulletin*, 19(1):3–5, 2012.

Evaluating Student Learning Through Problem-Based Learning Using an ERP Simulation Game*

Denise Duncan and Ed Lindoo
Regis University
Denver, CO 80221
{dduncan,elindoo}@regis.edu

Abstract

Companies worldwide use Enterprise Resource Planning (ERP) systems to effectively manage business processes across organizational departments and information among global subsidiaries. Many colleges and universities have incorporated ERP systems into their CIS curricula to provide students with a better understanding of business process theory and information sharing concepts. For years we have used various SAP tools at our university to help students with these theories, and more recently the SAP ERP simulation game (ERPsim) to provide students with a live, competitive interactive experience. However, we have never measured student learning or satisfaction in courses where we use the ERP simulation game.

To address this, using a Problem-based Learning (PBL) approach and the ERP simulation game, we have developed a survey instrument that will help us ascertain if our methodology enables students to (a) enhance their knowledge of enterprise systems and integrated business processes; (b) enhance data analysis and decision-making skills; (c) develop effective problem-solving skills; (d) become intrinsically motivated to learn; (e) enhance team collaboration and communication skills; and (f) prepare for cross-functional, multidisciplinary challenges in their professional career.

*Copyright ©2019 by the Consortium for Computing Sciences in Colleges. Permission to copy without fee all or part of this material is granted provided that the copies are not made or distributed for direct commercial advantage, the CCSC copyright notice and the title of the publication and its date appear, and notice is given that copying is by permission of the Consortium for Computing Sciences in Colleges. To copy otherwise, or to republish, requires a fee and/or specific permission.

Introduction

Students educated for the 21st century must develop habits of critical thinking, researching, and problem solving to succeed in an increasingly technical and complex world. The IT Industry is requiring more from our students than theory, knowledge, and application. Our graduates must be able to think critically on complex, unstructured problems, work effectively in interdisciplinary teams, communicate with stakeholders and peers, and be self-directed learners in order to cope with rapid changes [16]. As educators, we are faced with the dilemma of how to enable deep learning in the subject matter but also provide an environment for learning that enables the above characteristics while keeping the student motivated to learn. Problem-based Learning (PBL) with the use of a live simulation game is an instructional approach that may provide the solution.

Educating undergraduate Computer Information Systems (CIS) students is challenging in today's environment. In addition to technology skills (i.e., software engineering, database, and programming), CIS students should possess a hands-on, working knowledge of integrated business processes, enterprise systems, and data analytics. These students must be able to make business-related decisions in real-life situations where the solution is often not obvious and highly complex [3, 9]. Providing this type of education in a CIS curriculum is a challenge especially when most students have no exposure to integrated business processes either in the business curriculum or through personal work experience. In the undergraduate CIS curriculum at our university, we developed a Foundations of Information Systems and Enterprise Systems courses that apply a PBL learning approach using a live Enterprise Resource Planning (ERP) business simulation game called ERPsim, which was developed by HEC Montréal, a Canadian university. To evaluate our approach, we developed a study to investigate how effective this instructional approach is in the undergraduate CIS curriculum.

The goals of this study are to determine if using PBL and the ERP simulation game enable students to: (a) enhance their knowledge of enterprise systems and integrated business processes, (b) enhance data analysis and decision-making skills, (c) develop effective problem-solving skills, (d) become intrinsically motivated to learn, (e) enhance team collaboration and communication skills, and (f) prepare for cross-functional, multidisciplinary challenges in their professional career. To that end, a student survey is being conducted to assess the goals of the study. In this paper we describe how we implemented the PBL/ERP simulator approach in the classroom, our research on the subjects, and our survey instrument development. Study results will be presented after one year of data collection.

Research

PBL is an active approach to learning in which learners collaborate in understanding and solving complex, ill-structured problems [1, 15]. It is an instructional learner-centered approach that empowers students to think critically, to analyze and solve complex real-world problems, to find, evaluate and use appropriate learning resources, to work cooperatively, to demonstrate effective communication skills and to use content knowledge and intellectual skills to become life-long learners [7]. Critical to the success of the approach is the selection of ill-structured problems (often interdisciplinary) and a tutor who guides the learning process and conducts a thorough debriefing at the conclusion of the learning experience [6, 15].

Since PBL's conception in medical education nearly 50 years ago, it has been used in a variety of settings from middle school, higher education, and professional education. From a literature review, there have been studies on PBL in computer science education mostly in the area of pedagogy. Studies of students' actual engagement with the PBL process remains an open area for research in CS contexts. Likewise, issues pertaining to the overall goals of a CS PBL course, and the degree to which these met teamwork, and student motivation are still areas for research [13]. O'Grady [13] identified four outstanding knowledge areas where the harnessing of PBL is still outstanding; Computational Science, Social and Professional Issues, Information Management, and Graphics and Visual Computing. Finally, the authors found that PBL has not been systematically assessed in CIS curricula, particularly in the use of an ERP simulation game. One of the challenges of teaching PBL is the development of an engaging 'problem' for the PBL scenario. The ERP simulation game is an excellent vehicle to deliver PBL instruction for it is based on a PBL approach.

ERPsim is an ERP teaching-learning software program which uses a live production SAP ERP to teach ERP concepts, integrated business processes, critical thinking, teamwork, data analytics and more. ERPsim simulates a real-world marketplace in which virtual companies can operate using a commercial version of the SAP ERP software [11]. Students must operate the full cash-to-cash business cycle where they must plan, procure, produce, and sell their products. They operate their business using a real-life ERP system which is used by the world's largest companies. In order to be profitable, students must analyze real-time organizational data using reports in the ERP systems and data visualization applications such SAP Cloud Analytics and Tableau. The Open Data (OData) protocol is also supported. The ERPsim game simulates customer behavior for the market, administrative tasks (strategic and tactical decisions are left to the students), and the passage of time. In addition, ERPsim was designed to support a rich and representative manufacturing context typical for a medium-sized organization.

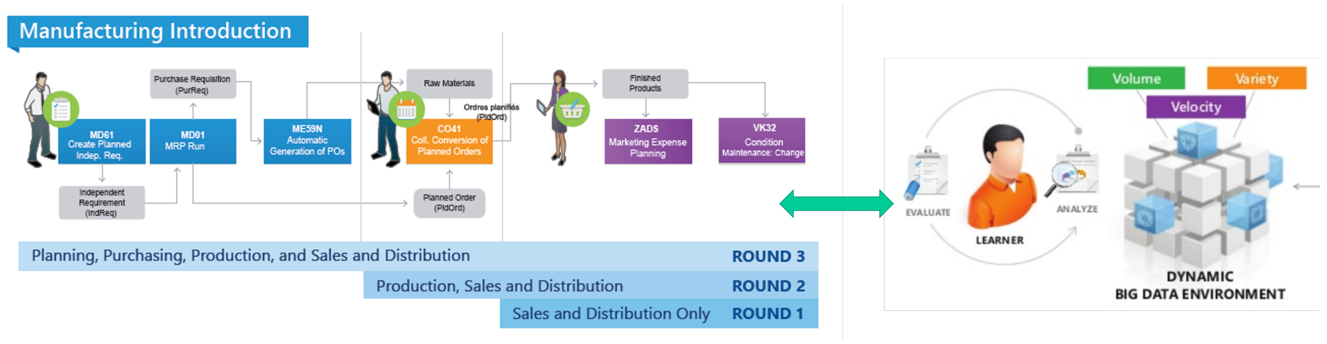
In the ERPsim, a business day is one minute, forcing learners to focus on making the best decisions in the most efficient way possible. As the simulation evolves, students need to find ways to elevate their knowledge of the ERP system and might come up with solutions to improve their decision making (for example, by creating data visualizations to support their decision model). As other teams of students do the same, the game becomes increasingly competitive, expanding knowledge even further [12]. Because pedagogical evidences suggest that ERPsim improves students' learning performance in IS courses [11, 17, 4], we intend to survey students after extensively using this simulator so that we may obtain first-hand data related to student perceptions of the simulation game, team dynamics, motivation, the PBL learning approach, and course outcomes.

ERP Simulation Game Overview

The ERP simulation game (ERPsim) is a business game played by teams of three to five players. Each team runs a make-to-stock cereal company and competes with other teams. Teams have to plan, procure raw materials, schedule production, and market their products. To perform these tasks, participants must be able to use an ERP system, SAP, to support decision making. To be successful in the game (i.e. to maximize profits), participants must not only be able to interact with an SAP ERP, but must also be able to collaborate as a team, understand integrated business processes, and implement the best possible business decisions based on real-time organizational data [5].

The ERP simulation game is an innovative "learn-by-doing" approach where students are given a minimal instruction to start playing the game and learning the SAP ERP system. The main idea is to give enough information so learners can start exploring on their own. Participants are given a scaffold on which they can rely on which eliminates the need for continuous help from the instructor [2]. In the ERP simulator world, scaffolding comes in the form of a well-designed job aid that students an refer to in order to get the job done quickly and simply. Learning takes place as students actually engage in the simulation and find solutions to problems they encounter [5]. Figure 1 below illustrates the integrated business processes in the Manufacturing Introduction game along with the dynamic big data environment.

The simulation game is played in 'Rounds' to simplify the learning curve. First, in Round 1, students learn to market and sell their cereal products with a fixed amount of inventory. In Round 2, we add production to sales and distribution. Finally, planning and purchasing functions are added in Round 3 so that the entire business cycle is experienced. Each Round is normally 20 minutes where a minute represents a business day (20 business days/month). During the game each student takes on a role in the company such a Pricing



Manufacturing Introduction
Purchase Requisition (PurReq)
MD61 Create Planned Indep. Req.
MD01 MRP Run
ME59N Automatic Generation of POs
Independent Requirement (IndReq)
Raw Materials
Ordres planifiés (PldOrd)
CO41 Coll. Conversion of Planned Orders
Planned Order (PldOrd)
Finished Products
ZADS Marketing Expense Planning
VK32 Condition Maintenance: Change
Planning, Purchasing, Production, and Sales and Distribution ROUND 3
Production, Sales and Distribution ROUND 2
Sales and Distribution Only ROUND 1
Volume
Variety
Velocity
EVALUATE
LEARNER
ANALYZE
DYNAMIC BIG DATA ENVIRONMENT

Figure 1: ERPsim Manufacturing Introduction Game Environment [12].

Manager, Marketing Manager, Production Manager, Planning and Purchasing Manager or Data Analyst. Students are encouraged to change roles each time the game is played. At all times during the game, students use standardized and customized ERP reports to analyze real-time company data to make business decisions to ensure profitability of their operations. Students have the choice to use internal ERP reports which are tabular in nature or graphic data visualizations built using SAP Cloud Analytics or Tableau.

To be successful, teams must establish a sustainable cycle of procurement, production and sales that generates enough cash to cover fixed costs. As this cycle continues to repeat, students see the real value in this real-life simulation, needing to understand all aspects of a business from sales, to accounting to procurement, fulfillment, production, warehousing and distribution [12].

During the class, the students experience at least three complete games. There are normally two practice games and then a final game with rewards for winning the team. The winning team gets a 5-point bonus on their ERPsim Game Report. In order to qualitatively assess our students on the course outcomes, we require each student to reflect on their marketing, pricing, production and planning strategies, lessons learned along with their data analysis in a written report due at the end of the course.

ERP Technology as Platform for Enactive Learning of Business Intelligence Concepts

Labonte-LeMoyne cite10 pointed out that in 2012, the Business Intelligence (BI) version of the ERPsim game was developed to allow quasi real-time data analytics. It relied on an extract, transfer and load (ETL) to provide updated data at the end of each virtual day. The extracted data was stored in a Microsoft SQL database. Then by using Microsoft Excel, students are able to extract views from database and refresh these views every minute. This approach was a bit cumbersome, but now SAP has developed HANA, a real-time in-memory database that enables real-time analytics on raw transactional data. Instead of using an ETL approach, where data used for analysis are available after the ETL process is completed, analysis can be performed directly on transactional data. Visualization tools, such as SAP Analytics Cloud can then be used to visualize the data and support the decision-making process during the simulation. Labonte-LeMoyne [10] also mentioned that as of 2015, more than 1,000 instructors have been trained to use ERPsim for teaching in more than 220 universities and colleges worldwide. Arguably, with this growth rate, ERPsim appears to be an excellent teaching tool, but we want to better understand the student experience, and thus this study.

Methodology: Instrument Development

To test our hypotheses, a survey instrument was developed based upon prior experience and research findings in the literature. We focused on two major areas, enjoyment and learning outcomes.

Enjoyment can be measured in many ways. Fu, Su, and Yu [8] developed a scale consisting of eight dimensions: Immersion, social interaction, challenge, goal clarity, feedback, concentration, control, and knowledge improvement. In modeling our questions based on their scale, we will be able to better evaluate student enjoyment. We feel this is important because these authors [8] believe that "the learner's enjoyment acts as a catalyst to encourage his/her learning initiative."

Learning outcomes can may be measured in several ways, however we focused on a grid developed by MIT professor Lori Breslow, et al., in 2007 that provides a way to categorize surveys of student attitudes when asking for reflections on their learning. As Rajkumar et al. [14] pointed out, learning outcomes can be measured with direct and indirect assessment methods. We followed the direct route similar to Rajkumar [14], developing a survey using a 5-point Likert-type scale with 31 questions.

Students will rate using the following: 1 - Strongly disagree, 2 - Disagree, 3 - Neither agree nor disagree, 4 - Agree, 5 - Strongly agree.

The following survey questions will be administered anonymously by using Survey Monkey. Questions and methods have been approved by the Institutional Review Board (IRB).

Demographic

1. Age in years: (18-25) (25-35) (36-45) (46-55) ($>$ 55)
2. Gender: (Male, Female, Other)
3. Course enrolled in (CIS 300, CIS 464)
4. Major: (CIS, Business)
5. Any previous experience working with SAP ERP? (Y, N)

After playing the ERPsim game in the class, rate your experience.

6. I feel that I have learned how to create, execute, and adapt a business strategy in a real-time environment.
7. I was able to learn from my mistakes quickly.
8. I was able to determine changes to our business strategy based on the results from the ERP simulation game.
9. The ERP simulation game gave me exposure to the real-world business problems.
10. The ERP simulation game helped me understand how an ERP can improve the business operations.

11. I feel I have gained a hands-on understanding of the concepts underlying ERP systems.
12. I feel I understand the flow of information from beginning to end in a typical organization using an ERP system.
13. I learned about the integration of business processes as a result of the ERP simulation game.
14. I was able to use real-time financial and operational data from the ERP simulation game to make informed business decisions.
15. The ERP simulation game increased my motivation to learn ERP systems.
16. The ERP simulation game increased my motivation to learn integrated business processes.
17. The ERP simulation game increased my motivation to learn data analysis techniques to make better informed business decisions.
18. The competitive nature of the ERPsim game motivated me to learn.
19. I felt it was an effective way to learn about an ERP system.
20. I felt it was a convenient way to learn about an ERP system.
21. I felt comfortable using it as a learning tool.
22. I found it intuitive to play the ERP simulation game.
23. Learning about ERP systems using the simulation game is more exciting than traditional teaching methods.
24. Team members help one another deal with problems or resolve issues.
25. Team members seek and give each other constructive feedback.
26. Team members are encouraged to express different points of view.
27. Team members willingly take on new responsibilities.
28. Team members follow through on decisions and action items.

Open-Ended Questions

29. What are the best aspects of the ERP simulation game?
30. What are the most challenging aspects of the ERP simulation game?
31. What recommendations do you have for improving the experience and administration of the ERP simulation game?

Conclusion and Recommendations

Just like pilots use real-time simulators, without real-world practice, it can be a challenge for CIS students or even employees to understand integrated business processes and enterprise software usage. This study will examine how a PBL learning approach using an ERP simulator enhances student learning in a CIS course. We anticipate that the findings will reveal the effects of two important IS constructs, 1) enjoyment and 2) learning outcomes during students' involvement with ERPsim. We believe that this study will provide evidence that the

SAP ERPsim game wrapped within a Problem-based Learning approach is an effective way to learn integrated business processes and ERP systems.
Note: To use the ERPsim simulation game, an instructor's institution needs to be a member of the SAP University Alliance. Information about how to become a member can be found at http://scn.sap.com/docs/DOC-7876. Because the games are complex, instructors must become certified through training at the HEC Montreal ERPsim Lab or other venues, ultimately taking a certification exam. Information regarding this can be found at http://erpsim.ca or by emailing erpsim@hec.ca. Finally, licenses for learning materials must be purchased to run the Manufacturing and Logistics Games while the Distribution Game is free.

References

[1] Howard S Barrows. *Problem-based learning applied to medical education*. Southern Illinois University School of Medicine, 2000.

[2] Liqiang Chen, Anthony Keys, and Donald Gaber. How does erpsim influence students' perceived learning outcomes in an information systems course? an empirical study. *Journal of Information Systems Education*, 26(2):135–146, 2015.

[3] Thomas Connolly and Mark Stansfield. Using games-based eLearning technologies in overcoming difficulties in teaching information systems. *Journal of Information Technology Education: Research*, 5(1):459–476, 2006.

[4] Timothy Paul Cronan and David E Douglas. A student ERP simulation game: A longitudinal study. *Journal of Computer Information Systems*, 53(1):3–13, 2012.

[5] Timothy Paul Cronan, Pierre-Majorique Léger, Jacques Robert, Gilbert Babin, and Patrick Charland. Comparing objective measures and perceptions of cognitive learning in an erp simulation game: a research note. *Simulation & Gaming*, 43(4):461–480, 2012.

[6] Peter Dolog, Lone Leth Thomsen, and Bent Thomsen. Assessing problem-based learning in a software engineering curriculum using Bloom's taxonomy and the IEEE software engineering body of knowledge. *ACM Transactions on Computing Education (TOCE)*, 16(3):9, 2016.

[7] Barbara J Duch, Susan E Groh, and Deborah E Allen. Why problem-based learning? a case study of institutional change in undergraduate education. *The power of problem-based learning*, 4, 2001.

[8] Fong-Ling Fu, Rong-Chang Su, and Sheng-Chin Yu. EGameFlow: A scale to measure learners' enjoyment of e-learning games. *Computers & Education*, 52(1):101–112, 2009.

[9] W Hung and SMM Loyens. Global development of problem-based learning: Adoption, adaptation, and advancement. *Interdisciplinary Journal of Problem-based Learning*, 6(1):4–9, 2012.

[10] Elise Labonte-LeMoyne, Pierre-Majorique Leger, Jacques Robert, Gilbert Babin, Patrick Charland, and Jean-François Michon. Business intelligence serious game participatory development: lessons from ERPsim for Big Data. *Business Process Management Journal*, 23(3):493–505, 2017.

[11] Pierre-Majorique Léger. Using a simulation game approach to teach enterprise resource planning concepts. *Journal of Information Systems Education*, 17(4):441, 2006.

[12] Pierre-Majorique Léger. Participants guide, manufacturing game, ERPsim lab. *HEC Montréal*, 2017.

[13] Michael J O'Grady. Practical problem-based learning in computing education. *ACM Transactions on Computing Education (TOCE)*, 12(3):10, 2012.

[14] TM Rajkumar, Paul V Anderson, John Benamati, and Jeffrey W Merhout. Are student self-assessments a valid proxy for direct assessments in efforts to improve information systems courses and programs? an empirical study. *CAIS*, 28:31, 2011.

[15] John R Savery. Overview of problem-based learning: Definitions and distinctions. *Interdisciplinary Journal of Problem-Based Learning*, 1:1, 2006.

[16] Maggi Savin-Baden. Using problem-based learning: New constellations for the 21st century. *The Journal on Excellence in College Teaching*, 25(3&4):197–219, 2014.

[17] Ravi Seethamraju. Enhancing student learning of enterprise integration and business process orientation through an erp business simulation game. *Journal of Information Systems Education*, 22(1):19, 2011.

Engaging CS2 Students via a Semester-Long In-Class Game Project[*]

Mike Wallinga
Department of Computer Science
Northwestern College
Orange City, IA 51041
mwalling@nwciowa.edu

Abstract

Object-oriented design principles and software engineering best practices are two key learning objectives of our Computer Science 2 (CS2) course. Many popular textbooks use short, generic examples that fail to capture student interest in the former and lack the scope and scale needed to successfully demonstrate the latter. This paper discusses how a semester-long game project was utilized to accomplish these objectives and improve student engagement, performance and retention.

1 Introduction

Our introductory programming sequence contains two semester-long courses, simply named Computer Science 1 (CS1) and Computer Science 2 (CS2). Both courses are taught using Java. The CS1 course is taken by a variety of students, including computer science majors, mathematics majors, actuarial science majors, and some business majors. Additionally, the course is open to all students as an option to fulfill the institution's quantitative reasoning general education requirement. Given the variety of student backgrounds present in the course, and the relatively large percentage of students who will not continue in the computer science department, our CS1 course focuses on computational thinking, algorithm design, and basic coding principles such as variables, conditional logic, decision structures, iteration, and arrays.

[*]Copyright ©2019 by the Consortium for Computing Sciences in Colleges. Permission to copy without fee all or part of this material is granted provided that the copies are not made or distributed for direct commercial advantage, the CCSC copyright notice and the title of the publication and its date appear, and notice is given that copying is by permission of the Consortium for Computing Sciences in Colleges. To copy otherwise, or to republish, requires a fee and/or specific permission.

Thus, students completing CS1 and enrolling in CS2 tend to be seriously considering a computer science major or minor, and are competent programmers but lacking in design experience. The CS2 course strives to help students develop object-oriented design skills, engage in thorough planning prior to writing code, and appreciate the value of software engineering best practices. Specifically, our CS2 course introduces inheritance, polymorphism, recursion, file processing, exception handling, and basic data structures. Significant time is spent on documentation, error handling, and testing, as well. Alongside these primary topics, students gain experience with simple client-server communication using sockets and graphical user interfaces using JavaFX.

This shift towards design thinking, intentional planning, and disciplined coding can be difficult for some students, particularly if they developed bad habits in CS1. Helping students move from a basic grasp of class definitions to a more complete understanding of inheritance, polymorphism, abstract classes, and interfaces is a continuous challenge. In reviewing several popular introductory Java textbooks, the large majority of examples used to explain these concepts are generic and fail to capture students' imaginations. They also tend to be brief and self-contained.

In the most recent offering of CS2, a larger, continuously evolving project was implemented throughout the course of the entire semester. At this point in our curriculum, students are considered too inexperienced to undertake a project of this size and scope, but keeping the project contained in the classroom as an ongoing example alleviated this concern while providing a practical, engaging experience. As new topics were introduced to the code base, students could see the effects of a particular design choice on the program as a whole. Seeing the "domino effects" of decisions made earlier in the semester helped students gain an appreciation for the importance of initial design and planning. In this specific offering, the project implemented a computer role-playing game (RPG), which proved to be an excellent match for the concepts presented in the course and helped maintain a high level of student interest, enthusiasm, and engagement with the material. However, nothing about this pedagogy is specific to RPG concepts, and other project choices should work equally well.

Qualitative student feedback regarding this approach was extremely positive, and student performance and retention to the next semester's course was higher than the previous course offering. This paper presents the details of the project, describes how each major course topic was included in the project, discusses student feedback and outcomes, and suggests future considerations.

2 Textbook Approaches and Related Work

Introductory Java textbooks tend to use generic examples for demonstrating inheritance. These examples typically model a two or three level inheritance hierarchy, often including an example of polymorphism or an abstract class.

One common example uses a Person base class, then subclasses that Person based on the domain being modeled. For instance, a model of a college campus would subclass Person into Student and Employee. Often, a third level is added to the hierarchy, such as Undergraduate and Graduate subclasses added underneath Student and Faculty and Staff subclasses added underneath Employee. If a business is modeled instead of a college, the Employee hierarchy might contain Salaried, Hourly, or Volunteer subclasses instead. Variations on this theme can be found in textbooks by Savitch [10], Liang [7], and Lewis and Loftus [6].

iang [7] and Anderson and Franceschi [1] both use a shape-oriented hierarchy, extending the base class (called GeometricObject in the former and Figure in the latter) into Circle and Rectangle subclasses. Gaddis [4] also uses the shape-based example, starting with a two-dimensional rectangle and extending the example into a three-dimensional cube.

Another common hierarchy, seen in both Lewis and Loftus [6] and Anderson and Franceschi [1] concerns vehicles. The example begins with a Vehicle base class, which may or may not be declared abstract. Subclasses might be Car, Boat, Airplane [6], or Automobiles and Trucks [1], with additional levels added to the hierarchy as desired.

Similar textbook examples of inheritance hierarchies include bank accounts [1], retail items such as CDs and DVDs [4], and animals [6].

While these examples illustrate their intended concepts using domains that are accessible and familiar to virtually all students, in personal experience they fail to excite students or encourage engagement with the material. This can impede students' ability to internalize the concepts.

Furthermore, these examples typically appear in a single chapter of their textbooks and are not revisited when new concepts are introduced. In fact, later examples introducing topics such as file I/O, networked communication using sockets, or exception handling may not use object-oriented examples at all. Instead, they fall back on practices that were eschewed in the chapters on object-oriented principles, such as using only static variables and placing all code entirely within a main method. (Examples are readily found in all five textbooks mentioned here.)

The use of such examples is understandable. By doing so, authors are able to demonstrate the current concept in far fewer lines of code than it would take to implement a full class. Likewise, students do not need to familiarize themselves with an inheritance hierarchy and additional boilerplate code just to see the current concept in action. But despite these advantages, it sends mixed messages to students when a textbook spends two or three chapters explaining object-oriented design principles and advocating for their advantages, only to forego them in the next chapter and return to an earlier presentation style. This practice was identified by Malan and Halland as one of four types of code examples that are detrimental to learning programming [8]. They argue that not applying previously taught concepts consistently will undermine students' previous learning of those concepts.

The approach described in this paper employed a semester-long example as a means of remediating both of these concerns. First, the in-class project (in this case, a computer role-playing game) modeled a hobby of many students, increasing their level of engagement by connecting design and coding exercises to a concrete implementation of something that is personal and important to them. Secondly, adding new functionality by continually building on previous versions of the project helped students contextualize each new topic within the scope of a complete software project. At the end of the semester, students were able to see how the various topics fit together to produce a complete, substantive program, rather than only seeing smaller "toy" programs that demonstrated a single concept without the context of a larger whole.

Previous research supports using lengthy projects to teach object-oriented programming in CS2. For example, Barry, Ellsworth, Kurtz, and Wilkes utilized an active classroom approach including four projects, two smaller two-week projects and two five-week "major" projects [2]. The project presented here was done completely in class, not as a graded assignment, and was continually revisited over roughly 12 weeks of a 16-week semester.

Unsurprisingly, it is not uncommon to use computer games to teach introductory courses. In particular, games have been a popular choice for illustrating object-oriented concepts. For example, Purewal and Bennett designed a Java game programming framework to teach polymorphism [9]. Their framework promotes the creation of multiple, shorter, stand-alone assigned projects, and they reported improved student understanding and motivation compared to more traditional polymorphism examples and assignments. The ongoing, semester-long, in-class approach described in this paper enjoyed similar results, but illustrating polymorphism was only one portion of the project, which intentionally extended to every topic covered in the course.

In some cases, specialized game development environments have proven beneficial. Hoganson used GameMaker to teach many introductory programming concepts, including object-orientation and inheritance [5]. The GameMaker environment is unique because it includes both a drag-and-drop user interface and a Java-like GameMaker Language (GML). Hoganson reported that students experienced a natural progression from GML to pure

Java syntax. Such an environment may have been considered for this project had it started in CS1, but given that our CS2 students had a semester of Java programming experience, it was unnecessary.

Edgington and Leutenegger employed the classic text-based role-playing game Rogue [3]. Their institution uses a three-course CS1 sequence based on the quarter system; the role-playing game project occurs during the third course as a group assignment to review and solidify the concepts presented in the first two courses and introduce software engineering principles and design patterns. The approach described in this paper is most similar to their technique, but this project was implemented gradually throughout the semester as an in-class exercise, concurrent with the concepts being introduced in lecture.

3 Mapping to Course Content

As previously mentioned, the project was fully designed and implemented during class. A brief lecture introduced the concept for each unit. This usually included a short, simple code example similar to those given in popular textbooks. This example code was posted in the course's learning management system and made freely available to the students for reference. After taking any questions about the lecture or the smaller example, the concept was applied to our larger project.

While working on the project, students typically divided into small groups of two or three to brainstorm, design, and implement a feature. After sufficient time to work (which often extended into the next day's class period), the small groups reconvened and reported to the class how they implemented the concept. Class discussion considered the pros and cons of each group's decisions, and a final decision was made for the "master" version of our class project.

3.1 Object-Oriented Design

The first topic covered in the class, and the starting point for the class project, was base class design. In a traditional computer RPG, the player assumes the role of a hero. The hero's characteristics, such as strength, intelligence, and dexterity, are represented numerically. In many cases, the hero is assigned an overall level number representing their progression, which may be controlled by experience points (XP). The player's health is also given a numerical value, often labeled HP. If the game contains a magic-based system, the hero may also have a certain number of points (MP) controlling how frequently a spell may be cast.

Standard actions for a hero included movement, attacking an enemy with a weapon, attacking an enemy with magic, and healing (either through the use of consumable items or magic). Heroes may or may not have a "special" technique which is more powerful or useful than a standard action, but whose use is restricted in some way, such as a "cool-down" timer that prevents consecutive uses.

Nearly all students were already familiar with these concepts and immediately started contributing to the design of our Hero base class. In fact, many students discussed more sophisticated mechanics and wanted to design a larger and more complex class. It was a challenge to keep the design simple enough to remain manageable in a classroom environment where the primary goal was to illustrate object-oriented concepts.

3.2 Inheritance

In most role-playing games, players are not limited to a single type of hero. Different character "classes" allow players to customize their hero's strengths, weaknesses, abilities, and tendencies to their liking. A common example is a warrior who is very strong and deals a lot of physical damage to their enemies but lacks magic abilities, while another example is a wizard who is physically weaker (both in terms of HP and ability to attack enemies) but able to use magic to attack and regain HP through healing.

Initially, students attempted to make these different characters as instances of our Hero class, passing different combinations of values to the Hero constructor to create differences in HP, MP, etc. However, not only did they want the numerical values to differ between these characters, but they wanted different implementations for actions, too. For example, both a warrior and a wizard will have an attack method, but the warrior's implementation will deal more damage, while a wizard's will deal less damage and may miss the target entirely more frequently. Students also wanted some actions to be unique to one character; for example, the wizard should be able to use magic to heal, but a warrior could not.

Through this discovery process, students learned to appreciate the usefulness of an inheritance hierarchy. Splitting into groups, each group developed a single subclass (one wrote the warrior, another wrote the wizard, etc.) that extended our Hero base class. Each group had autonomy to decide which behaviors were inherited from the base class without overriding, which behaviors were overridden in the subclass, and which new characteristics and behaviors (if any) were added to the subclass. This process led to some changes in the design of our Hero base class, such as the removal of a heal method that became unique to the wizard subclass.

3.3 Abstract Classes and Interfaces

Once an inheritance structure was in place, the concept of abstract classes and interfaces was introduced. Once again, the concept was introduced by simple examples as presented in a textbook, but the point was driven home by considering the question, "should a player be allowed to create a generic Hero, or must they choose one of the specific character types?" Students' experiences playing similar games led them to conclude that an instance of a generic Hero was not desirable, so the Hero class became abstract.

This was also an appropriate time to introduce abstract methods. We raised the question whether or not we should have a default implementation for the attack method. Students concluded that each subclass should be forced to write their own implementation, so it should be an abstract method.

A related topic that is often presented in textbooks alongside abstract classes is the concept of an interface. This leads into discussions of multiple inheritance, how Java does not provide a mechanism for it, and how interfaces can be used to get around that restriction and simulate multiple inheritance. The difference between abstract classes and interfaces is often difficult for students to articulate. Expanding our role playing game helped in this regard. One student suggested that our game include races of characters, such as humans, trolls, elves, or dwarves. This greatly increased the number of character types, because now a character could be a human warrior or a dwarven warrior, or an elven wizard or a human wizard, etc. For complexity reasons, we did not fully implement this idea in code, but during class discussion students were forced to wrestle with which set of classes should be extended in an inheritance hierarchy and which should be implemented as interfaces. This was a valuable design-focused conversation that helped students see the differences between these concepts and apply them to a real-world example.

3.4 Polymorphism, Collections, and Generics

Up to this point, our game featured a singular hero. However, many computer games involve a group of heroes going on an adventure or quest, often called a party. For strategic reasons, players often choose to diversify their party to include different character types. For example, a player may want at least one warrior to deal heavy damage, but also at least one wizard in order to heal other party members.

This gameplay convention provided a perfect entry point to discuss polymorphism. When building a party consisting of multiple character types, we could take advantage of the fact that all of these subclasses inherit from the base Hero class, and store them in an array of type Hero. Iterating through the array during a turn-based combat sequence allowed each character to attack using its specific method implementation, demonstrating that even though we were using a Hero array, each character was behaving according to its own specific class definition.

Immediately after this, we introduced the Java Collections Framework, specifically ArrayLists, and replaced the Hero array. Through this change, students learned several new syntax features, including generics, iterators, and the for-each construct. Among other advantages (but perhaps most importantly in the eyes of the students), this change allowed our party of adventurers to grow or shrink throughout the course of the game.

3.5 File Processing and Exception Handling

Anyone who has played a computer role playing game can attest that they tend to be expansive, lengthy experiences, requiring dozens of hours of playtime to complete. While our project did not come close to that narrative complexity or length, students inherently understood the need to save the game and resume it at a later time.

We compared and contrasted multiple techniques to implement a game saving and loading feature. First, we used each object's toString() method to generate a textual representation of the game's status and saved the game state into a plain text file. Next, we used binary file processing to save each object's state in a binary format, which students found preferable.

Due to the possibility of files being accidentally deleted or otherwise inaccessible, file processing is an obvious topic to couple with exception handling. This resonated with students who had the unfortunate experience of losing dozens of hours of game progress due to a corrupted or deleted save file.

3.6 Networking

Since many games today include multiplayer functionality over the Internet or local network, students were excited about the prospect of adding that feature to our project. Our CS2 course does not cover multithreading, so we implemented a simple two-machine client-server interaction. The server retained the bulk of our code and handled all logic and progression through the game, while the client contained only enough code to receive messages, enter responses, and send them back to the server.

Some students found this simple implementation unsatisfying; they were hoping to build more robust multiplayer functionality in which multiple players each controlled one character in a party and progressed through the game together. While this would have more accurately mirrored their past experiences, it fell outside the scope of the course.

3.7 GUIs and Event-Driven Programming

Up to this point, our game operated at the console; players entered text commands and received text responses. Towards the end of the semester, our CS2 course includes a unit on graphical user interface (GUI) programming using JavaFX. As usual, the concepts were introduced in lectures and illustrated with small examples, providing exposure to event-driven programming concepts and implementing event listeners for both mouse and keyboard actions.

Equipped with those basic examples, it was natural for students to ask about building an interface for our game. In addition to providing an opportunity for creative interface design work, which some students greatly enjoyed, this exercise also demonstrated the power of a model-view-controller (MVC) design pattern. After mocking up an interface to replace our console-based interactions, it was straightforward to connect the JavaFX event handlers to the methods in our Hero subclasses without altering any code in the subclasses. Students were impressed by this concrete demonstration of something previously discussed in the abstract sense, and gained a deeper appreciation for design patterns and disciplined coding practices as a result.

3.8 Data Structures

Our CS2 course concludes with an introduction to basic data structures, namely stacks, queues, and linked lists. This unit primarily serves as a bridge to their next course, Data Structures and Algorithms. Revisiting our project here acted as a callback and extension to a previous unit. Having already transitioned from the use of an array to an ArrayList earlier in the semester, students were already comfortable with the idea of switching to a different type of collection to store their characters, so we could alter that code quickly and focus on the differences between the collections. For example, students had to articulate how turn-based combat encounters would change if their party was represented using a last-in-first-out (LIFO) structure like a stack versus a first-in-first-out (FIFO) structure like a queue.

4 Student Feedback, Performance, and Retention

Student feedback to the role playing game project was extremely positive. Student comments and course evaluation responses highlighted the project as "genuinely cool" and "made the course material seem more real and applicable." One student commented that the days spent in groups designing the Hero subclasses were "the best days of the semester and helped me engage with the material on another level." Another student posited that using a more engaging example beyond those provided in the textbook "got people involved that otherwise wouldn't have participated in class discussions," an observation shared wholeheartedly by the instructor.

While the relationship is only correlative, student performance also supported the use of the semester-long project. Average scores on the unit test covering object-oriented design rose by seven percentage points over the previous year, and average scores on the cumulative final exam rose five percentage points year-over-year. Regarding a question about polymorphism on the final exam, one student commented, "I would not have been able to answer this question effectively without the role playing game example. It helped this concept make sense to me."

Beyond student understanding and performance, another sign of positive student engagement is continuation in the program. Among the previous year's cohort, only 55% of students in CS2 enrolled in our CS3 (Data Structures and Algorithms) course. In the most recent CS2 course offering, which used the role playing game project, retention to the third semester course was 83%. Once again, this relationship is only correlative, but provides a positive indicator for continued use of a similar semester-long example in future course offerings.

5 Discussion and Conclusion

An advantage to using generic examples, as found in many textbooks, is a high probability that students will be familiar with the example, and thus are free to focus on the programming concepts without needing to first internalize the domain knowledge. Another advantage is that a generic example will be inoffensive to all students. Both of these concerns weighed heavily on the decision to implement this semester-long example. What if students in the course were unfamiliar with role-playing games and did not understand the conventions and mechanics? What if some students found the fantasy setting and emphasis on violent encounters unappealing or offensive? Similarly, one student comment questioned whether the project was "too male." This was seriously considered prior to using the example in the course, too. If the project unintentionally validated stereotypes, reinforced negative perceptions of the department, alienated some students, or otherwise proved to be an impediment to learning, it would nullify any gains realized.

In this specific course, these concerns were alleviated by the instructor personally knowing all of the students from the previous semester's CS1 course. The majority of the students previously expressed an interest in a game-related project, and the instructor spoke with those who had not to ensure such a project would be welcomed. However, it is certain that future sections of CS2 will contain a different mix of students with different preferences, experiences, and feelings. Just because a role-playing game proved to be an effective and engaging example this time, it cannot be assumed that will always be the case.

The strength of this pedagogical approach is found in iteratively layering concepts onto a larger project throughout the course of the semester, using a subject domain that is interesting and engaging to the students. Thus, while a role-playing game proved to be one such subject domain, it should not be the only one. With the role-playing game project serving as a proof of concept, additional projects of varying subject domains must be developed in order to actively engage all students.

Another student critique felt that the project was too linearly guided during class periods and wanted the opportunity to customize the project outside of class and take more ownership of the project. While the rigid in-class guidance was intentional, opportunities for branching or customizing the project outside of class will be explored for future offerings.

A final student recommendation was to add a third layer to the inheritance hierarchy, rather than implementing only a single layer of subclasses below the abstract base class. This student felt adding one more layer would better define the similarities and unique characteristics among the character classes, help illustrate the concepts more completely, and (perhaps most importantly to the student) more closely represent commercially available games. There is a pedagogical balance to maintain between the size and scope of the example and the ability to focus on the concepts at hand. Even so, multiple textbooks contain examples employing a three-level hierarchy, so this suggestion should not be viewed as too complex for inexperienced students.

While both the qualitative and quantitative results from this project are encouraging and student response was very positive, it must be acknowledged that these results are from a single section of a course with annual enrollment in the low teens, and these results are being compared with a single section from the previous year with similar enrollment. Ideally, a more rigorous analysis of the pedagogical approach would be conducted involving multiple concurrent sections, allowing for a test group and control group. Current institutional enrollment does not allow for such a methodology, but future sections will be closely monitored to observe longitudinal comparisons.

In conclusion, after the positive experience and results of our most recent CS2 course offering, the inclusion of a semester-long in-class project is highly recommended as a means of demonstrating object-oriented design and software engineering practices. This project does not need to replace the smaller, more generic examples commonly found in introductory programming textbooks; it worked quite well alongside such examples, serving to reinforce the current concept and contextualize it within the scope of a larger project. While some care must be taken to ensure the project's theme is accessible and acceptable to all students, there is a substantial benefit in terms of student engagement in choosing a theme and domain that is highly specific and relevant to the lives and interests of the students in the course. Student interest and reception was positive via qualitative measures. Quantitatively, exam performance and retention in the department increased over the previous year. Planned future work includes the development of a more diverse set of project choices, the addition of a third layer to the class hierarchy, options for additional features as outside-of-class homework assignments, and more rigorous.

References

[1] Julie Anderson and Herve Franceschi. Java illuminated: an active learning approach. 2016.

[2] E Frank Barry, Christopher C Ellsworth, Barry L Kurtz, and James T Wilkes. Teaching OO methodology in a project-driven CS2 course. In *Companion to the 20th annual ACM SIGPLAN conference on Object-oriented programming, systems, languages, and applications*, pages 338–343. ACM, 2005.

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

[4] Tony Gaddis. *Starting Out with Java: Early Objects*. Pearson, 6th edition, 2018.

[5] Ken Hoganson. Teaching programming concepts with GameMaker. *Journal of Computing Sciences in Colleges*, 26(2):181–188, 2010.

[6] John Lewis and William Loftus. *Java software solutions*. Pearson, 9th edition, 2017.

[7] Y Daniel Liang. *Introduction to Java Programming and Data Structures*. Pearson Education, 2018.

[8] Katherine Malan and Ken Halland. Examples that can do harm in learning programming. In *Companion to the 19th annual ACM SIGPLAN conference on Object-oriented programming systems, languages, and applications*, pages 83–87. ACM, 2004.

[9] Tarsem S Purewal Jr and Chris Bennett. A framework for teaching polymorphism using game programming. *Journal of Computing Sciences in Colleges*, 22(2):154–161, 2006.

[10] Walter Savitch. *Java: An Introduction to Problem Solving and Programming*. Pearson, 2018.

To Heck With Ethics: Thinking About Public Issues With a Framework for CS Students[*]

Michael Glass
Computing and Information Sciences
Valparaiso University
Valparaiso, IN 46383
michael.glass@valpo.edu

Abstract

This paper proposes that the ethics class in the CS curriculum incorporate the Lawrence Lessig model of regulation as an analytical tool for social issues. Lessig's use of the notion of architecture, the rules and boundaries of the sometimes artificial world within which social issues play out, is particularly resonant with computing professionals. The CS curriculum guidelines include only ethical frameworks as the tool for our students to engage with societal issues. The regulation framework shows how the market, law, social norms, and architecture can all be applied toward understanding social issues.

1 Introduction

To fill a gap in the social issues segment of the CS 2013 computer science curriculum guidelines [2] we introduce our students to a framework for analyzing regulatory issues, a way of thinking that is suited for technically educated citizens to engage in the public discussion.

Knowledge Area SA in the CS 2013 Computer Science curriculum guidelines [2, pp. 192–199] is Social Issues and Professional Practice. For professional practices the topics and learning outcomes in the curriculum guidelines address what to teach in service of this goal, viz: professional codes of ethics, responsibility, liability, and life-long skills such as keeping technically up to date. There is also a topic on ethical theories and principles.

Regarding social issues, the expectations of our CS students are less clear. The document says "Students must also be exposed to the larger societal context of computing to develop an understanding of the relevant social, ethical, legal, and professional issues." Instead of clear expectations, the curriculum guidelines then list several social-issue topic areas that students should be acquainted with, such as intellectual property, privacy, and civil liberties. ABET accreditation [1] lists two outcomes related to this topic: "analyze the local and global impact of computing on individuals, organizations, and society" and "understand professional, ethical, legal, security and social issues and responsibilities."

This paper proposes a sharper learning expectation for CS graduates, beyond exposure to a few specified societal issues:

> Students will be able to apply their technical knowledge and tcechnical ways of thinking as they participate in public discussions of social issues.

We can show the students how to bring to the public discussion the technical things they know beyond what the general public knows.

In this paper we illustrate a framework for thinking about societal issues as problems of regulation [5]. This framework is especially suitable for students accustomed to engineering because it introduces the architecture of the world being regulated, and acknowledges that architecture is malleable. We illustrate learning the framework with an exercise that has been successfully applied in the ethics class at Valparaiso University. We also show how students can apply this framework to help understand a specific issue from CS 2013.

In the context of the ethics class this framework can function in conjunction with analyzing social issues in terms of the common good [7]. The common good helps to guide the desirable outcomes of social policy, while the regulatory framework shows the different mechanisms that society utilizes to achieve that good.

A review of three common textbooks that address the ethics class reveals the same lacuna as the curriculum guidelines. In addition to covering specific professional issues, the books provide ethical theories, professional codes, and other intellectual tools for thinking about professionalism. The books contain extensive coverage of specific social issues and impacts of computing on society. But they do not provide much general-purpose intellectual scaffolding for the CS student to engage with social issues in a systematic manner [3, 4, 6].

2 Background

2.1 The Lessig Model of Regulation

“Regulation” refers to regulating behaviors, promoting some behaviors and suppressing others. Regulating behaviors is not restricted to laws or written rules of conduct. In his seminal book *Code and Other Laws of Cyberspace* legal scholar Lawrence Lessig proposes a model of regulation [5, ch. 7] where the four forces acting to regulate behaviors are:

- Social norms
- Markets
- Architecture
- Law

Two key concepts work in conjunction with the above forces:

- Regulability
- Forces acting indirectly

Lessig suggests cigarette smoking as a real-world example of regulating behavior. Some illustrative examples of the four forces are:

- Social norms affect the ability to smoke. For example it may be acceptable to smoke at a picnic, but the norm is to ask permission before smoking in a car.

- The market for cigarettes, which sets price, variety, and availability for purchase, influences smoking.

- The architecture of cigarettes (the physical properties) exerts regulatory persuasion in multiple ways. Nicotine is addictive, encouraging more smoking, while cigarettes cause cancer, which disincentivizes their use. The odor restricts the ability to smoke in secret. Debris from smoking must be disposed of.

- Laws prohibit children from smoking and prohibit smoking in many locations.

Regulability and indirect action are illustrated by laws sanctioning vendors who sell to minors. This is indirect action, it does not directly regulate the minors who buy the cigarettes. However vendors are more regulable: there are mechanisms which can control business activities and hold vendors accountable. The architecture of retail business involves licenses, physical locations, incorporation papers, and financial accounting standards. Retail sale is a regulable point in the system, through which the law and government can apply regulatory force to reduce underage smoking.

2.2 Regulation is Suitable for the CS Ethics Class

The regulation model analyzes social issues according to an engineering approach. Once a desired social outcome is identified, the model shows the different mechanisms that can be manipulated to achieve it. One advantage for the ethics class is that this framework is neutral with respect to the students' existing ideas of governance. It allows all students to discuss on a common ground. For example some students may prefer less government intervention. In the framework of the model they would argue that personal responsibility (social norms) should be relatively more important in the regulation of smoking. Or people may argue that government mandating less addictive cigarettes (an architectural change) would require less personal responsibility to achieve the same reduction in smoking. Different political and ideological constraints result in structurally different ways of regulating the behavior. Ultimately all four types of forces are acting in all scenarios. The framework provides a common model that all people in the discussion can use.

Treating architecture as a malleable factor, and making the role of architecture explicit, is congruent with how software people think and work. For example, as software engineers, they know that well-designed user interfaces afford some user actions, make other actions difficult, and prohibit others. Lessig introduces his thinking through a series of vignettes that are largely situated in cyberspace [5, ch. 2], using the possibilities afforded by online community websites to make his point.

3 Applying Regulatory Analysis in the Ethics Class

3.1 Sample Assignment: Washing Hands

To learn to analyze social issues with Lessig's model of regulation, we present the students with an anodyne relatively uncontroversial behavior: washing hands after going to the bathroom. Some advantages of picking this topic are: a) students are familiar with hand-washing, they don't have to learn new material, b) students often have strongly-held political opinions and ethical notions, but hand-washing does not trigger these opinions, c) students have not thought about it. The task is as follows:

> Washing hands after going to the bathroom is a regulated behavior. People would not necessarily do it in the wild, it became a common practice in civilized societies only within the past 150 years. What regulatory forces are in play that moved us from non-hand-washing to hand-washing behaviors?

The topic starts with a class discussion, no preparation is needed beyond having learned the Lessig model. It continues with a written assignment, and finishes with class discussion.

The most common reaction at the start of the assignment is that washing hands is basic for health and hygiene, furthermore it exists in conjunction with providing facilities that keep human waste out of our water. Students notice that the harm from infection isn't restricted to the individual, as contamination can spread and diseases are contagious. Thus washing hands is established as serving the common good. But merely serving the common good does not explain the regulatory forces that persuade most people to wash their hands most of the time. We didn't transition from not-washing to washing without some persuasion.

Here are some of the regulatory forces at work that my students discover. They often suggest items that I did not think of myself.

- There is a considerable architecture that was created for washing hands. There are water pipes and sewer pipes serving most habitable structures in the civilized world. There are large public works to provide clean water and to treat waste water.
- The law works directly to provide this architecture, and thus indirectly to encourage hand washing and sanitation. The law created the public sanitation facilities, and also requires their incorporation into individual buildings via building codes.
- People are educated into washing hands when they are young. Students sometimes recall the jingles they learned in primary school. Washing hands is definitely a social norm.
- There are signs in restaurant rest rooms telling the staff they are required to wash hands. Some students have experienced this as food-service employees. Why regulate only employees? It is because the law has mechanisms to sanction restaurants for non-compliance with sanitation regulations. Employee hand-washing is more regulable than customer hand-washing.
- Where public works for water and sewer services are available they reduce the marginal costs for providing facilities in buildings. When each building owner needs to make provisions for water and sewer in the absence of any public infrastructure, they are more troublesome and expensive. By paying for infrastructure separately, the law operates indirectly via economic forces.
- Similarly, if individual people were to be responsible for washing hands, and habitable structures did not include the facilities, it could be extremely cumbersome for even the most motivated person to wash one's hands.

A question that arises is: since washing hands is personal behavior and a matter of personal responsibility, would it be better to simply legally require it? If compliance becomes an issue, pass heavy fines for violations. The short answer, the students conclude, is that this would not achieve the goal. The absence of a convenient architecture for hand-washing would discourage the practice. The moment of needing to wash hands is not a highly regulable point in the process, it usually happens in private. Beyond that, there are considerations such as whether the punishment would be proportionate to the crime.

Ultimately students come to see how all four parts of the regulatory model have been marshalled in service of promoting this simple and basic human behavior. There is no theory or ideology which readily explains which regulatory actions are preferable. In sparsely populated areas, for example, the law may specify individually owned septic fields and wells instead of publicly-paid infrastructure. Because governments control building codes and commercial sanitation codes, these are built-in regulable places where the government can require sanitation fixtures and food service employee hand-washing. The regulatory regime that causes high compliance with hand-washing behavior was engineered by combining various forces according to cost and effectiveness.

Another valuable outcome of the hand-washing assignment is that students also come to see that architecture is malleable, and it has a role in regulating behavior.

3.2 A Follow-on Technology Question: Botnets

The hand washing assignment leads to an assignment about keeping computers free of malware:

> A large fraction of Windows desktop computers become infected by malware during their lifetimes. In addition to inconveniencing computer users, infections often join computers to botnets. Should Microsoft be held to account for releasing Windows software with so many vulnerabilities?

Many students start from a standpoint of: it is the user's responsibility to maintain all updates and to not click on infected e-mail attachments. Software inevitably has bugs, and Microsoft has no control over user behavior, so Microsoft should not be held to account.

However the point here is not the legalities of possibly suing Microsoft, the point is to consider the behaviors that increase or decrease computer infections. The analogy to public hygiene is easy to see. Most users with botnet infected computers are unaware of the infection, but their computers are participating in distributed denial-of-service, spam, and phishing attacks. As knowledgeable technologists, students grasp the architecture of the Internet, datagrams, and

IP addressing. It follows that stealthy botnet infections come to be a primary mechanism for deleterious activities that impact everybody. So there is a public good in reducing the level of computer infections. And a policy that requires a high level of compliance by a large fraction of the world's unsophisticated computer users is not likely to produce the desired result. Similarly, relying on individual governments to take action may not address the problem of botnets, which are randomly distributed across the net.

Structurally the most regulable points in the system are then the software providers (such as Microsoft) and the botnet operators. Of these it is easier to hold Microsoft to account. And indeed Microsoft makes software updates universally available and free to the user. It has been steadily reducing the friction to applying updates, and encouraging automatic updating. This is analogous to providing the hand-washing infrastructure. Economically, it once was common for PC vendors to offer to sell anti-virus software in conjunction with new PCs, meaning there was an economic disincentive to install and maintain protection. Microsoft has absorbed the cost by providing Windows Defender, much as governments have absorbed some of the cost of sanitation systems.

Another question evokes the professionalism discussions from elsewhere in the ethics curriculum. Does Microsoft use best practices, or are there software development practices that would reduce the number of software vulnerabilities and thus the public exposure to botnets?

An extension to the botnet question is: as Windows infections have become less of an issue, botnets have been growing by infecting Internet of Things devices such as security cameras and thermostats. Typically IoT devices are not routinely updated, and do not have built-in frequently-updated anti-malware protections. What are the regulatory mechanisms that can be brought to bear on this threat to public cyber-health?

3.3 Application to CS 2013 Privacy and Civil Liberties Topic

Privacy and Civil Liberties is one of the topics in the CS 2013 Social Issues and Professional Practices knowledge area [2, pp. 198–199]. A computer science student's knowledge of architecture, in the context of Lessig's regulatory model, helps to understand the legal evolution and application of Fourth Amendment protections on information privacy. This analysis helps the technically educated citizen to join the public discussion of information privacy as information architecture changes in the future.

The idea is as follows. The Fourth Amendment provided a particular balance between individual information privacy and government ability to access a person's information. As the architecture of possessing information changes, the forces regulating government access to personal information also change.

The regulatory forces thus have been adjusted to roughly maintain the original balance.

As written, the Fourth Amendment regulates government behavior within an information architecture of physical papers stored in a person's residence or transmitted by mail in sealed envelopes, like this:

> The right of the people to be secure in their persons, houses, papers, and effects, against unreasonable searches and seizures, shall not be violated, and no Warrants shall issue, but upon probable cause, supported by Oath or affirmation, and particularly describing the place to be searched, and the persons or things to be seized.

In 1924 in *Olmstead v. United States* [9] the Supreme Court decided that the government listening to phone calls, known as wiretapping, did not require a warrant. This permitted a regime where the government could eavesdrop provided it applied the listening device at the telephone office and not on the telephone user's property. Chief Justice Taft wrote:

> The amendment does not forbid what was done here. There was no searching. There was no seizure. The evidence was secured by the use of the sense of hearing and that only. There was no entry of the houses or offices of the defendants.

Olmstead did not deal with the new information architecture, where accessing information was not the same as physically seizing papers. The Supreme Court remedied this in 1967 with *Katz v. United States* [8], where the Supreme Court overturned *Olmstead* and concluded: listening to phone calls required a warrant, but collecting phone call metadata did not. From a CS perspective, the primary reasoning can be understood in terms of architecture and the regulation model as follows. In the phone-call architecture world, we exchange personal information by phone that in the late 18th century would have been in letters and documents. For information privacy, permitting unrestricted access to the contents of your phone calls is similar to unrestricted access to your papers. The government is prohibited from seeing this information capriciously. Thus the government needs a warrant.

Thinking further about access to information in the two architectures, the address on the outside of a sealed envelope sent through the mail is exposed for the world to see. Phone call metadata (who called who, and when) is similar to postal addresses. In information space the post office and government could see addresses in the paper-architecture world, thus the government should be able to see the same information in the telephone-architecture world.

The next step in our story of the Fourth Amendment is the conversion of voice communication to digital. The architectural shift caused the regulatory forces to work now in the opposite direction: the government was losing its ability to wiretap even when it obtained a warrant. In the analog telephone world of 1967 *Katz v. U.S.* listening to a phone call involved physical wires. CS students who understand the Internet quickly grasp the problems with wiretapping Internet telephony. In the datagram architecture, there is no regulatory point to attach the figurative wiretaps. Furthermore even if you can monitor traffic by IP address, IP address alone won't be sufficient to identify individual phone calls. The monitor has to open up and examine many packets not part of the target phone call in order to wiretap, but examining other people's phone calls is not part of the warrant.

For digital telephony, therefore, Fourth Amendment levels of permission for the government to see private information was best accomplished via an architectural intervention. In 1994 the Communications Assistance for Law Enforcement Act (CALEA) mandated that digital telephony architecture must be amenable to government wiretap requests. Telecommunications companies were already subject to special regulation, the regulatory point was thus in place. CALEA re-balanced the regulatory forces to achieve the same level of Fourth Amendment privacy protection. To achieve this the law acted indirectly through regulation of architecture.

4 Conclusion

The Lessig framework for regulation gives CS professionals a way to apply their knowledge and talents in public policy discussions. It is a framework comfortable to engineering thinking, with forces acting on objects that can be manipulated. The framework is agnostic with regard to pre-conceived ideas of politics and governance. The framework explicitly incorporates architecture in its analysis, where:

- CS professionals can apply their habit of considering the interaction of architectural features and human behavior, as in the discussion of hand-washing.

- CS professionals can apply their knowledge of the computing world, as in the discussion of the evolution of Fourth Amendment protections.

Notice further that the analyses in this paper all assumed a common good that was the goal of the regulatory model: reduced smoking, hand-washing hygiene, reduced public threats from botnets, and a balance between information privacy and government searches. A CS ethics class which helps students to frame social issues in terms of the common good has been described elsewhere [7]. The framework described here begins where that leaves off, building a more extensive intellectual apparatus for analyzing regulation in service of the common good.

CS 2013 and ABET lack specificity in what to expect from CS graduates with respect to social issues curriculum goals. This lack is partially addressed here. In addition to calling out a few specific issues that students will be exposed to in the ethics class, we can provide the students with a general intellectual tool for analyzing many social issues. This tool can serve the students throughout their lives as technologically knowledgeable citizens engaged in the public debate.

References

[1] ABET. *Criteria for Accrediting Computing Programs, 2017-2018.* 2017. `https://www.abet.org/accreditation/accreditation-criteria/criteria-for-accrediting-computing-programs-2017-2018/`.

[2] ACM/IEEE-CS Joint Task Force on Computing Curricula. *Computer Science Curricula 2013.* ACM Press and IEEE Computer Society Press, December 2013.

[3] Sara Baase. *A Gift of Fire.* Pearson, 4th edition, 2013.

[4] Josepth Migga Kizza. *Ethics in Computing.* Springer, 2016.

[5] Lawrence Lessig. *Code: And Other Laws of Cyberspace, Version 2.0.* Basic Books, 2009. `http://codev2.cc`.

[6] George Reynolds. *Ethics in Information Technology.* Cengage, 6th edition, 2019.

[7] Johanna E. Urman and Richard Blumenthal. An Undergraduate Ethics Course for Promoting Common Good Computing: A Progress Report. *J. Comput. Sci. Coll.*, 34(2):39–45, December 2018.

[8] Katz vs United States. 386 U.S. 954 (1967).

[9] Olmstead vs United States. 277 U.S. 438 (1928).

Design and Development of an Open Private Educational Cloud Storage Solution for Application Development*

Kevin Pyatt and Mohamed Lotfy
Regis University
Denver, CO 80221
{kpyatt, mlotfy}@regis.edu

Abstract

CS, IT, and software engineering students need to learn and master web application development and cloud computing skills on an educational full-stack Cloud architecture. In this paper we provide the design of an educational full-stack architecture integrating a private storage cloud. The focus was on how to integrate, configure, secure and deploy web-software applications, microservices and private storage within a full-stack architecture. In this research study we determined the feasibility of designing and developing an open private-cloud storage solution for cloud software. We also share the results of the deployment and testing of phases 0 and 1 of the educational full-stack architecture.

1 Introduction

The ACM/IEEE Computer Science Curricula 2013 (CS2013) added platform-based development (PBD) and recommended adding web application development and applying it over a wide range of ecosystems as part of the required PBD knowledge area. The ACM/IEEE Information Technology 2017 Final Curriculum Report (IT2017) included web and mobile app development in the Integrated Systems Technology (ITE-IST), System Paradigms (ITE-SPA), and Web and Mobile Systems (ITE-WMS) essential IT domains. Web application

*Copyright ©2019 by the Consortium for Computing Sciences in Colleges. Permission to copy without fee all or part of this material is granted provided that the copies are not made or distributed for direct commercial advantage, the CCSC copyright notice and the title of the publication and its date appear, and notice is given that copying is by permission of the Consortium for Computing Sciences in Colleges. To copy otherwise, or to republish, requires a fee and/or specific permission.

development technical skills, including asynchronous web development techniques, are recommend courses in the IT curriculum. To address CS2013 and IT2017 recommendations, web application development courses, both required and elective, were added to computer science and information technology programs to address the growing demand of these skills in the graduates of these programs.

Web applications are client-server software programs where the back-end runs on a web server and the client part (front-end), which includes both the user interface and client-side logic, runs inside a web browser. There are many different frameworks and ecosystems used to create web applications. When these frameworks and components are used together they are referred to as a stack or platform [7]. When it comes to design and development of database-driven web software, the architecture will typically follow an architecture termed full-stack [5]. This is a 3-Tier design, where client and back-end are the two tiers joined by a specialized middle tier called the representational state transfer layer or "REST" [1, 9]. Full-Stack development blends the traditional tiers like data, client and application-server, which requires integration of the all three service architectures [6]. For example, a full-stack architecture will generally be represented by software-as-a-service (SaaS - PHP or Angular), infrastructure-as-a-service (IaaS - Apache II web-server) and platform-as-a-service (PaaS - MySQL, Mongo) layers [5].

Cloud storage solutions are attractive options for layered-architecture design, especially as it relates to storage and micro-service domains [11, 16]. In full-stack environments, the security for data storage (i.e., storage as a service) must be controlled, described, maintained and tested [1]. Most full-stack architectures utilize cloud storage services to a varying degree. Because many storage services are fee-based and commercially available, developers may lose autonomy when they integrate such services into their application design. We are therefore interested in the security and scalability considerations associated with cloud storage services.

Application development from a layered architecture design requires SaaS options. As such, software engineers need to be able to safely and securely integrate storage services into application design and development. This requires knowledge and expertise of layered architecture, including SaaS, and how the security-layer of the architecture is guaranteed with existing or new storage services. It also requires knowledge of client-side design and development as well as server-side integration (i.e., full-stack). Because many storage services are fee-based and commercially available, developers may lose autonomy when they integrate such services into their application design. This may be necessary for some application development as there may be micro-services or APIs that provide functionality that would not be possible without their integration.

However, from a security perspective, application development should include feasibility studies to determine which solutions provide greatest security for data storage and access. As such, open-cloud options should be taken into consideration as possible SaaS solutions.

2 Educational Cloud Computing Design

Regis Private Educational Cloud is a specific research initiative associated with Regis University College of Computer & Information Sciences (CC&IS) educational Cloud which is based on cloud architecture, private storage and microservices. Specifically, the focus is on how to integrate, configure, secure and deploy websoftware applications, micro-services and private storage within a full-stack architecture. In addition, enable CS and IT students to learn and master web application development and cloud computing skills on an educational full-stack Cloud architecture. Through this research we seek to:

1. Design, develop and test an open, private-cloud storage framework which is integrated into a full-stack architecture;
2. Investigate and test security considerations for full-stack architectures, cloud architectures and private storage frameworks (i.e., high-availability, high-security);
3. Design, develop and test cloud software which integrates full-stack architectures with private-cloud frameworks.

Figure 1 shows the Regis Private Educational Cloud project phases.

2.1 Phase 0: Feasibility Study

A feasibility study was first conducted in Jul 2017 to determine which server hardware to use, which operating system to use, which Web software development architecture to use, and which private storage framework to use. Because a focus of this project was to build a web-development stack for cloud software development, we wanted to start with "metal" (i.e., server) and build from there. Furthermore, because we wanted to use what we learned from this process to teach others to do the same, we wanted to use inexpensive "metal". We wanted to use a server operating system (OS) that was mature, secure, open, ubiquitous, inexpensive and well-documented. Specifically, the prototype we designed needed a webserver running an open-stack (i.e., LAMP or MEAN). At the time of this study, a well-documented, ubiquitous, and secure Linux distribution was Ubuntu 16.04 long term support (LTS) OS until 2022. Since the focus of this project is open-stack architectures for software development, we decided to use this version of Linux as our server OS. Regarding the server

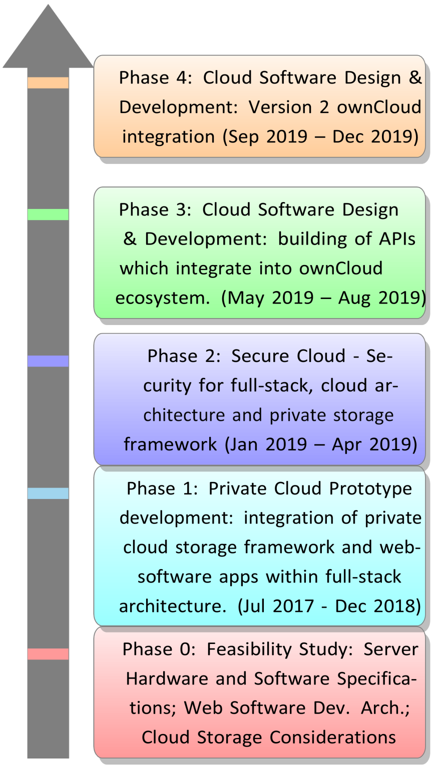

Phase 4: Cloud Software Design & Development: Version 2 ownCloud integration (Sep 2019 – Dec 2019)
Phase 3: Cloud Software Design & Development: building of APIs which integrate into ownCloud ecosystem. (May 2019 – Aug 2019)
Phase 2: Secure Cloud - Security for full-stack, cloud architecture and private storage framework (Jan 2019 – Apr 2019)
Phase 1: Private Cloud Prototype development: integration of private cloud storage framework and web-software apps within full-stack architecture. (Jul 2017 - Dec 2018)
Phase 0: Feasibility Study: Server Hardware and Software Specifications; Web Software Dev. Arch.; Cloud Storage Considerations

Figure 1: Project Phases.

hardware, a decision was also made to use an inexpensive and available HP Pavilion 500-281 Desktop, Intel Core i3-4130 (3.40 GHz), 4GB RAM, 1TB HDD. This hardware had the necessary RAM and processing speed to run Ubuntu 16.04 LTS.

The applications developed for this project were designed and developed using a cloud solution prototype which was database-driven and web-based [5, 6]. We focused on two types of web-software architectures. For applications which used MySQL and PHP, a "LAMP" stack (i.e., Linux, Apache, MySQL, PHP) was used. For applications which would utilize Mongo as database and Node.js as web-server, a "MEAN" stack (i.e., Mongo, Express, Angular, Node) was used. In each of these stacks, there was no reference or specification of where the application and storage layers should reside [5, 6]. This was because, both of these layers existed outside the stack. The storage layer for the application prototype, therefore, needed to be "remote", and on the cloud where it could be accessed from a variety of machines and devices. Creating a private cloud for enterprise-level software development and deployment has become very accessible and reliable with the help of a service called ownCloud [10]. ownCloud is an open storage framework which allows anyone to build or modify and share with user community changes and improvements to the framework. ownCloud was chosen to provide the private cloud storage solution.

2.2 Phase 1: Private Cloud Prototype Design and Implementation

Upon completion of the feasibility study [14], we began designing a prototype for our private-cloud storage framework. The first prototype was built in Jul 2017. The design included server architecture and configuration using Ubuntu-server 16.04 LTS, which was chosen based on recommendations from the feasibility study.

2.2.1 Security Considerations

We took necessary security provisions, based on established best-practices [15], to ensure and protect our server against brute-force attacks. Specifically, we: 1) secured shared memory, 2) enabled ssh login for specific users, 3) added a security login banner, 4) hardened the networking layer, 5) prevented IP spoofing. To a security expert, these security measures may not be complete. However, the scope for Phase 1 was to build a first version of our prototype and to get a reasonably secure server up and running; onto which we could install the private-storage framework and application server. A more comprehensive focus on security and server hardening was the focus of Phase 2 of this project.

2.2.2 Installation and Configuration of Private-Storage Framework

Following this we installed the ownCloud framework for our private-cloud solution. ownCloud 9.0 was installed, and configured.

2.2.3 Development Architecture: Integration of Private-Cloud with Application-Layer

Next, we integrated the private cloud storage framework and web-software apps within full-stack architecture. The server was partitioned to separate the application layer from database. Ideally each of these layers would be running on separate servers. However, this was beyond the scope of Phase 1. We also configured our development server for MEAN and full stack architectures. Again, because this was an application-testing environment, we wanted to be able to run a variety of web-software apps on this server, integrated with private storage.

2.2.4 Software Applications

Throughout the first phase of this study, there were several software prototypes which were designed, developed and integrated into our test server. Several of these applications were developed using MEAN architectures while several more were developed using a LAMP stack. Two examples are shown here.

- Achievement hound [12] is a database-driven web application where faculty and staff can create, read, update, and delete relevant database fields associated with their work and achievements in the areas of scholarship, teaching, service and mission. The first iteration of this application was comprised of a MySQL database, login/registration screen with several customized and responsive report views. The database and application layer for this first iteration were both deployed on the test server. Future iterations will have the database layer deployed on Heroku.

- Course Scheduler [13] is a database-driven web application which facilitates and enables Regis students and academic advisors to plan, project, record, authenticate and revise student schedules. The first iteration of this application was comprised of a MySQL database, login/registration screen with several customized and responsive report views. This database was later transferred to an Heroku Server which was integrated with PostgreSQL.The DaaS service model focused on the data used by mobile applications, by providing database storage technology; e.g., SQL relational and/or NoSQL implementations.

3 Deployment and Testing

The server went live in July 2017. Between July 2017 and December 2019, there were thirty test accounts which were running ownCloud. All of these accounts were running the client on at least one additional computer. Two accounts (alpha and beta) had clients running on five different machines which were synced.

3.1 Storage Size

The default file storage for these accounts was set to 10 GB (10000Mb), although there were several accounts, including alpha and beta, which were running 30 GB storage (30000 Mb). In total, eleven accounts were running 10 GB and four were running 30 GB.

3.2 Functionality Tests

Once clients on each of these machines were installed and configured, we tested for basic functionality issues. This took place shortly after client installation. Specifically, we performed functionality tests on machines associated with alpha and beta clients. We ran a general ownCloud server test, ensured the WebDAV API was working, used a WebDAV command line tool to test, and isolated other issues (i.e., log files and core dumps). It should be noted that, in our research design, functionality tests were conducted on set-up, and then on an as-needed basis if server issues emerged.

3.3 Performance Tests

There are several ways to determine performance of a cloud-storage service [2, 1, 8]. For this study, we were particularly interested in performance issues in terms of ease of use to install and configure client. We measured upload-/download speeds associated with sync service to determine this. We carried out three types of tests to study performance in these areas. We tested file synchronization speeds, upload/download speeds of our server prototype, and database performance.

3.3.1 Synchronization tests

File synchronization, sync time, is the time it takes for a file that exists on client machine to be uploaded to server, and the time it takes for a file that exists on a server to be downloaded to client machine. This is also referred to as initial backup [4]. Synchronization speeds are impacted by many factors [4, 8]. The scope of this study was to get a general measure of file synchronization speeds related to prototype server and client machines. The way we ran these tests was by obtaining and analyzing data from the sync activity logs [3]. IP addresses were not recorded. File/folder sizes ranged from 1 GB to 30 GB (1000 Mb to 30000 Mb).

3.3.2 Speed Tests

For this study, we conducted speed tests to determine the upload and download speeds of our server prototype. This was necessary to get a general idea of server and network performance for our server. Speed tests were defined as "tests which are conducted to determine upload and download speeds for a specific IP address and server at a given moment in time". The tool used for our speed tests was the command-line tool called speedtest-cli.

3.3.3 Database Performance

The database for our ownCloud framework was MySQL. To test database performance, we set our my.cnf file to log three query types: slow, all and those without index [3]. For Phase 3 of this study, we plan to also run test scripts to generate load on server and test auto-loader [3].

4 Results

4.1 Functionality Tests Results

Functionality tests were conducted on the prototype server as well as on machines running ownCloud client. Results from the server tests showed that the WebDAV API was working and configured properly. Based on these findings, the initial server setup was successful.

4.2 Speed-tests Results

Forty-five speed tests (twice monthly) were conducted for the prototype server between Jul 2017 and Dec 2018. This was a large enough sample to determine the general network performance in terms of upload and download speeds for our prototype server. The results are presented in Table 1 below.

Table 1: Descriptive Statistics for Speed Tests

	N	Mean	SD	Min	Max
Upload Speed	45	11.03 s	0.23	8.61 s	14.60 s
Download Speed	45	28.28 s	0.61	20.15 s	32.60 s

4.3 Sync Tests Results

Fifteen tests for file sync speeds were conducted between Jul 2017 and Dec 2019. Data from these tests were captured from sync activity logs [3]. To determine performance for synchronization speed, we plotted time vs. size (Mb), see Figure 2 below. We also looked at initial backup time for a variety of files and sizes where file/folder sizes ranged from 1 GB to 30 GB (1000 Mb to 30000 Mb). There was generally a linear relationship for file synchronization speed. For example, we found a file of size 200 Mb would take approximately 4.5 min to be synchronized. Similarly, we found a file of size 1000 Mb would take approximately 16 min to be synchronized. This equates to approximately 16 min to sync file 1 GB in size. Initial backup times are also reported in Figure

2. We got backup times ranging from less than 1 minute for sizes 500 Mb or less, and over ten hours for backups over 30000 Mb. The relationship between backup size and backup time was relatively sigmoidal, where a performance increase took place between sizes ranging from 1 GB to 1.5 GB.

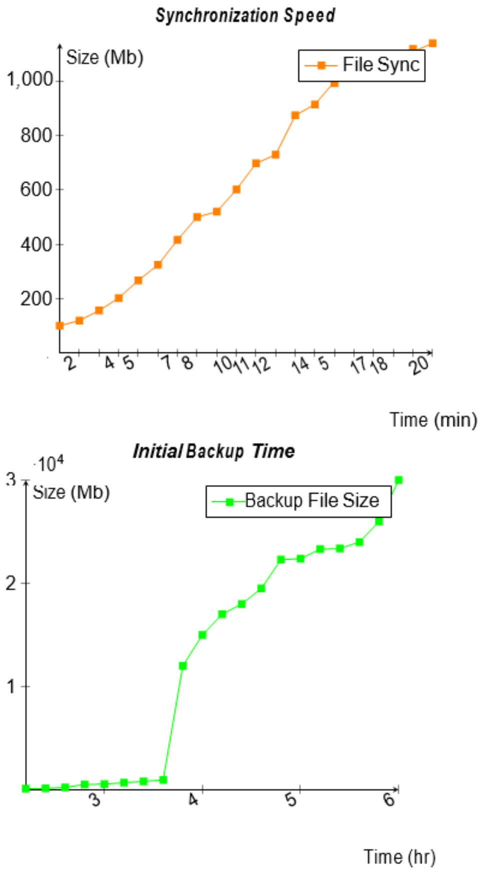


Figure 2: Comparison between File Sync Time and Initial Backup Time.

4.4 Database Performance Results

The database for the ownCloud framework was MySQL. For Phase 1 of this study our tests for database performance were limited to analysis of log files. We set our my.cnf file to log three query types: slow, all and those without index [3]. For Phase 2 we plan to measure database performance in terms of security best practices. In Phase 3 of this study, we plan to also run test scripts to generate load on server and test auto-loader [3].

5 Discussion and Conclusions

CS, IT, and software engineering students need to learn and master web application development and cloud computing skills on an educational full-stack Cloud architecture. The Regis Private Cloud Project is a research initiative associated with Regis University College of Computer & Information Sciences (CC&IS) educational cloud. Specifically, this research focused on how to integrate, configure, secure and deploy web-software applications, microservices and private storage within a full-stack architecture. In this research study we determined the feasibility of designing and developing an open private-cloud storage solution for cloud software. We then designed, developed and tested private-cloud storage and cloud software. These were then integrated into a full-stack architecture. We then deployed software using the LAMP and MEAN stacks which were on test server. In phase 2 we tested and improved the security layer of our server. In the next phases we plan to expand cloud software design through building of APIs which integrate into ownCloud.

We plan to report on the remaining project phases in future publications. In future studies we might employ a sampling or auditing strategy to understand further the performance of the private storage cloud. Future backup tests we may consider running multiple tests from a single IP address, or testing single accounts across multiple IP addresses to better understand upload/download anomalies. The mean upload times reported originated from forty-five speed tests. Data from these tests was reported in terms of file size over time. A more refined test would look at defined file sizes (i.e., 1 Mb) per unit of time.

References

[1] Bashir Alam, MN Doja, Mansaf Alam, and Shweta Mongia. 5-layered architecture of cloud database management system. *AASRI Procedia*, 5:194–199, 2013.

[2] Paul Clements, Rick Kazman, Mark Klein, et al. *Evaluating software architectures*. Tsinghua University Press Beijing, 2003.

[3] Holger Dyroff. ownCloud server manual, January 2019. `https://doc.owncloud.org/server/10.1/user_manual/`.

[4] Joseph Gildred. How long should my backup take?, 2018. https://www.cloudwards.net/how-long-should-my-backup-take/.

[5] Mohamed Lotfy and Kevin Pyatt. Introducing the MEAN stack: a web application development second course. *Journal of Computing Sciences in Colleges*, 34(2):30–38, 2018.

[6] Mohamed Lotfy and Kevin Pyatt. The MEAN stack web application development platform: tutorial presentation. *Journal of Computing Sciences in Colleges*, 34(2):99–101, 2018.

[7] Mohamed Lotfy and Kevin Pyatt. A two-course web application development sequence covering the LAMP and MEAN stacks. *Journal of Computing Sciences in Colleges*, 34(2):22–29, 2018.

[8] Jakub T Mościcki and Massimo Lamanna. Prototyping a file sharing and synchronization service with ownCloud. In *Journal of Physics: Conference Series*, volume 513, page 042034. IOP Publishing, 2014.

[9] NITROSPHERE. 2-Tier vs. 3-Tier application architecture? could the winner be 2-Tier?, 2019. https://nitrosphere.com/uncategorized/2-tier-vs-3-tier-application-architecture-could-the-winner-be-2-tier-2/.

[10] ownCloud. Cloud collaboration platform, 2019. https://owncloud.org/.

[11] Aditya Patawari. *Getting started with ownCloud.* Packt Publishing Ltd, 2013.

[12] Kevin Pyatt. Regis University's achievement hound, 2019. https://github.com/RegisUniversity/achievementhound.

[13] Kevin Pyatt. Regis University's scheduler application, 2019. https://github.com/RegisUniversity/SchedulerApp.

[14] Kevin Pyatt and Ishmael Thomas. Prototype of a private storage-as-a-service solution (i.e., ownCloud) for application development: security, performance and scalability considerations. Technical report, Regis University, 03 2018.

[15] Uday R Sawant. *Ubuntu Server Cookbook.* Packt Publishing Ltd, 2016.

[16] Wazi Wazi. Host your own document storage service with ownCloud, February 2013. https://www.networkworld.com/article/2224072/host-your-own-document-storage-service-with-owncloud.html.

Teaching C# Using Xamarin and Android Tablets[*]

Daniel McDonald, Kodey Crandall, Kimberly Bartholomew
Information Systems and Technology
Utah Valley University
Orem, UT 84058
{dmcdonald, kcrandall, barthoki}@uvu.edu

Abstract

Students taking their first programming courses are more familiar with mobile applications than they are with desktop applications. In our introduction to programming courses, we have prioritized creating assignments that are similar in appearance to the types of applications students use. We recently evolved the introduction to programming courses by switching from Windows forms to Xamarin.Forms. In this paper, we report students' perceptions of using Android tablets to test two different Xamarin.Forms assignments. Survey data showed that students most perceived the use of the tablets to be an engaging and hands-on exercise. Next, students agreed with statements that the tablets facilitated general learning of computing and specific learning of Xamarin. Next, students agreed that using the tablets increased student interaction in the classroom. Finally, the lowest agreement was attributed to statements that the tablets motivated them to want to learn more about Android and mobile development.

1 Introduction

With the release of the second iPhone, called the iPhone 3G on July 11, 2008, Apple's App Store was also unveiled[3]. With an App Store, users could purchase applications to download and run on their phones. On August 28, 2008,

[*]Copyright ©2019 by the Consortium for Computing Sciences in Colleges. Permission to copy without fee all or part of this material is granted provided that the copies are not made or distributed for direct commercial advantage, the CCSC copyright notice and the title of the publication and its date appear, and notice is given that copying is by permission of the Consortium for Computing Sciences in Colleges. To copy otherwise, or to republish, requires a fee and/or specific permission.

Google announced the Android Market for phone applications[8]. In October 2008, the first Android phone, the HTC Dream was released[1]. More and more would-be programmers began running applications on their phones. The use of phone applications has only grown. In 2011, 6.1 percent of global web pages were served to mobile phones. More recently, in 2018, 52.2 percent of global web pages were served to mobile phones[2].

Students learning to program as part of the Information Systems curriculum are more familiar with mobile applications than they are with desktop applications. In an effort to attract students to programming, we prioritize creating assignments that students can relate to and recognize. In keeping with this value, we have evolved our programming assignments in our earliest programming courses to include mobile development. In conjunction with mobile development, we have introduced Android tablets in the classroom, even requiring students to purchase a low-cost tablet in their first programming course. In this paper, we describe the student's perception of using the Android tablets to test the applications they developed using Xamarin.Forms technology.

2 Background

Back in the fall of 2011, we adopted a C# textbook in our two required introduction to programming courses as part of the Information Systems curriculum. Prior to 2011, we taught C# using a command-line interface. This new textbook taught programming from the first assignment using Windows forms applications. We used this new textbook, with all of its new editions, up through spring semester of 2018.

As technologies have changed over the years, we wanted to evolve our introduction to programming courses to stay consistent with our established value of creating applications similar in appearance to the ones used by our students. We decided to first target the second introduction to programming course for the change. Our assignments, instead of using Windows forms, would use interfaces based on two different interface technologies. First, we introduced assignments using the Windows Presentation Foundation (WPF), which uses XAML (Extensible Application Markup Language) as the interface language. Second, we introduced the cross-platform mobile toolkit Xamarin, which also uses XAML in its Xamarin.Forms technology and is based in C#. Once students had been introduced to XAML using WPF, students would create the two Xamarin.Forms applications and then deploy them in class to Android Tablets. Xamarin.Forms technology is a cross-platform library that allows developers to code in C# and deploy to IOS, Android, or Universal Windows Platform (UWP) devices. We had purchased 23 Pixel C tablets and 23 Microsoft Surface

Pros as part of a Perkins Grant to use in our experiment. However, we ended up focusing entirely on the Android tablets.

3 Project Goals

We evaluated the new Xamarin.Forms assignments and student experience on five major research questions that we have used in prior research[5, 6].

First, we wanted to add assignments to the course that were "engaged". Utah Valley University (UVU) received the Community Engagement designation in 2008 by the Carnegie Foundation for the Advancement of Teaching. Consistent with this designation, UVU encourages classes to be hands-on and experiential when possible. Our goal was to "engage" students by being able to handle the tablet, attach it to the computer, put it in development mode and make sure it was visible to Visual Studio and then deploy their applications to the device.

Second, we wanted to motivate and inspire students to want to learn more about Android. According to the Expectancy-Value Model of motivational theory, one component that impacts motivation is students' perception of the importance or value of a task[7]. Fewer than half of the students had worked with an Android tablet prior to the project. We hoped the novelty of the device and the cutting-edge nature of the computing platform would motivate them to want to learn more. In addition, we thought working with the Android tablet was fun. Research has shown that students are more willing to start a project they see as fun and where success can be achieved[4].

Third, we wanted the students to learn how to program in C#. After all, we introduced the tablets in our introduction to programming courses. Our hope was that imagining the possibilities of using a tablet in real projects would motivate the students to put more effort into understanding fundamental programming concepts, such as variables, keywords, and control-flow.

The fourth goal for the project was to teach the students something about computing platforms in general. We wanted to see if all the steps outside of working with Visual Studio, such as tethering the tablet to the computer using a USB cable, booting the Android operating system, and altering the device settings would teach them something new about computing.

Finally, we wanted students to help one another and interact while troubleshooting the device as well as the programming projects. With 27 students trying to boot their tablets and learn Xamarin, we hoped that there would be more student interaction than the typical teacher-class interaction.

4 Methodology

By the time we introduced the tablets in the classroom, we had completed two assignments using WPF and XAML. We spent a total of three and a half weeks working with Xamarin and the tablets. We first brought the tablets to class March 26, 2018, and the last day we used them was on April 16, 2018.

The class was held in a computer lab. We handed out the Android tablets with USB cables before each class. Being short on tablets, some students used their own Android phones, while others used the Microsoft Surface Pro tablets.

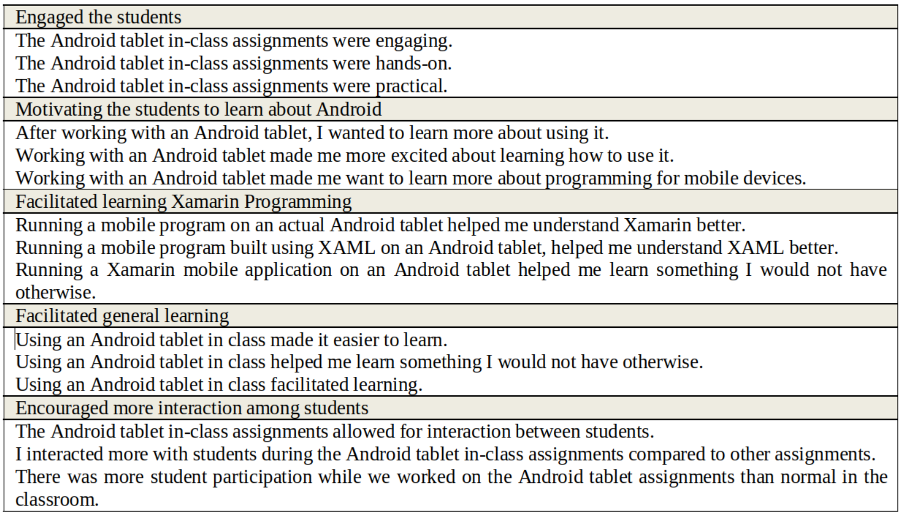
The first Xamarin.Forms assignment required students to develop an interface to send requests to Amazon Web Services and then display the returned XML results as HTML in a web view. Students implemented an ISBN lookup feature where they entered an ISBN number and Amazon returned all the product information about the matching book. Students also implemented an Amazon book search feature that allowed them to submit queries to Amazon's index of books and then get the results of the search back in XML format. Students then processed the XML using LINQ and converted the information into HTML for the web view. The interface for the application was a single form coded in XAML. The form used a grid layout. The grid layout contained several labels, an entry, several buttons, and a web view control to display the returned results from Amazon.

The second Xamarin.Forms assignment required students to create a calculator application. In this assignment, much more work went into creating the user interface and creating event methods for all the calculator operators. Again students used a grid layout class to structure the form into six rows and four columns. A single label was added across the top of the application for users to see the results of calculations. Under the calculator display label, sixteen buttons were added to the interface. Ten of the buttons represented single digit numbers. Four of the buttons were used for division, multiplication, subtraction and addition operators. The remaining two buttons were for the equals functionality and the clear the screen functionality.

After the seven class periods of using the Android tablets and completing the two assignments, students filled out a fifteen-question five-point Likert-scale survey. All fifteen questions are shown grouped by project goal in Table 1.

There were three questions for each project goal. The survey was administered online through our learning management system. Each question required the student to choose their degree of agreement. The scale included the following phrases: strongly disagree (1), disagree (2), neither agree nor disagree (3), agree (4), and strongly agree (5). The questions were mixed together for the actual survey. Students earned five extra credit points for completing the survey. Twenty-seven students took the survey. We excluded data from three

Table 1: Student Survey Questions



Engaged the students
The Android tablet in-class assignments were engaging. The Android tablet in-class assignments were hands-on. The Android tablet in-class assignments were practical.
Motivating the students to learn about Android
After working with an Android tablet, I wanted to learn more about using it. Working with an Android tablet made me more excited about learning how to use it. Working with an Android tablet made me want to learn more about programming for mobile devices.
Facilitated learning Xamarin Programming
Running a mobile program on an actual Android tablet helped me understand Xamarin better. Running a mobile program built using XAML on an Android tablet, helped me understand XAML better. Running a Xamarin mobile application on an Android tablet helped me learn something I would not have otherwise.
Facilitated general learning
Using an Android tablet in class made it easier to learn. Using an Android tablet in class helped me learn something I would not have otherwise. Using an Android tablet in class facilitated learning.
Encouraged more interaction among students
The Android tablet in-class assignments allowed for interaction between students. I interacted more with students during the Android tablet in-class assignments compared to other assignments. There was more student participation while we worked on the Android tablet assignments than normal in the classroom.

students, once where the student answered all ones and twice where the student answered all fives. Our results, therefore, represent the summarized opinions of 24 students.

5 Results

Table 2 shows the results of the survey ordered by the mean of the 5-point scale for each of the five goals. An average of three on the survey represents "neither agree nor disagree". An average of four represents "agree", while an average of five represents "strongly agree".

First, students felt most strongly that using Android tablets in class was an engaging activity with a mean of 4.25. Using the tablets for seven class periods was viewed as useful, hands-on, and practical. The question asking whether the Android assignments were hands-on received the highest agreement of any individual question with a mean of 4.45. Next, students felt using the Android tablets facilitated learning in general with a mean score of 4.19. Students had to put the tablet in development mode and properly tether it to the PCs in the lab. They perceived that using the tablets helped them learn something they would not have otherwise. Third, students perceived that using the Android tablets in class helped them better understand Xamarin.Forms with a mean score of 4.04. Students perceived that deploying apps to the Android tablet helped them learn something about Xamarin and XAML that they would not have otherwise. Fourth, students mostly agreed that using the Android tablets in class encouraged more interaction among students with a mean of 3.97. Students perceived they interacted more with fellow students when using the tablets than they normally did, and that student participation overall was higher than with normal assignments. Finally, students generally agreed that working with the Android tablets motivated them to want to learn more about Android tablets and mobile development with a mean score of 3.92. Generally, students were more excited to learn about Android tablets and mobile development after using them in class than before. However, this goal had the least agreement among students on average and included the question with the lowest mean score. Students had a mean score of 3.79 when asked whether working with an Android tablet made them want to learn more about programming for mobile devices. This goal of motivating the students to want to learn more about mobile development also had the highest standard deviation of all the goals with a value of 1.14.

Table 2: The results of the student survey

Project Goals	Android Tablet	Std Dev	Rank
Engaged the students	4.25	1.06	1
Facilitated general learning	4.19	.91	2
Facilitated learning about Xamarin	4.04	1.01	3
Encouraged more interaction among students	3.97	.97	4
Motivated the students to learn Android	3.92	1.14	5

6 Discussion and Conclusion

Similar to prior research we have conducted, students perceive the introduction of new hardware they use in class to increase the level of their engagement [5; 6]. Being in a computer lab, students can be easily distracted by surfing the web or checking email. Having physical devices that require setup and use help students to stay focused on course content.

Another positive outcome is that students perceived the use of Android tablets in class to facilitate learning, both in general and learning about Xamarin in particular. This outcome appears to be influenced by how related the device is to the assignments being completed. In our case, Xamarin apps are made to be deployed on Android tablets. The tablet helps make testing our applications possible. In prior research, we introduced Raspberry Pi computers in class and used the credit-card sized computers to teach Python. Students did not perceive the use of the Raspberry Pi to facilitate learning about Python. The mean agreement for that goal was 3.81; an agreement score lower than any of the current averages. It is also interesting that students perceived that using the Android tablets in class facilitated general learning more than it facilitated learning about Xamarin. Perhaps because we tested our applications on the tablets as opposed to developing on the tablets, students felt they learned more about general computing principles.

Most students were not familiar with Android, but a few had Android phones and even used those in place of the tablets. Those students, in particular, were helpful in showing other students how to configure their Android tablets for development. This difference in exposure to Android helped promote more interaction among the students.

The lowest performing project goal was the motivating of students to want to learn Android and mobile development after using the tablets in class. While students still mostly agreed with the statement, bringing in hardware to class was more motivating to students in prior research. For example, bringing Raspberry Pi computers to class motivated students to want to learn Python with an average score of 4.17 and motivated students to want to learn the LAMP stack with an average of 4.10. The motivation goal was the second highest average agreement in our previous research. It is possible that Information Systems and Information Technology students are just less interested in mobile development than Raspberry Pi computers. In addition, it could be that students prefer Apple products, such as the iPhone and were less motivated because the tablet was not an iPad.

Overall, students agreed that using the tablets both facilitated general learning and specific learning about mobile application development. This perceived benefit to learning is enough to justify continuing to use the tablets in class.

References

[1] B. Cha. All T-Mobile retail stores to carry G1, 2009. `https://www.cnet.com/news/all-t-mobile-retail-stores-to-carry-g1/`.

[2] J. Clement. Percentage of all global web pages served to mobile phones from 2009 to 2018, 2019. `https://www.statista.com/statistics/241462/global-mobile-phone-website-traffic-share/`.

[3] B. Dolan. Timeline of Apple iPhone rumors (1999-present), 2008. `https://www.fiercewireless.com/wireless/timeline-apple-iphone-rumors-1999-present`.

[4] T. Heafnerm. Using technology to motivate students to learn social studies. *Contemporary Issues in Technology and Teacher Education*, 4(1):45–53, 2004.

[5] D. McDonald. Comparing python and wordpress assignments on a raspberry Pi. *Comput. Sci. Coll.*, 34(2):5–11, 2018.

[6] D. McDonald and S.J. Cold. Using raspberry Pi computers to teach LAMP and wordpress. *J. Comput. Sci. Coll.*, 33(2):148–154, 2017.

[7] D.H. Schunk, J.R. Meece, and P.R. Pintrich. *Motivation in Education: Theory, Research, and Applications.* Pearson, 2013.

[8] S. Shankland. Google announces Android Market for phone apps, 2008. `https://www.cnet.com/news/google-announces-android-market-for-phone-apps/`.