Teaching Statement

My primary teaching responsibility at the University of Hawaii for almost 30 years has been software engineering, and frankly, it has been both rewarding and exasperating. It has been rewarding to enroll students at the beginning of the semester whose software development experience to date consisted of two week homework assignments, and provide them with a feeling for what it means to be a professional software engineer, and that this goal is within their grasp. It has been exasperating because I've never been able to, once and for all, "figure out" how to teach software engineering. Instead, every few years, I've had a new idea about how to teach it, or learned about a new and better component for the technology stack, requiring me to redesign some to all of the course in response. For example, in Spring 2018 I will need to redesign about a third of the course in order to incorporate React, the most recent "final solution" for front-end frameworks.

The following sections present a snapshot of my current educational initiatives: public course evaluations, athletic software engineering, the Morea Framework, Courses.ICS, TechFolios, RadGrad, and community initiatives (HACC, March for Science).

Public Course Evaluations

Starting in 2007, I began making my unedited eCafe Course Evaluations public, and posting a link to these evaluations on the home page for my courses. I tell my students on the first day of class that they should look at these evaluations to get a better sense for what students think about my course and my approach to teaching it.

I also tell them that at the end of the semester, they will have a chance to submit their own evaluations, and that I will be making their comments public. I also tell them that if I get a 100% response rate, then everyone in the class will receive extra credit points, but if even one person does not respond, then I can't award any extra credit to anyone, since the responses are anonymous and I don't know who didn't do the evaluation.

I am very pleased with this initiative in transparency. First, it leads to very high response rates. Most of the time, 100% of the class responds, and they all receive extra credit. I don't think I've ever had a response rate below 90%.

Second, the quality, as well as the quantity of responses is high. Because students know that I will be making their comments public, and because they know that I tell students about the eCafe site on the first day of class, they provide feedback to future students as much as they provide feedback to me. Not all of the feedback is positive, to be sure, and sometimes I might disagree, but I always feel that the evaluations are sincere, civil, and well meaning.

Please take a look for yourself: Philip Johnson's published eCafe surveys.

Athletic Software Engineering

For the past four years, I have been developing a pedagogy called Athletic Software Engineering (ASE) which I use in ICS 314: Introduction to Software Engineering. This pedagogy involves a high intensity, time-constrained, and often stress-inducing approach to acquiring competency with software engineering skills. Despite those characteristics, a survey of ICS 314 students from AY 2016-2017 found that 90% of them prefer athletic software engineering to a more traditional teaching approach.

To motivate athletic software engineering, its necessary to first understand the goals of my software engineering course:

- Acquire competency with a minimal technology stack for modern software application development. In the case of ICS 314, this minimal stack includes Javascript, git, GitHub, ESLint, IntelliJ, HTML, CSS, Semantic UI, markdown, mocha, chai, npm, Meteor, MongoDB, and Galaxy.

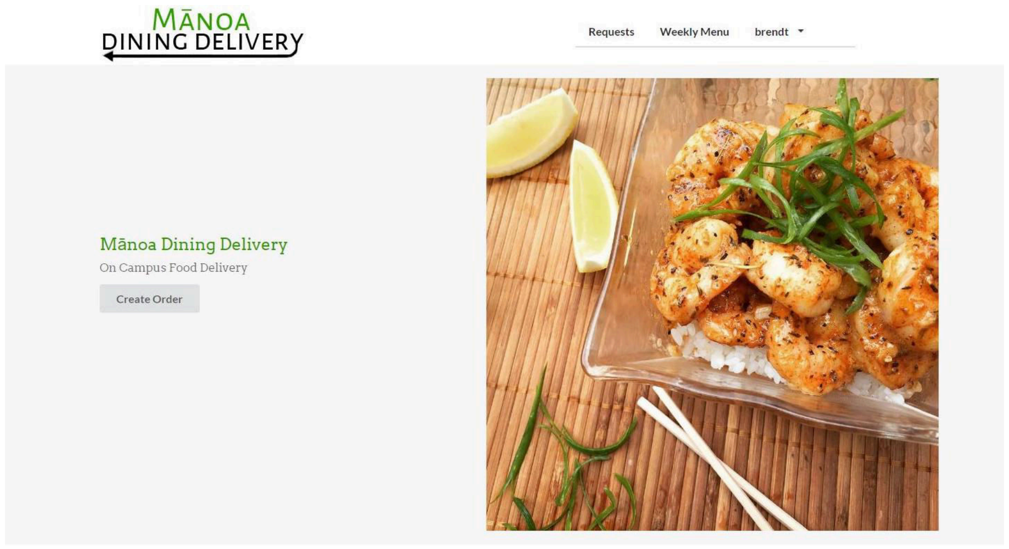


Figure 1: *Student Project: Manoa Dining Delivery*

- Experience "design" problems. Design problems occur at the code level (for example, how to structure code so that it is testable), as well as at the user interface level (for example, how to organize the presentation and manipulation of data so that user needs are satisfied).
- Experience "process" problems. Process problems involve group communication and coordination, and how to define, delegate, and manage tasks such that a group of developers can work together to create a functional application in a limited amount of time.
- Design and implement an application that solves a "real world" problem for UH students. For example, students on a meal plan are allocated a certain number of points each week for buying food, which disappear if unused. To solve this inefficiency, one team developed an app called Manoa Dining Delivery, in which students can turn their expiring points into cash by purchasing food at the cafeteria and delivering it to another student on campus. Figure 1 shows their home page.

These four goals for ICS 314 build on each other: without competency with a technology stack, it is difficult to experience significant coding and user interface design problems. Significant design problems are needed in order to provide a context for teams to experience process problems. And without the ability to solve both design and process problems, it is difficult to successfully design and implement a "real world" application.

Note that developing an application to solve a "real world problem" requires a substantial portion of the semester. I find students need a minimum of six weeks to work on the final project, which leaves only 10 weeks to develop competency with the underlying tools and techniques. Athletic Software Engineering is designed to solve the pedagogical problem of developing competency with a significant set of tools and technologies in the time available during a semester.

Most students report that the WOD-based athletic software engineering pedagogy is very stressful. To obtain a detailed perspective, I ask students each semester to complete a questionnaire about their view of WODs and Athletic Software Engineering. Here are the results for AY 2016-2017.

There were a total of 119 students across the four sections of ICS 314, and 86 filled out the survey, for a 72% response rate. Here are two responses from the survey:

Do you prefer ASE to the traditional way of teaching?

90% of the students surveyed prefer Athletic Software Engineering. Positive comments include:

- I got accepted to an internship from a company and they put me through 3 rounds of WODs to test me before giving me the offer. I think the WODs in 314 helped.
- Something I've noticed in my 111 and 211 class is a lot of people having a terrible time because the student wouldnt put forward enough time outside of class to practice, resulting in a poorer performance in class. Athletic software engineering promotes programming as a more of way of life (because it requires a lot of outside practice before you get comfortable with any framework) than a homework assignment like a math problem set or English paper.

Did WODs help you become more confident in your software development capabilities?
80% of the students surveyed responded that WODs increased their confidence. Comments include:

- Yes. Working under pressure is something that I have not experienced much of. In the real world, there are deadlines, some of which are unfair (My brother works as an architect, and he has told me this numerous times). Thus, its imperative that one learns how to perform efficiently under pressure, when stakes are high (100 points is a lot!)
- I do. Its allowed me to think on my feet faster and approach problems in a calm and level manner.
- Not so much at first since I didn't do too great at first, but the more WODs I did, the more confident I did and my performance improved too.

Details on the Athletic Software Engineering initiative and the survey results are at:
Athletic Software Engineering (2017).

The Morea Framework

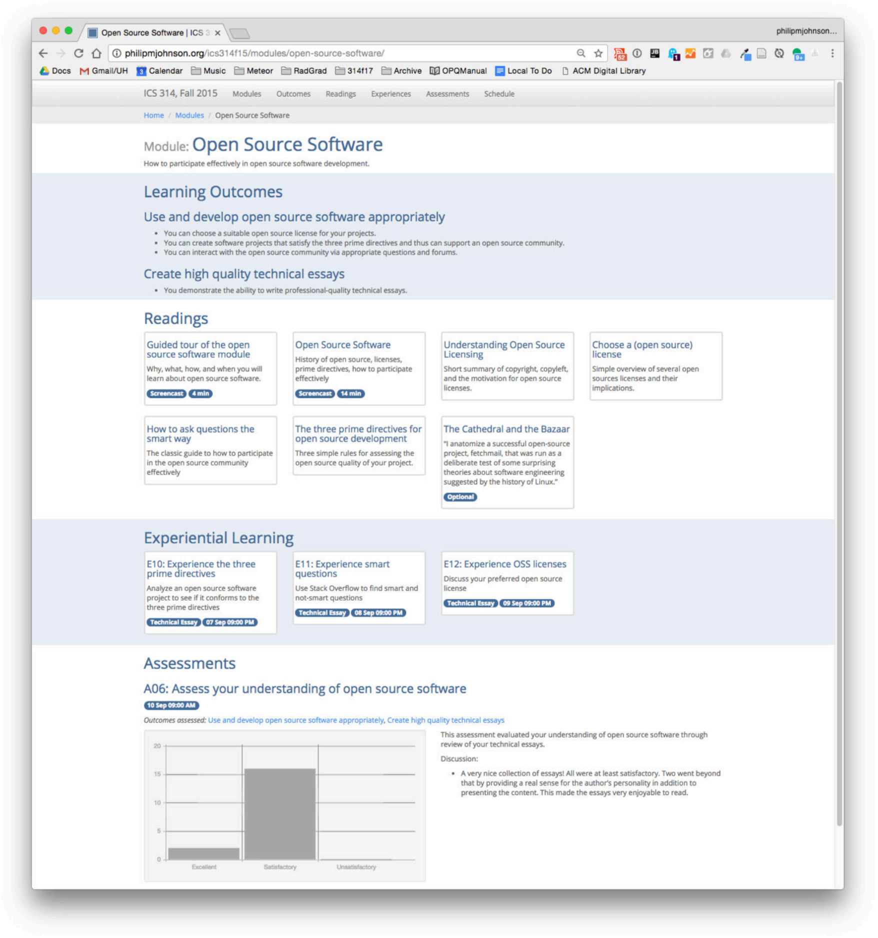
About five years ago, I became frustrated by the available technology for course websites. I think Laulima has a terrible user interface for both students and teachers. I tried building my course websites using WordPress for a few years, but no WordPress theme implemented the concepts important for a course website. So, I began designing a framework using the Jekyll static site generator and GitHub Pages which I named Morea, available at http://morea-framework.github.io/.

Morea is an acronym for a simple pedagogical design pattern: a course is a sequence of *Modules*, each with one or more learning *Outcomes*. Modules can combine one or more passive *Readings* (providing background) with one or more active *Experiences* (in-class work or homework). Finally, a module can contain one or more *Assessments* to help students determine if they've achieved the learning outcomes.

Since 2014, I've used Morea to build all my course websites. Several other ICS faculty (Lipyeow Lim, Henri Casanova, Carleton Moore) have also adopted it. Though I haven't made any attempts to publicize it, according to a quick Google search, there are Morea course websites at University College London, UH Hilo, Drew University, Hennepin Technical College, and the Jerusalem College of Engineering.

In my experience, students and teachers both benefit by using this pedagogical design pattern. The ability to view course content in multiple ways makes it easier for teachers to design well structured courses, and easier for students to understand what they need to learn and how to learn it.

Figure 2 illustrates a page for the "Open Source Software" module from ICS 314, with sections displaying Learning Outcomes, Readings, Experiential Learning, and Assessments for this specific module. The navbar provides links to "directory" pages for all Modules, Learning Outcomes, Readings, Experiences, and Assessments. There is also a calendar-based Schedule page showing due dates and which days of the semester are dedicated to which modules.



Module: Open Source Software

How to participate effectively in open source software development.

Learning Outcomes

Use and develop open source software appropriately

- You can choose a suitable open source license for your projects.
- You can create software projects that satisfy the three prime directives and thus can support an open source community.
- You can interact with the open source community via appropriate questions and forums.

Create high quality technical essays

- You demonstrate the ability to write professional-quality technical essays.

Readings

Guided tour of the open source software module
Why, what, how, and when you will learn about open source software.
Screencast 4 min

Open Source Software
History of open source, licenses, prime directives, how to participate effectively
Screencast 14 min

Understanding Open Source Licensing
Short summary of copyright, copyleft, and the motivation for open source licenses.

Choose a (open source) license
Simple overview of several open sources licenses and their implications.

How to ask questions the smart way
The classic guide to how to participate in the open source community effectively

The three prime directives for open source development
Three simple rules for assessing the open source quality of your project.

The Cathedral and the Bazaar
"I anatomize a successful open-source project, fetchmail, that was run as a deliberate test of some surprising theories about software engineering suggested by the history of Linux."
Optional

Experiential Learning

E10: Experience the three prime directives
Analyze an open source software project to see if it conforms to the three prime directives
Technical Essay 07 Sep 09:00 PM

E11: Experience smart questions
Use Stack Overflow to find smart and not-smart questions
Technical Essay 08 Sep 09:00 PM

E12: Experience OSS licenses
Discuss your preferred open source license
Technical Essay 09 Sep 09:00 PM

Assessments

A06: Assess your understanding of open source software
10 Sep 09:00 AM

Outcomes assessed: Use and develop open source software appropriately, Create high quality technical essays

This assessment evaluated your understanding of open source software through review of your technical essays.

Discussion:

- A very nice collection of essays! All were at least satisfactory. Two went beyond that by providing a real sense for the author's personality in addition to presenting the content. This made the essays very enjoyable to read.

Figure 2: *Morea Example: Open Source Software Module*

Courses.ICS

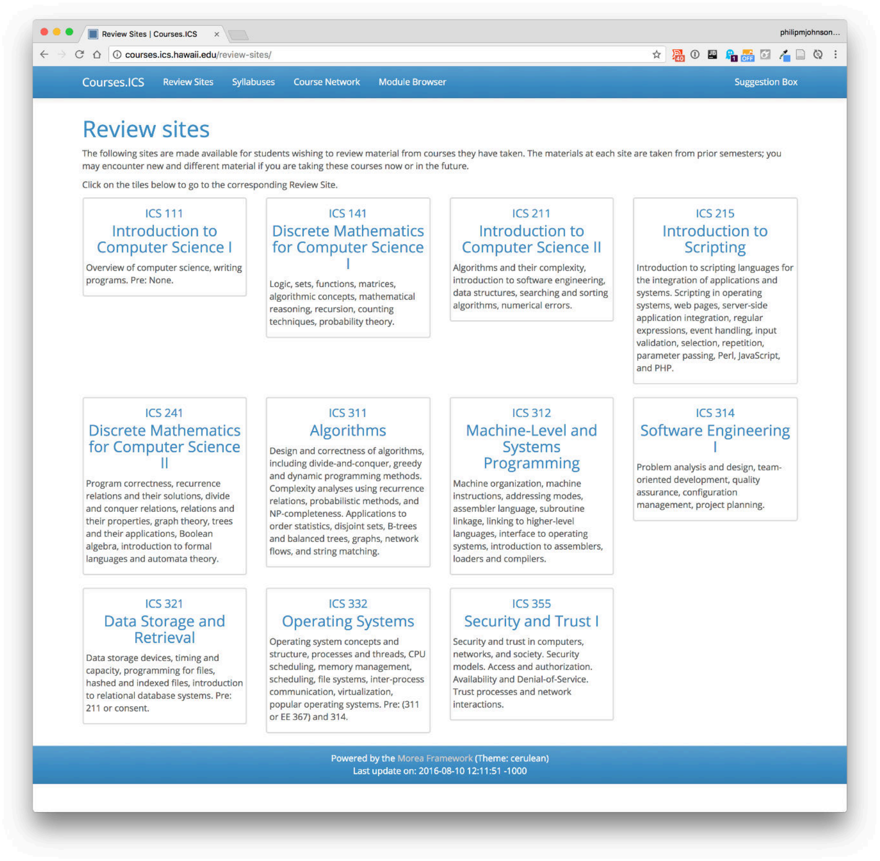
Students in ICS courses sometimes respond to material with one or both of the following questions:

- *To learn Y, you're expecting me to know X, but I never learned X in my previous classes!*
- *I don't understand why I'm learning Y; will I really need it in the future?*

Faculty often respond to the first question by reviewing the material they thought the students already knew, then complaining about having to do so in a faculty meeting. They respond to the second question by a brief exposition on its importance, which may or may not convince the students their need to know.

After a few semesters of using the Morea Framework, it occurred to me that I could build a site in which the standard material from all our required courses was represented as modules in a form amenable to subsequent review by students. This site could then represent the knowledge linkages between courses at the module level. So, for example, a student studying the Discrete Probability module in ICS 141 would know that this material would be necessary for them to understand the Probabilistic Analysis module in ICS 311.

Such a site could eliminate the need for faculty to hastily review forgotten material; they can just point students to the appropriate module in the appropriate prior course. In addition, students could see for themselves how modules at lower level courses are building blocks for modules in higher level courses.



Review sites

The following sites are made available for students wishing to review material from courses they have taken. The materials at each site are taken from prior semesters; you may encounter new and different material if you are taking these courses now or in the future.

Click on the tiles below to go to the corresponding Review Site.

ICS 111
Introduction to Computer Science I
Overview of computer science, writing programs. Pre: None.

ICS 141
Discrete Mathematics for Computer Science I
Logic, sets, functions, matrices, algorithmic concepts, mathematical reasoning, recursion, counting techniques, probability theory.

ICS 211
Introduction to Computer Science II
Algorithms and their complexity, introduction to software engineering, data structures, searching and sorting algorithms, numerical errors.

ICS 215
Introduction to Scripting
Introduction to scripting languages for the integration of applications and systems. Scripting in operating systems, web pages, server-side application integration, regular expressions, event handling, input validation, selection, repetition, parameter passing, Perl, JavaScript, and PHP.

ICS 241
Discrete Mathematics for Computer Science II
Program correctness, recurrence relations and their solutions, divide and conquer relations, relations and their properties, graph theory, trees and their applications, Boolean algebra, introduction to formal languages and automata theory.

ICS 311
Algorithms
Design and correctness of algorithms, including divide-and-conquer, greedy and dynamic programming methods. Complexity analyses using recurrence relations, probabilistic methods, and NP-completeness. Applications to order statistics, disjoint sets, B-trees and balanced trees, graphs, network flows, and string matching.

ICS 312
Machine-Level and Systems Programming
Machine organization, machine instructions, addressing modes, assembler language, subroutine linkage, linking to higher-level languages, interface to operating systems, introduction to assemblers, loaders and compilers.

ICS 314
Software Engineering I
Problem analysis and design, team-oriented development, quality assurance, configuration management, project planning.

ICS 321
Data Storage and Retrieval
Data storage devices, timing and capacity, programming for files, hashed and indexed files, introduction to relational database systems. Pre: 211 or consent.

ICS 332
Operating Systems
Operating system concepts and structure, processes and threads, CPU scheduling, memory management, scheduling, file systems, inter-process communication, virtualization, popular operating systems. Pre: (311 or EE 367) and 314.

ICS 355
Security and Trust I
Security and trust in computers, networks, and society. Security models. Access and authorization. Availability and Denial-of-Service. Trust processes and network interactions.

Powered by the Morea Framework (Theme: cerulean)
Last update on: 2016-08-10 12:11:51 -1000

Figure 3: *Review Site Page for Courses.ICS*

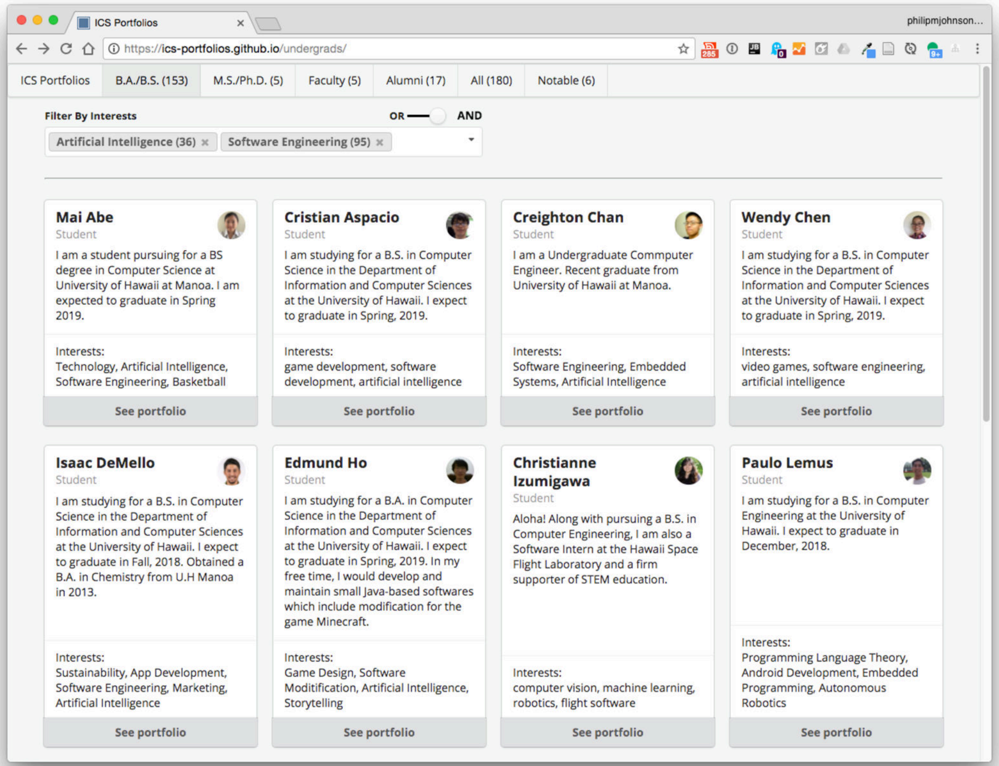
Figure 3 illustrates one page from the site that I built as part of this initiative, called Courses.ICS. The site currently provides review material for 11 ICS courses. During 2016, the site was visited approximately 16,000 times, of which approximately 5,000 visits were from users located in Hawaii. (It turns out that a review site for computer science is useful not just to ICS students!) Interestingly, the site was visited from all 50 states, with 943 visits from California and just 1 visit from a user in Maine.

TechFolios

About 10 years ago, I began requiring my software engineering students to develop a professional portfolio. The goal is to teach them how to create an online professional presence that allows them to go into more detail regarding their skills and accomplishments than is possible with a resume or LinkedIn profile. More importantly, the process of building a portfolio helps students identify gaps and weaknesses in their technical background while they still have time to fix it prior to graduation.

Initially, I required them to use WordPress, but I found that most students found it difficult to adapt existing WordPress themes for the purpose of a professional portfolio. Most online portfolio sites are designed for graphic art students. I decided to create a portfolio framework designed around the needs of computer science students called TechFolios, built using GitHub Pages, Jekyll, Semantic UI, and JSON Resume. Using TechFolios, students could create a modern portfolio site that includes a resume page without any custom coding (although the framework supports the development of custom themes for those students who want to go beyond the basics).

In TechFolios, students create a public JSON file that contains basic data about their portfolio in a


ICS Portfolios

https://ics-portfolios.github.io/undergrads/

ICS Portfolios | B.A./B.S. (153) | M.S./Ph.D. (5) | Faculty (5) | Alumni (17) | All (180) | Notable (6)

Filter By Interests — OR / AND

Artificial Intelligence (36) × Software Engineering (95) ×

Mai Abe
Student
I am a student pursuing for a BS degree in Computer Science at University of Hawaii at Manoa. I am expected to graduate in Spring 2019.
Interests:
Technology, Artificial Intelligence, Software Engineering, Basketball
See portfolio

Cristian Aspacio
Student
I am studying for a B.S. in Computer Science in the Department of Information and Computer Sciences at the University of Hawaii. I expect to graduate in Spring, 2019.
Interests:
game development, software development, artificial intelligence
See portfolio

Creighton Chan
Student
I am a Undergraduate Commputer Engineer. Recent graduate from University of Hawaii at Manoa.
Interests:
Software Engineering, Embedded Systems, Artificial Intelligence
See portfolio

Wendy Chen
Student
I am studying for a B.S. in Computer Science in the Department of Information and Computer Sciences at the University of Hawaii. I expect to graduate in Spring, 2019.
Interests:
video games, software engineering, artificial intelligence
See portfolio

Isaac DeMello
Student
I am studying for a B.S. in Computer Science in the Department of Information and Computer Sciences at the University of Hawaii. I expect to graduate in Fall, 2018. Obtained a B.A. in Chemistry from U.H Manoa in 2013.
Interests:
Sustainability, App Development, Software Engineering, Marketing, Artificial Intelligence
See portfolio

Edmund Ho
Student
I am studying for a B.A. in Computer Science in the Department of Information and Computer Sciences at the University of Hawaii. I expect to graduate in Spring, 2019. In my free time, I would develop and maintain small Java-based softwares which include modification for the game Minecraft.
Interests:
Game Design, Software Moditification, Artificial Intelligence, Storytelling
See portfolio

Christianne Izumigawa
Student
Aloha! Along with pursuing a B.S. in Computer Engineering, I am also a Software Intern at the Hawaii Space Flight Laboratory and a firm supporter of STEM education.
Interests:
computer vision, machine learning, robotics, flight software
See portfolio

Paulo Lemus
Student
I am studying for a B.S. in Computer Engineering at the University of Hawaii. I expect to graduate in December, 2018.
Interests:
Programming Language Theory, Android Development, Embedded Programming, Autonomous Robotics
See portfolio

Figure 4: *ICS Portfolios*

structured format. After a few years of use, I built a simple system that reads a data file containing the URLs of student portfolios, retrieves the public JSON file, and generates a public portal to all the associated portfolios. The goal of this site, called ICS Portfolios is to provide students, faculty, and the community with better visibility into the computer science and computer engineering student community.

Figure 4 shows a page from the ICS Portfolios site. The site currently contains 180 portfolios. The page illustrates the filtering feature, which in this case shows those students who have expressed an interest in both Artificial Intelligence and Software Engineering.

RadGrad

The RadGrad Project is my most ambitious educational initiative. I discuss it in detail in my research statement.

Community Initiatives

I have participated in two significant educational initiatives that transcend university boundaries: HACC and March for Science.

The Hawaii Annual Code Challenge is a month-long hackathon in which government agencies pitch challenges to teams of software developers. The goal is to "engage the local tech community in modernizing state functions and services and support IT workforce development."

I participated in HACC 2016 in a team that developed an application for the Oahu Community Correctional Center (Kipa). I found the experience so compelling that I joined the HACC organizing committee in

2017, and taught a 400-level special topics course in Fall 2017 focusing on HACC. Students enrolled in the course participated in HACC 2017 during the first month of the semester, then spent the remainder of the semester refining their application. This course was very successful: in fact, students from the course took first place, second place, and top college team (see this News Announcement).

In addition, during 2017 I served as one of seven principal organizers of the Hawaii March for Science. The goal of the March for Science, held on Earth Day, was to provide a non-partisan celebration of the scientific process, and to help build a diverse and inclusive scientific community of scientists, teachers, practitioners, and other stakeholders. This was an amazing community experience for me: from an initial meeting of six faculty and graduate students in a biology seminar room in February, we created an organization of over 70 active volunteers, raised over $12,000 in funding through tshirt sales, and had a verified attendance over 2500 people at the march in April. For more details on the experience, please see my post A newbie guide to organizing your first demonstration.

Future directions

If I join the Department of Electrical Engineering, I would like to continue these efforts while adapting them to the needs of students in the program, and participate in ongoing and new educational initiatives within the Department.

It is my understanding that EE 467, Object Oriented Software Engineering, has not been taught recently. I could begin teaching this course immediately using my materials for ICS 314 as a basis. I currently enroll many CE students in ICS 314, and I believe I could improve the course with a smaller, more tightly focused student demographic.

At the graduate level, I could introduce a graduate-level version of Software Engineering, which I have taught as ICS 613 for many years. I have also taught a graduate seminar course called "Software Engineering for the Smart Grid", which I believe would align nicely with other courses in renewable energy in the EE curriculum.

Also at the graduate level, I believe that the local community would respond positively to a professional masters program in software engineering. I believe the EE Department would be an excellent home for such a program.

If the faculty is interested, I could lead an initiative to create a review site for EE courses similar to Courses.ICS. I have helped develop NSF proposals with Tony Kuh in which we proposed a similar initiative to provide a "modularized" perspective on a renewable energy curriculum, and it was well received by reviewers.

I would like to promote the use of professional portfolios among EE students, and create a site similar to ICS Portfolios to showcase their skills and interests.

I would like to create an instance of the RadGrad system focused on the career goals and interests of EE students, and gather data to see if it has a positive impact on engagement, retention, and diversity within the Department. If this is successful, I would like to make RadGrad available to all students in the College of Engineering.

Finally, due to my participation in the VIP Program headed by Aaron Ohta, I know that the Department of Electrical Engineering embraces educational innovations. I look forward to learning from you and broadening my own horizons.