Software Engineering Workshops (SENG2021/3011) 2022 - 2023 Revision Debrief

Introduction

Over the course of 2022 and 2023, significant revisions to the Software Engineering workshop courses SENG2021 and SENG3011 were made. These revisions were a result of a review of the Software Engineering Degree conducted in 2021 which consolidated feedback from academics, industry representatives and students.

In 2022, SENG2021 (Requirements and Design Workshop) was revised from a focus of problem definition, requirements elicitation and prototyping to Application Programming Interfaces (APIs), microservices and software architecture.

In 2023, SENG3011 (Advanced Software Engineering Workshop) was revised from a focus of APIs and software implementation to DevOps and microservice ecosystems.

This writeup describes:

- Revised structure and delivery of SENG2021 and SENG3011;
- Student, staff and industry feedback; and
- Lessons learned from delivery of the workshops.

Motivation for Changes

During the 2021 Software Engineering Degree Review, a number knowledge and skill deficiencies in the SE curriculum were identified.

Knowledge	Skills
• DevOps; • Cloud computing; • Deployment; • Advanced testing techniques; • Quality engineering; • Frontend programming (in core SE); • Microservice architecture;	• Contributing to an evolving software system • Engineering changes in a large codebase • Inter-team collaboration; • Designing secure software systems; • Full-stack design and construction.

As a result of the review, the high level strategy of changes in the workshops was to:

- Move content on requirements engineering, UI/UX and prototyping from SENG2021 to DESN2000;
- Move content on APIs, microservices and software architecture from SENG3011 into SENG2021
- Create a new course within SENG3011 focusing on Advanced Software Engineering and DevOps.

SENG2021: Requirements & Design Workshop

Revised Learning Outcomes

1. Explain the principles and processes involved in the software development life cycle;
2. Synthesise and interpret requirements to create tangible architectural and technical designs in a team-based dynamic development environment;
3. Understand stack-based software architecture and acquire skills in full-stack design and construction
4. Build software systems using appropriate languages, libraries and frameworks;
5. Understand and apply the principles of project management in a dynamic software development environment; and

6. Contribute to system documentation for a software product.

Project: e-Invoicing

In the major project, teams are required to research, analyse the requirements and create a microservice for an API in an e-invoicing ecosystem - one of Invoice Sending, Receiving, Validation, Creation, Storage or Rendering. The teams follow the full software development lifecycle for this API, using project management techniques to plan the project and implement a MVP in an Agile fashion.

In the second phase, teams iteratively build a frontend application that orchestrates their API and other teams' APIs to deliver end-user value. This application is then pitched to assessors in a “Shark Tank” style presentation and a final design report.

Assessment

The major project is the primary vehicle by which learning takes place in the course and is broken down into four Agile Sprints.

Assessment	Due Date	Weighting
Sprint 1: Planning a Service	Week 3 Monday	10%
Sprint 2: Architecting a Service	Week 5 Monday	20%
Sprint 3: Building a Proof of Concept	Week 8 Monday	20%
Sprint 4: From Prototype to Product	Week 10 Monday	35%
Portfolio & Weekly Blogging	Weekly, and Week 11 Tuesday	15%

SENG3011: Advanced Software Engineering Workshop

Revised Learning Outcomes

1. Understand and explore the architecture of cloud-based microservice and API-based software systems.
2. Explore and work with modern DevOps and software toolchains used in industry development.
3. Synthesise and adapt large-scale software systems to changes in requirements, evolutions in technology, and the need for scalability.
4. Develop skills in leadership and management in a dynamic Software Engineering environment.
5. Deepen the ability to plan, design, implement, test and document quality software in an agile environment.

Project: Event Intelligence

In the major project, teams were required to select a data source by which to extract “events” to manipulate in an event-driven microservice ecosystem. Based on this data source, teams develop a microservice to either import, transform or export data from a shared bucket within the ecosystem.

In the second sprint, students deploy their service into a production environment with the assistance of DevOps tools and create a testing service to test another team’s API, ensuring Quality Assurance via peer review.

In the third sprint, students self-selected an initiative by which to improve the vitality of the ecosystem, and developed and pitched an application to assessors in a “Shark Tank”.

The work developed in SENG3011 is directly used by the Fintech AI Researc

Assessment

The courses is structured similarly to SENG2021:

Assessment	Due Date	Weighting
Sprint 1: Evolving the Ecosystem	Week 4 Friday	25%
Sprint 2: Validation and Delivery	Week 7 Friday	25%
Sprint 3: To Vitality and Beyond	Week 10 Wednesday	35%
Portfolio & Weekly Blogging	Weekly, and Week 11 Tuesday	15%

Common Pedagogy in both Workshops

Agile Software Development

The projects are completed in 2-3 week “sprints” as shown above, where students are provided with a sprint specification and a series of tasks to complete as a team. Each sprint has a generally similar structure, where the team plans the sprint, designs and develops, and moves their overall project towards a product. At the end of the sprint, there is a demonstration or “Shark Tank” presentation to assessors.

Agile project management techniques are taught and teams are expected to work in an agile way, holding standups, sprint planning and retrospective meetings, and using Jira and Confluence for managing their projects.

Weekly Blogging

Each week, students are required to write a short blog post reflecting on their project work. These blogs are prompted with a series of questions, but are free-form. Reflections on teamwork, challenges faced, learning from previous mistakes and planning ahead are encouraged.

Portfolio

At the end of the term, students individually submit a portfolio that reflects on their development of core skills across the term. This assessment form is derived from an older iteration of COMP1917, where students create a portfolio across three main areas:

- Time and Project Management
- Artisanship in SENG2021, Technical Maturity in SENG3011
- Teamwork, and Leadership in SENG3011

Self-Learning

The pedagogical structure of these courses deviates from the typical CSE template of lecture-tutorial-lab-assignment-exam by which content is assimilated. Instead, lectures provide a high-level introductory overview of principles, frameworks, technologies and tools. Students are then required to take this knowledge and undertake self-learning on the specific uses of the technology and application of the principles into their projects.

Mentoring

Instead of “tutors” the Software Engineering workshops have “mentors” - previous students of the course who have industry experience and the ability to guide others. Each team has a weekly 20-minute meeting with a mentor where guidance and feedback is provided. This support is typically less direct (e.g. debugging code) and more high-level (e.g. giving the team a contact by which they can receive technical help, or validating the direction of the team in a sprint).

Conflict Resolution and Communication

Teams are not given a form by which they can “peer review” their teammates at the end of term, or at the end of each sprint. Instead, the “peer review” is ongoing throughout the term. Students are required to have an open and continuous dialogue with their mentor about how the team dynamic is functioning, and bring up issues to their mentor either during project check-ins or elsewhere in the week. This allows

the mentor to mediate and support the team to resolve conflict immediately, rather than after the fact that is the case in traditional peer-review models.

Collaboration with Industry

Both courses collaborated with a number of industry partners to provide frameworks and guest speakers for the workshops:

Company	Involvement
Atlassian	• Providing a free Atlassian instance where staff can provision Jira and Confluence spaces for each group to run their project • Confluence used as the workshops Learning Management System platform • Guest Lectures on Advanced Deployment, Culture (SENG3011)
Nine	• Guest Lecture from George Wright on “Tools of the Trade” (SENG2021). • Provision of an archive of Australian Financial News for SENG3011.
Amazon Web Services	• Teams' e-invoicing microservices deployed using AWS (SENG2021) • Course ecosystem that students contribute to hosted using AWS (SENG3011) • AWS account provides sandbox environment where students can use services including Lambda, ECS, RDS, DynamoDB and S3 to develop infrastructure (SENG3011) • Guest Lecture on AWS Tools (SENG2021)
GitHub	• Team repositories managed using GitHub Classroom (SENG2021) • Course ecosystem code hosted using GitHub Organisation (SENG3011) • GitHub Actions used to run Continuous Integration checks on code • GitHub Actions used to run Terraform scripts to create/manage AWS resources (SENG3011)
NewRelic	• NewRelic observability tools used to provide metrics and logs on student microservices (SENG3011)
Optiver	• Guest lectures on Software Engineering and Software Performance • Sponsoring the SENG3011 Optiver Prize
Macquarie	• Sponsoring the SENG2021 Macquarie Prize
Tyro	• Guest lectures on Software Security and Software Quality
Bureau of Meterology (BOM)	• Access to weather datasets for SENG3011.

In addition to the above a mix of alumni individual guest speakers who work in industry came and spoke to students.

Infrastructure Cost

In 23T1, AWS Infrastructure for SENG2021 cost around $3,000, with each team hosting an Elastic Beanstalk application and in some cases other applications. For SENG3011, the hosting of the entire ecosystem, including at least 2 microservices for each team (predominantly AWS lambdas) cost $10,000. These costs were met from research resources, however a more stable way of managing these costs in future years needs to be found.

Student Feedback

Both of the revised workshops received significant positive feedback from students - in both myExperience and when informally providing feedback, including:

- Students were excited to be exposed to and gain hands-on experience with industry-grade tools not taught anywhere else in the degree. One student wrote “Using AWS as a cloud provider for the course is a great idea. It exposed me to AWS tools and resources that you’re not normally exposed to at uni”. Another student wrote “[The best thing was] being given the opportunity to delve into new stuff we haven’t learned about before (Terraform, Docker, AWS, Containers, Lambda, Feature Flags”
- Students appreciated the open nature of the project and chance to lead the creation of a full-stack product, with a student writing “I loved this course! We were given a lot of freedom to research and learn our own things, and the course really mirrors the real world and delivering products in a business”
- Students enjoyed receiving guest lectures from industry: “*very industry-relevant information presented, excellent teaching staff and student support*”
- Students appreciated the support from their mentors, with several students writing how engaging and helpful their mentors were.

In the course myExperience, taken by 86 out of 128 students enrolled in the course:

- 70% of students agreed with the statement “Overall I was satisfied with the quality of the course”
- 94% of students agreed with the statement “The course encouraged me to be self-directed in my learning”
- 80% of students agreed with the statement - The feedback helped me learn.
- 85% of students agreed with the statement “I felt part of a learning community”

In addition, students outside the Software Engineering degree have shown significant interest in taking these workshops or a similar variation due to its focus on industry practices and tools. In a survey with 101 responses from students in Computer Science, Information Technology and similar degrees, 86% of respondents indicated they would strongly consider taking a course if offered as an elective.

Several points of constructive feedback were also elicited - including:

- Many students struggled with the open-ended nature of the project and disliked this aspect. Aspects of this included:
  - Ambiguity in the project specification, with several students commenting that the specification was “vague” and “occasionally made us unsure on what was truly required and marked on”
  - Qualitative layers of marking and feedback;
  - Level of self-learning required for the course. This was reflected in the quantitative course survey as well, with only 57% of students agreeing with the statement “the course resources helped me learn”
- Some students disliked the weekly blogging and portfolio aspect, felt it was unnecessary and dislike writing.
- Some teams struggled with delegating learning about different parts of the toolchain, causing individuals to feel overwhelmed about the number of new tools they needed to understand. One student wrote “the large range of technologies was quite hard to handle, especially when doing two other courses at the same time”.
- While students liked the concept of inter-team collaboration, in practice it proved challenging with many teams unresponsive and continuing to work in a siloed fashion. One student wrote “[I] liked the idea of collaborating with other groups, but think execution needs some improvement”

Lessons Learned and Changes for the Future

For the teaching team, the rapid redevelopment of both workshops in the span of two years provided us with a series of insightful lessons to learn from on the design and delivery of the workshops.

Use of Amazon Web Services as a toolchain. In 23T1, SENG2021 used an approach of providing root accounts within an AWS organization to each group, while in SENG3011, individuals were provided with IAM accounts to access the ecosystem. The IAM approach worked far better than the root account approach as it allowed for tight control of costs. AWS proved to be a highly complex tool that took many hours of testing and learning to operate workably for the workshops.

Specialised experts in toolchains and course delivery. As seen in similar courses in other universities, one of the challenges of running courses rooted in rapidly evolving technologies is having staff up-to-date and trained in using them for both teaching and assisting students.

Delivering the course with subject matter experts in three different types of areas proved most successful:

- Case-study / requirements SMEs (who functioned as customers/product managers for teams and maintained the project specification). In the workshops, this was the LIC and associated PhD students mentoring the course.
- Tool / technology SMS providing technical support (casual academic mentors, ex-students with industry and toolchain experience)
- Course and assessment SMEs, who bridged between students and different parties (casual academic course admin/s with industry and toolchain experience)

Approaches to workshop progression and structure. While the workshops function as capstone courses, new skills are knowledge are taught in both. Due to the intensity of trimesters and feedback on the workload of the workshops, in future years the workshops will move away from having teams undertake “the complete software development lifecycle” in both workshops instead to a “pieces of the puzzle approach”. In particular, this means:

- Removing content on deployment and CI/CD from SENG2021, since SENG3011 covers deployment in depth
- Reducing the requirements for testing in the workshops, and redesigning COMP3141 to a course on Software Quality and Testing. In the workshops, the teaching team will do more testing of student microservices.

Evolution of the SENG3011 Ecosystem. Next year, students will be working on an existing ecosystem with this year’s services rather than a fresh one. There will be more established work items for teams to select, such as a backlog where teams can pick a service to implement. There will be an increased focus on the vitality and health of the overall ecosystem as it continues to grow. The SENG3011 sprint specifications will be adjusted accordingly.

General Challenges

Challenges in the Course Design

DevOps as a field is rapidly evolving and unstable. There is very little literature on DevOps, a shortage of academics who are well-versed in it and it is changing at a high rate due to its prevalence in industry. The course relies on mentors and admins with industry experience and an understanding of the tools.

Infrastructure is costly, and must be maintained. AWS is an ongoing cost for the course, and is complex to manage. In future years, we aim for costs to be come more predictable. In addition, backwards compatibility of external services used in course infrastructure is not guaranteed, requiring continual upkeep from course staff.

Scalable testing of student work. In SENG3011 since student work is used in the research infrastructure, finding ways to validate and reward teams' work in a scalable way is challenging. The research team will work on developing a process/pipeline by which a team’s microservice can be developed, tested, reviewed and validated by the teaching team.

Challenges in the Course Delivery

Security risks and safeguards. During the course AWS API keys were placed by students in a public GitHub repo - the infrastructure is risk-prone, particularly with many students working on the AWS account. AWS scans for public API keys and informs us and UNSW Cyber of incidents, and privileged access is enforced to mitigate this risks. They require careful management by course staff and vigilance from both staff and students.

Student inertia towards intractable problem-solving, written and oral assessment. The high student enrolments in large CSE courses, taken by COMP as well as SENG and other engineering students has led to most assessments taken by students in their first years of their degree being automarked and as a result close-scoped and highly tractable, where strict adherence to a specification is prioritised over creative thinking and dealing with ambiguity.

As a result, students struggle with the nature of Software Engineering workshops and complain regularly about the lack of specificity (see above). Similarly, the lack of written and oral assessment in previous courses is reflected in a general dislike of assessment over these mediums as well as a mediocre writing standard. These skills are core to Software Engineering in any setting, and cannot be developed solely in these workshops.

Lack of student engagement in aspects of the course considered ‘optional’. In both workshops (150 students) it was found that around 10 attended lectures in-person while less than 70 watched the recordings in early weeks, while less than 20 watched the recordings

in later weeks. It was also found that lectures most directly related to the immediate assessment items were watched, while those connected to longer-term course learning outcomes were not watched.

For example, the slides for SENG3011 *Microservice Testing*, which directly related to a major part of the Sprint 2 assessment viewed 757 times by students, while slides for *Influencing Behaviour* were only viewed 41 times. This lack of engagement is attributed to a deprioritisation of lectures over competing commitments (other courses' assessments, jobs) as well as more immediately gratifying sources of "content" (social media, streaming platforms) present in students' lives. Specifically for the workshops, since there is no final exam students feel less of an obligation to study the course content, rather instead to consume the bare minimum to succeed in the project.

Proactive conflict resolution is challenging. Across both workshops, some teams struggled to identify and bring forward conflicts early on in the term despite encouragement to do so. In many cases, teams would go through most of the term raising no issues, but in the later weeks when assessment was more intense and they were more stressed, raise a host of issues with seeds back in early weeks.

Conclusion

Overall, the changes to the workshops made were highly successful with many lessons learned and places to grow further in the future. We would like to thank all involved for their work.

The next major area of work for UNSW Software Engineering is revising COMP3141 (Software System Design & Implementation, Haskell) to a course on Software Quality and Testing. We will be taking advantage of a new lecturer (Dr Yuekang Li) expert in this area joining CSE in August 2023.

Please direct any questions on the content of this writeup to Nick Patrikeos (n.patrikeos@unsw.edu.au) and Fethi Rabhi (f.rabhi@unsw.edu.au).

Appendix A - Feedback on Tools used in Workshops

Tool	Good	Bad
AWS	• Very powerful • Realistic • APIs useful for bulk setup	• Very complex • No support for / provisioning for if you want a simplified version of an account with a subset of features for education purposes • No SSO • Very easy to incur large costs if you're not careful (not idiot-proof) • No direct support (can we get someone from AWS involved directly in the course?) • Need a way to test terraform scripts locally without having for waiting for AWS deploy
Jira + Confluence	• APIs useful for bulk setup • Great support (thanks Chris!) • Reliable	• Slow • APIs don't work sometimes • No SSO (NP working on this) • Need a way to group spaces and projects under a hierarchy, since the directories are becoming a big mess with spaces for each group for each term • Need a way to archive projects/spaces • Need a way to archive users that are no longer active • Filtering permissions within spaces is annoying – if you want to allow students to edit a single page you have to enable editing on the entire space, and then go and restrict the other pages to specific people. It'd be nice to be able to have page

		permissions independent of space permissions in this regard (maybe as a configurable option). • UI for Atlassian admin is clunky – slow, hard to search for users, pagination on users page is annoying • Group names approach is brittle, it would be good to have the group ID be a UUID which is unique and then be able to have duplicate group names, as opposed to a single name/id field which is limited to 10 characters. • Having configurable roles across the instance with preset permissions to put on spaces/projects would be ideal rather than having to specify exact permissions each time • Confluence search function doesn't work well
NewRelic	• Interesting and powerful tool	• Setting up log monitoring with NewRelic incurs a huge AWS cost as it makes thousands of requests to AWS per day • Some students felt that NewRelic was an unnecessary part of the toolchain given similar monitoring is available in AWS (CloudWatch)
GitHub	• GitHub actions were great • Good UX • It was great having students be able to select marketplace add-ons and request they be approved to use in their projects. Some add-ons only had a limited number of seats for the free version though.	• Need more minutes on the UNSW enterprise instance (BH working on this) • Self-selection for GitHub classroom enrolment is problematic. We had several students who messed this up and identified as the wrong user, selected the wrong group for their team repo. Created unnecessary admin overhead. • Need a way to group repositories under a hierarchy (like GitLab). Everything is flat within the organization, meaning that you can't organise repos into teams/terms/projects/course • GitHub outage right before deadline inconvenienced students