SE464 Software Design & Architectures

November 2018
Report by SE Curriculum Sub-Committee

- Joanne Atlee
- Werner Dietl
- Patrick Lam
- Leanora McVittie
- Derek Rayside

Calendar Description

SE 464 LAB,LEC,TUT 0.50 Course ID: 010035

Software Design and Architectures

Introduces students to the design, implementation, and evolution phases of software development. Software design processes, methods, and notation. Implementation of designs. Evolution of designs and implementations. Management of design activities.

[Note: Lab is not scheduled and students are expected to find time in open hours to complete their work. Offered: F]

Prereq: CS 246 or 247; Software Engineering students only.

Antireq: CS 446, ECE 452

Differences Between SE and CS Curriculum

SE students take CS138, which CS students have no equivalent of. CS138 introduces OOP and templates. Because of this, CS247 (which only SE students take) introduces design patterns, whereas CS246 (taken by CS students) does not. CS students are not introduced to design patterns until CS446.

Many recent instructors of SE464 have not been aware of these differences between the SE and CS curriculums. The Sub-Committee has several recommendations to address these matters.

Recommendation: SE Director to Archive CS247 Exams

Going forward, the Sub-Committee recommends that the SE Director archive the CS247 final exam for the future reference of the SE464 instructor. This will help reduce overlap between the courses.

Recommendation: SE464 Instructor gives early assignment/test on CS247 material

Going forward, the Sub-Committee recommends that the SE464 instructor gives an assignment or test early in the term that covers the CS247 material that was learned by that cohort. This will help reduce not only the actual overlap, but the perceived overlap between the courses. The early SE464 tutorial can be used to review this material.

Recommendation: No Course Project

CS446 should have a project with significant programming/implementation, since it is the first time the students have seen design patterns etc.

SE464 should not have a significant implementation project for three reasons:

- SE students have taken CS247
- SE students already have a lot of programming in 3B: SE390, CS343, CS348. Survey data says that 3B is one of the heaviest workloads in the SE program. (When CS 348 moves to 2B, ECE 358 will take its place and probably won't be lighter).
- The Capstone/FYDP provides a mandatory place in the SE curriculum where students must pursue a large open-ended project. CS students have an optional team project in their curriculum.

Expected Content in CS247 (not in CS246)

This material has been included in several past offerings of CS247:

- Representation Invariants + Abstraction Functions
- GoF Design Patterns
    - Visitor
    - Template
    - Factory
    - Decorator
    - Facade
    - Singleton
    - Adapter

    - Strategy
- Design By Contract
    - Pre-Conditions
    - Post-Conditions
- SOLID Design Principles
    - *Single responsibility principle*: a class should have only a single responsibility
    - *Open/closed principle*: software entities should be open for extension but closed for modification
    - *Liskov substitution principle*: objects in a program should be replaceable with instances of their subtypes without altering the correctness of that program
    - *Interface segregation principle*: many client-specific interfaces are better than one general-purpose interface
    - *Dependency inversion principle*: one should depend upon abstractions, not concretions

Design Patterns in Other Courses

- CS240 Data Structures
    - Composite
    - Iterator
- CS349 User Interfaces
    - MVC
    - Observer
    - Decorator
- CS343 Concurrency
    - Template

Core Topics in SE464

These topics should be in every offering of SE464:

- Architectural styles
- Architectural case studies

Optional Topics in SE464

The instructor should have latitude to include contemporary topics of interest. Roughly half the semester could be spent like a mini-graduate course on selected topics in software design and architecture. Topics might include, but not be limited to:

- Architecture description languages
- Type systems
- Mining software repositories

- Reverse engineering / architecture extraction
- Design formalisms and tools (e.g., Alloy, Spin, etc)
- Case studies from the systems programming community, e.g.:
    - Butler Lampson's Hints for Computer System Design is a classic in this area
    - MIT's 6.033 is a class on these case studies
    - ETH Zurich's 252-0217-00L Computer Systems course looks to be in this spirit
    - The seL4 fully-verified microkernel OS is an interesting case study
- Architecting systems with Machine Learning components
- Other topics taught by in graduate courses by SE faculty

Intellectual Approach Differs from Other Undergraduate Courses

The intellectual approach of SE464 and the Capstone/FYDP courses differs from other textbook-based undergraduate courses. This is intentional and inherent in the subject material. But it is a constant source of struggle for the students.

From the class rep:

> For my personal perspective, I think 464 is the first class, possibly ever for many of us, certainly the first core SE class, where we're being asked to come up with a solution without being given a method for getting there. This makes sense, and is important, and I think actually should be (and possibly already is) the main goal for the class: giving students the opportunity to think about a problem, pick a solution to apply, and justify their choice. If that is indeed the main goal for the class (which is a possibility, now I'm thinking of our lectures in that context) the problem comes in the communication of this to the students.
>
> All of our courses up to this point have been very "follow these steps to get the right answer", and we're approaching 464 as if it were one of these and then getting frustrated because we're not being given the steps. I think the course could be run in a way that really emphasizes the challenges of design decisions, and where it is explicit that most of the marks will be given based on your explanation of your thought process, rather than getting the "right" answer (this may be the case currently, I don't know, communication is a problem at the moment). Students would still hate the course, but they'd hate it because it's making them think in ways they're not used to, rather than hating it because they don't see its purpose.
>
> I'm not sure how much of that is relevant to a discussion specifically of the overlap between the two courses, but I've been thinking about this a lot since you emailed me last week, and it's sort of what I've settled on as the main issue with 464 at the moment.