Software Design and Architecture

Werner Dietl <wdietl@uwaterloo.ca>

Table of Contents

1. Overview 1
2. Contact 2
    2.1. Teaching assistants 2
3. Course objective 2
4. Course expectations 2
5. Overview of topics 3
6. Course material 3
7. Course schedule 4
8. Project 5
9. Assessment 6
10. Graduate Student Project 7
    10.1. Build a Software Tool 7
    10.2. Literature Survey 7
    10.3. Use an Advanced Software Development Tool 7
11. Outline 8
    11.1. Introduction (1 hr) 8
    11.2. Software Design Process Models (3 hours) 8
    11.3. Architecture/Design Representations (9 hours) 8
    11.4. Design Plans/Architecture (9 hours) 8
    11.5. Design Strategies and Methods (6 hours) 8
    11.6. Design Assessment (3 hours) 8
    11.7. Design Quality Assurance and Verification (3 hours) 9
12. Policies 9
    12.1. Academic Integrity 9
    12.2. Grievance 9
    12.3. Discipline 9
    12.4. Appeals 10
    12.5. Note for Students with Disabilities 10
13. Acknowledgments 10

1. Overview

SE2: Software Design and Architecture is the second course of the three software engineering capstone project courses, offered jointly by the Department of Electrical and Computer Engineering[1] and the David R. Cheriton School of Computer Science[2] at the University of Waterloo[3].

[1] https://ece.uwaterloo.ca/
[2] https://cs.uwaterloo.ca/

SE2 is offered under course codes ECE452, CS446, CS646, and SE464. Note that this section is **only** for ECE452[4], CS446[5], and CS646[6]. A separate section for SE464[7] is offered.

2. Contact

Lectures are held **Monday, Wednesday, and Friday from 9:30 to 10:20 in RCH 307**. There are no lab or tutorial slots. Students are expected to organize project meetings on their own.

My office hours are **by appointment** and will be held in DC 2522. See contact details[8].

Course material, announcements, and submissions will be handled through Learn[9].

Piazza discussion forums[10] are available for student discussions.

Begin all email subjects with `[ECE452]`, `[CS446]`, or `[CS646]`, as appropriate.

Try not to leave your questions until the last minute.

2.1. Teaching assistants

- Oleksii Kononenko[11], DC 3334B, `okononen`
- Xiaoye Lan, DC 3334B, `x5lan`
- Wei Sun, DC 3313, `w56sun`

Meetings by appointment.

3. Course objective

To introduce students to the software design process and its models. Representations of design/ architecture. Software architectures and design plans. Design methods. Design state assessment. Design quality assurance. Design verification. Group design and implementation of an application.

4. Course expectations

It is expected that students attend lectures and complete the required assignments. Lectures will often include a hands-on activity; participation in these exercises is essential to succeed in the class. Slides will

[3] https://uwaterloo.ca/
[4] http://www.adm.uwaterloo.ca/cgi-bin/cgiwrap/infocour/salook.pl?level=under&sess=1145&subject=ECE&cournum=452
[5] http://www.adm.uwaterloo.ca/cgi-bin/cgiwrap/infocour/salook.pl?level=under&sess=1145&subject=CS&cournum=446
[6] http://www.adm.uwaterloo.ca/cgi-bin/cgiwrap/infocour/salook.pl?level=grad&sess=1145&subject=CS&cournum=646
[7] http://www.adm.uwaterloo.ca/cgi-bin/cgiwrap/infocour/salook.pl?level=under&sess=1145&subject=SE&cournum=464
[8] https://ece.uwaterloo.ca/~wdietl/contact.html
[9] https://learn.uwaterloo.ca/d2l/home/146014
[10] https://piazza.com/uwaterloo.ca/spring2014/cs446/home
[11] https://cs.uwaterloo.ca/~okononen/

be provided via Learn. Any material discussed in class or in the required readings will be testable unless otherwise noted.

By the end of the course you should be able to:

- propose and analyze software architectures.
- explain the strengths and weaknesses of various architectural styles and design patterns / techniques.
- communicate and rationalize architectural and design decisions.
- ideate, justify, and implement software designs.
- evaluate, compare, and contrast different architectures and designs.

5. Overview of topics

- Software architecture, architectural styles, and architectural representations
- Software design, design patterns, design representations
- Software architecture and design conception, analysis, and communication
- Architecture and design recovery / reverse engineering
- Architecture and design visualization / understanding
- Cloud / grid computing architectures

6. Course material

While the course does not have a required textbook, much of the materials will be sourced from the first two texts; additional books are supplementary.

- Richard N. Taylor, Nenad Medvidovic, and Eric Dashofy. *Software Architecture. Foundations, Theory, and Practice*. Available in the library or for purchase (e.g., through Amazon.ca[12]). Slides for this book are available online[13].
- Ian Gorton. *Essential Software Architecture.* Available online[14] or for purchase (e.g., through Amazon.ca[15]). Slides for this book are available online[16].

[12] http://www.amazon.ca/gp/product/0470167742?ie=UTF8&tag=s01ae8-20&linkCode=as2&camp=15121&creative=330641&creativeASIN=0470167742
[13] http://www.softwarearchitecturebook.com/resources/
[14] http://bit.ly/9RF18P
[15] http://www.amazon.ca/gp/product/3540287132?ie=UTF8&tag=s01ae8-20&linkCode=as2&camp=15121&creative=330641&creativeASIN=3540287132
[16] https://docs.google.com/viewer?url=http://www.ug.it.usyd.edu.au/~iango/home/ESA-Slides.pdf

- Fred P. Brooks Jr. *The Mythical Man Month*. Available in the library or for purchase (e.g., through Amazon.ca[17]).

- Fred P. Brooks Jr. *The Design of Design*. Unfortunately not in the library but still available e.g. through Amazon.ca[18].

7. Course schedule

Lecture material is available through Learn[19].

This is a tentative schedule that might get adapted during the term.

Week	Date	Class	Project
1	May 5	Intro	Design Impressions
	May 7	Architecture Introduction	
	May 9	Project work: Build teams	
2	May 12	Architectural views & decomposition 1	Project groups
	May 14	*Patrick Lam*: Views & decomposition 2	
	May 16	Non-functional properties	
3	May 19	**Holiday: Victoria Day**	
	May 21	Architectural Styles 1	
	May 23	Architectural Styles 2	
4	May 26	Proposal presentations	Proposal deliverable
	May 28	Proposal presentations	
	May 30	Architectural Styles 3	
5	Jun 2	Architectural Styles 4	
	Jun 4	Architecture for Security	
	Jun 6	Architecture Description Languages	
6	Jun 9	Architecture Summary	
	Jun 11	Design Introduction	
	Jun 13	Design Patterns 1	
7	Jun 16	Design Patterns 2	
	Jun 18	Design Patterns 3	

[17] http://www.amazon.ca/gp/product/0201835959?ie=UTF8&tag=s01ae8-20&linkCode=as2&camp=15121&creative=330641&creativeASIN=0201835959
[18] http://www.amazon.ca/gp/product/0201362988?ie=UTF8&tag=s01ae8-20&linkCode=as2&camp=15121&creative=330641&creativeASIN=0201362988
[19] https://learn.uwaterloo.ca/d2l/home/146014

Week	Date	Class	Project
	Jun 20	MVC / MVP	
8	Jun 23	Prototype presentations	Prototype deliverable
	Jun 25	Prototype presentations	
	Jun 27	Prototype presentations	
9	Jun 30	**Additional Holiday before Canada Day**	
	Jul 2	Dependency Injection	
	Jul 4	Mini-Midterm	
10	Jul 7	Cloud/REST Architectures	Arch + Design deliverable
	Jul 9	Design tools 1	
	Jul 11	Design tools 2	
11	Jul 14	Final project presentations	
	Jul 16	Final project presentations	
	Jul 18	Final project presentations	
12	Jul 21	Final project presentations	
	Jul 23	Final project presentations	
	Jul 25	Final project presentations	
13	Jul 28	Review	
	Jul 30	Review	

8. Project

The project forms an integral part of this course. The goal of the project is to produce a significant application that performs some useful function (even better, something **awesome**!). This software must have a considered and defensible design and architecture.

There are only two real restrictions on the application idea itself: no database management apps will be accepted (e.g., simple CRUD apps); also, apps that require crowd buy-in are not acceptable (e.g., apps that would require large numbers of people to contribute content to be viably useful).

Mobile applications have been popular in previous years. To make the architecture interesting, aim to make the app executable on at least two mobile platforms (from: iOS, Android, BB10, WP8, FirefoxOS); the library has iOS devices that can be signed out. The app can work on tablet or phone form factors. Using sensors and external devices could also make for interesting apps.

However, other project ideas are welcome: if you have an idea for a project of a suitable size and complexity and find a project team to work with, propose it!

The projects will be completed in teams of four. You are free to select your own team; if you do not have a team or your team has less than four members, try using the course Piazza forums to organize a team.

If all else fails, please talk to me and I will set you up. Each of the deliverables for the project can be considered an assignment.

Projects will have a difficulty scale applied to them by the instructor and TAs. The scale formula will be:

(project + bonus) * scale = final project grade

Scale will range between 0.75 and 1.0. The components of the scaling mark will be determined by:

5 completeness (compared to proposal)
5 utility
5 polish
10 difficulty

There will also be various sources of bonus marks during the term; each will be worth 2%:

- Best pitch
- Best prototype demo
- Best final demo
- Accepted to curated App Store

Note: The expectation is that you will work approximately 12 hours per week on this course; at least 8 of these hours will be on the project. Given that the course lasts 13 weeks, each team member is expected to work on the project at least 100 hours. You should be able to accomplish something pretty great in this time; please make the most of this opportunity.

9. Assessment

Deliverable	**Date**	**Format**	**Value**
Design background	May 5	In Class	Pass/Fail
Project groups	May 12	Learn	Pass/Fail
D1: Proposal presentations	May 26	In Class	5%
D2: Prototype demo	June 23	In Class	5%
D3: Arch + Design	July 7	Learn + exam	30%
D4: Presentation + video	July 14-25	In Class	10%
Final Exam	TBD	Written	50%

All "In Class" deliverables include a Learn submission.

You must pass the final exam and all pass/fail assignments to pass the course.

10. Graduate Student Project

For graduate students only: in addition to the class project, you will perform an individual graduate project. The graduate project is worth 25% of your grade; this will come by compressing the value of your final exam and project grade to 75% of your total mark.

There are two deliverables for the graduate project:

- Project proposal. Before you undertake your project you will need to submit a proposal for approval. The proposal should be short (1-2 pages in ACM format). The proposal should include a problem statement, the motivation for the project, a set of objectives you aim to accomplish, and a set of milestones. I will read these and provide comments. The proposal is not for marks but *must* be completed in order to pass the course. This will be due on May 19 @ 09:00 via Learn.

- Written report. The required length of the written report varies from project to project; all reports must be formatted according to the ACM format and submitted as a PDF. This artifact will constitute 100% of the graduate project grade. This will be due on July 30 @ 09:00 via Learn.

Three types of graduate projects are possible, as listed below.

10.1. Build a Software Tool

The goal of this style of project is to identify some architecture/design problem developers encounter in practice, find some solution, and validate that the solution helps with the initial problem. I would recommend drawing upon your experience as you write code to identify some problem that has inhibited you in the past and attempt to fix it.

The outcome of this project will be a short (5-6 page) paper describing the problem, your solution, a comparison to related approaches, and some form of validation.

10.2. Literature Survey

The goal of this kind of project is to gain a more complete understanding of a topic relevant to this course. The outcome of this project will be a critical summary of the state-of-the-art on your selected topic; this summary should be 8-10 pages. It is essential that this summary synthesizes the surveyed literature to identify important themes, findings, and open questions.

10.3. Use an Advanced Software Development Tool

The goal of this project is to provide a validation of some previously-existing development tool from the research community. The tool you validate must be related to the course material. The outcome of

this project will be a 6-8 page paper describing your experience with the tool outlining its strengths, weaknesses, and avenues for future improvement.

11. Outline

This is the high-level outline provided by the CS department[20]; this course will follow the general guideline, but will be adjusted according to your feedback, interests, and experience.

11.1. Introduction (1 hr)

Why design? Input, output, and constraints of the design process. Types of design. Relationship to software quality and evolution. Design in more mature implementation technologies.

11.2. Software Design Process Models (3 hours)

Design as search. Design spaces. Design state, goal structure, generative design operations, early quantitative evaluations, control of design process. Basic models of design (transformational, plan/ architecture driven). Relationship to other life-cycle activities.

11.3. Architecture/Design Representations (9 hours)

What should be represented (structure, behaviour)? Informal representations of design, examples of design notations. Formal representation of design. Domain specific architecture descriptions. Role of standards, reference architectures. Design documentation.

11.4. Design Plans/Architecture (9 hours)

Review of small/medium scale plans (data structures, programming language structures, concurrency). Plans/architectures for common types of software systems (translators, embedded, real-time, user interface).

11.5. Design Strategies and Methods (6 hours)

Design strategies. Selected methods: object modeling technique, structured design, real-time, user interfaces. Methods for design with off-the-shelf components.

11.6. Design Assessment (3 hours)

Assessment dimensions and factors affecting their relative importance. Design tradeoffs. Evolvability/ understandability criteria. Design complexity metrics. Assessment strategies (analytical, simulation, rapid prototyping), example: response time/throughput estimation.

[20] https://cs.uwaterloo.ca//current/courses/course_descriptions/cDescr/CS446

11.7. Design Quality Assurance and Verification (3 hours)

Design reviews, scenarios and test cases, testing of executable design representations. Verification of properties.

12. Policies

12.1. Academic Integrity

- In order to maintain a culture of academic integrity, members of the University of Waterloo community are expected to promote honesty, trust, fairness, respect and responsibility. [See the academic integrity[21] site for more information.]

12.2. Grievance

- A student who believes that a decision affecting some aspect of his/her university life has been unfair or unreasonable may have grounds for initiating a grievance. Read Policy 70[22], Student Petitions and Grievances, Section 4.

- When in doubt please be certain to contact the department's administrative assistant who will provide further assistance.

12.3. Discipline

- A student is expected to know what constitutes academic integrity[23] to avoid committing an academic offence, and to take responsibility for his/her actions.

- A student who is unsure whether an action constitutes an offence, or who needs help in learning how to avoid offences (e.g., plagiarism, cheating) or about "rules" for group work/collaboration should seek guidance from the course instructor, academic advisor, or the undergraduate Associate Dean.

- For information on categories of offences and types of penalties, students should refer to Policy 71[24], Student Discipline.

- For typical penalties check Guidelines for the Assessment of Penalties[25].

[21] http://www.uwaterloo.ca/academicintegrity/
[22] http://www.adm.uwaterloo.ca/infosec/Policies/policy70.htm
[23] http://www.uwaterloo.ca/academicintegrity/
[24] http://www.adm.uwaterloo.ca/infosec/Policies/policy71.htm
[25] http://www.adm.uwaterloo.ca/infosec/guidelines/penaltyguidelines.htm

12.4. Appeals

- A decision made or penalty imposed under Policy 70 (Student Petitions and Grievances) (other than a petition) or Policy 71 (Student Discipline) may be appealed if there is a ground.
- A student who believes he/she has a ground for an appeal should refer to Policy 72[26], Student Appeals.

12.5. Note for Students with Disabilities

- AccessAbility Services[27], located in Needles Hall, Room 1132, collaborates with all academic departments to arrange appropriate accommodations for students with disabilities without compromising the academic integrity of the curriculum. If you require academic accommodations to lessen the impact of your disability, please register with the AccessAbility Services at the beginning of each academic term.

13. Acknowledgments

Thanks to Derek Rayside and Reid Holmes for sharing their experience and materials from previous iterations of this course, in particular CS 446, Winter 2014[28].

[26] http://www.adm.uwaterloo.ca/infosec/Policies/policy72.htm
[27] https://uwaterloo.ca/disability-services/
[28] https://cs.uwaterloo.ca/~rtholmes/teaching/2014winter/cs446/index.html