Week 1 Software Engineering Studio 1B

Yulei Sui

University of Technology Sydney, Australia

Administration

- Lecturer: Yulei Sui
- Email: yulei.sui@uts.edu.au
- Office: CB.02.12.202
- Telephone: +61 2 9514 4427
- Important message will be announced on the subject homepage on Canvas and urgent messages also sent to you via email.
- Tutor/TA: Guanqin Zhang (guanqin.zhang@uts.edu.au)
- Subject topic: Software Analysis

Goal of the Software Studio Subjects

- Practice software engineering skills to solve real world problems.
  - (software requirement, quality, performance, security, etc.)
- On the path to software developer/tester/analyst through practical projects.
  - Build systematic and specialized tools for software diagnosis and analysis.

Software Engineering Studio 1A Review

What you have learned in Software Engineering Studio 1A?

Software Engineering Studio 1A Review

- General **software agile** methodology
  - Discovering and understanding software requirements via team charter and user stories.
  - Practicing software development tools to build up a prototype.

- Apply engineering synthesis and design processes to foster **iterative learning**.
  - Devops through multiple sprints and trouble shootings.

- **Work effectively in a team** to conduct and **manage engineering projects**.
  - Collaborative effort of self-organizing and cross-functional teams via management tools (e.g., trello)

Software Engineering Studio 1B (Software Analysis Studio)

Teaching and Learning Strategies

Where are we heading in SES 1B?

Software Engineering Studio 1B (Software Analysis Studio)

Teaching and Learning Strategies

Where are we heading in SES 1B?

- Two hours class
  - First hour: teaching
  - Second hour: practicing, development and group discussion
  - Project-driven with guidance, materials, examples, and project specifications.
  - Though we will form groups, individual evaluation is through each assignment.

Software Engineering Studio 1B (Software Analysis Studio)

Teaching and Learning Strategies

Where are we heading in SES 1B?

- Two hours class
  - First hour: teaching
  - Second hour: practicing, development and group discussion
  - Project-driven with guidance, materials, examples, and project specifications.
  - Though we will form groups, individual evaluation is through each assignment.

- 1A vs 1B (Software Agile vs Software Analysis)
  - Harder than 1A? Yes & No. Implications:

Software Engineering Studio 1B (Software Analysis Studio)

Teaching and Learning Strategies

Where are we heading in SES 1B?

- Two hours class
  - First hour: teaching
  - Second hour: practicing, development and group discussion
  - Project-driven with guidance, materials, examples, and project specifications.
  - Though we will form groups, individual evaluation is through each assignment.

- 1A vs 1B (Software Agile vs Software Analysis)
  - Harder than 1A? Yes & No. Implications:
  - **More programming practices** but **less** work on **documentations**
  - **System programming with C/C++** and get yourself with some **graph algorithms**
  - Understanding of **compilers** and **source code analysis** principles
  - **Open source development** and GitHub

Subject Learning Objectives

Canvas website 41094 (`https://canvas.uts.edu.au`)

- Apply system programming skills to solve complex software engineering problems.
- Learn to write high-quality source code through developing open-source software tools.
- Apply systematic software engineering synthesis and design processes to develop automated software/program analysis tools.
- Apply systematic approaches to conduct and manage software engineering projects.
- Work efficiently in a team to develop open-source software products.

Subject Schedule

Canvas website 41094 (`https://canvas.uts.edu.au`)

Project topic/goal : **develop an automated software analysis checker using C++ to analyze the information flow of a C program**.

Week	Content	Assignment
1	Subject overview and introduction to software analysis	-
2	C++ programming practice	-
3	C++ programming for graph traversal	Assignment-1 (25%) due date:15/03/21
4	LLVM compiler and intermediate representations	-
5	Graph representation of code	-
6	Control dependence in software analysis	-
7	Control dependence analyzer implementation	Assignment-2 (20%) due date 16/04/21
8	Data dependence in software analysis	-
9	Data dependence analyzer implementation	Assignment-3 (25%) due date 10/05/21
10	Taint information flow devops	-
11	Taint information flow devops	-
12	Taint information flow checker implementation	Assignment-4 (30%) due date 24/05/21

Subject Schedule

Canvas website 41094 (`https://canvas.uts.edu.au`)

Project topic/goal : **develop an automated software analysis checker using C++ to analyze the information flow of a C program**.

Week	Content	Assignment
1	Subject overview and introduction to software analysis	-
2	C++ programming practice	-
3	**C++ programming for graph traversal**	Assignment-1 (25%) due date:15/03/21
4	LLVM compiler and intermediate representations	-
5	Graph representation of code	-
6	Control dependence in software analysis	-
7	**Control dependence analyzer** implementation	Assignment-2 (20%) due date 16/04/21
8	Data dependence in software analysis	-
9	**Data dependence analyzer** implementation	Assignment-3 (25%) due date 10/05/21
10	Information flow checking devops	-
11	Information flow checking devops	-
12	**Information flow taint checker** implementation	Assignment-4 (30%) due date 24/05/21

Marking and Plagiarism

Marking and late submission:

- Please refer to rubrics for each assessment before you start at Canvas
- Late submission is strongly discouraged. Late submission of assignments will incur the following penalties: 10% of the total possible mark for that piece of assignment for each day past the deadline
- For example, if an assessment receives a mark of 70% out of 100% and is one day late, it will receive a mark of 60% out of 100%

Plagiarism:

- UTS will adopt a uniform set of penalties for all assignments in all courses
- A wide range of penalties. Please refer the Subject outline
- `https://www.uts.edu.au/research-and-teaching/learning-and-teaching/assessment/preventing-plagiarism`

Subject Materials and Resources

- Static Value-Flow Analysis Framework for Source Code
  - https://github.com/SVF-tools/SVF
  - https://github.com/SVF-tools/SVF-Teaching
- Compilers: Principles, Techniques, and Tools Hardcover, https://www.amazon.com.au/Compilers-Alfred-V-Aho/dp/0321486811
- LLVM Compiler https://llvm.org/
- Anders Møller and Michael I. Schwartzbach, Static Program Analysis, https://cs.au.dk/~amoeller/spa/spa.pdf