Advanced Software Engineering
Agenda

- Syllabus
- Review of Software Engineering Concepts
- UML
- Self-introduction
CSc 8350
ADVANCED SOFTWARE ENGINEERING

♦ Instructor: Xiaolin Hu
♦ Email: xhu@cs.gsu.edu
♦ Phone: 404-413-5716
♦ Office: 25 Park Place Building, Room 746
♦ Hours: Tuesday/Thursday, 1:00pm - 2:30pm
♦ TAs: Sai Deepika Gopala (sgopala1@student.gsu.edu), Office Hour: Monday, 2:00-3:30pm, 650 M, 6th floor, 25 Park Place

Prerequisites: CSC 4350/6350 Software Engineering or equivalent

♦ Course Web Page: http://www.cs.gsu.edu/xhu/CSC8350/csc8350.htm
Textbooks

- Textbook
E. Gamma, R. Helm, R. Johnson, J. Vlissides, “Design Patterns”, Addison-Wesley, 1995

The literature
- Journals:
  - ACM Transactions on Software Engineering and Methodology (TOSEM)
  - IEEE Transactions on Software Engineering (TSE)
  - Annals of Software Engineering (Kluwer)
  - Automated Software Engineering (Kluwer)
  - International Journal of Software Engineering and Knowledge Engineering (IJSEKE) (World Scientific)
  - CROSSTALK: The Journal of Defense Software Engineering
  - etc.
The course…

is not about

- Programming
- Software Project Development

is about

- Design
- Modeling
- Research

So, What is modeling?
Credit: Slides from Daniel Westreich
Department of epidemiology
UNC-Chapel Hill
How is this related to software development?

From the design point of view, a software system has two (three) aspects:

- Structure (architecture)
- Behavior
- Data – for data intensive systems

The structure and behavior are driven by the function requirement.

Given a description of a software system to be developed, such as a software similar to iCollege, how do you develop the software? How do you design the structure/behavior of the software?

How are these handled in UML?
Objectives of the Class

To enable students to

(1) comprehend and appreciate software reuse and design patterns;

(2) understand a variety of advanced research topics in software engineering;

(3) conduct literature review on specific topics and give professional presentations.

(4) learn and employ cutting-edge software engineering tools (e.g., modeling tools, program analysis tools, testing tools) and understand their theoretic and/or methodological foundation
Content of the course

The contents of the course are roughly divided into three parts

1) Software reuse and design patterns.
2) System modeling and design as exemplified by FSM, Petri net, and Discrete Event System Specification (DEVS).
3) Advanced research topics in software engineering.

♦ For the “advanced topic” part, students will review related literature and present the results to the entire class.

♦ The course will be conducted in a seminar format.
What is DEVS?

- DEVS stands for Discrete Event System Specification
- The DEVS formalism is based on dynamic systems theory
- DEVS allows to develop models in a modular hierarchical fashion
- It also supports simulation of models
Grading

The course will include

1. Two take-home midterm exams (part I and part II) -- Together 40%. This is broken down as part I: 45%; part II: 55%

2. Literature review on a selected topic of software engineering (includes report and a presentation) -- 32%. This is broken down as 60% presentation, 40% report

3. Hands-on project (includes report and demonstration) -- 18%. This is broken down as 40% demonstration, 60% report

4. attendance – 10%  Attendance will be taken randomly and is mandatory for specific announced classes.

♦ Both project and research report are team work with 2 members
  ♦ Note: the project group and research group may be different.
**Advanced Topic Review**

- Each group has 45 minutes (subject to change) presentation.
- Tow types of advanced topic review projects
  - The first type is research-oriented. It is more like the traditional literature review. The goal is to cover an advanced topic and review the “state of the art” for that topic. For this type, “state of the art” and breadth are crucial.
    - **Examples:** The State of the Art in End-User Software Engineering
    - **Examples:** Software Engineering in Industrial Automation: State-of-the-Art Review
  - The second type is learning-oriented. It covers a useful, interesting, and relatively new topic and introduces the topic using well-designed examples to help others to have in-depth understanding of the subject. For this type, depth and understandability are crucial.
    - **Examples:** Petri Net Model, Software Testing Techniques
Hands-on Project

- Modeling project based on DEVS using the DEVSJAVA environment
- Some cutting-edge tools related to software engineering, such as
  - Program analysis tools
    - Control flow and dataflow analysis. Example: FindBugs
  - Automated software test tools
  - Extreme Programming tools or supporting tools
  - Software security tools
  - Cannot be based on the examples provided by the tool. Develop your own project or examples.

- Both the advanced topic review and hands-on project have to be related to “Software Engineering”. Consider the following examples: 1) data mining; 2) bio-informatics; 3) could computing; 4) IoT
- The hands-on project may be “extended” from the advanced topic review. But keep in mind that the hands-on project needs to be a concrete, self-contained project.
Old sample literature review topics

- Open Source Software
- Real-time embedded software, Real-time UML
- Software for Smart Mobile Devices (Apps Development)
- Software Visualization
- Testing and Test Methods
- Self-Adaptive software, Autonomy-oriented computation
- Reliability and dependability
- Verification and validation, Formal methods
- Software quality assurance, metrics
- Software Architectures
- Extreme programming, agile methods
- Aspect-oriented programming
- Management issues in software development
- System specification, requirement specification
- Web service and service-oriented architecture (SOA)
- Software Engineering Education

(note: for whatever topic you choose, you should obtain the general background in the field as well as two or three specific approaches that are being researched)
How to find a topic for reviewing?

- Something you are interested in or wondering about.
  - E.g., I wonder how the big data influences software engineering…
  - I wonder how the cloud computing paradigm is changing the software engineering practices…
  - I wonder how security concerns impact software engineering…
  - I wonder how AI, deep learning have an impact on software engineering…
- Something related to your own work.
- State of the art survey-type of papers.
- Find papers from the programs of recent software engineering conferences.
- As an example, a list of software engineering topics can be found from ICSE 2017: http://icse2017.gatech.edu/?q=main_conference_program
Resources (list not complete)

- Note: most papers can be accessed through IEEE digital library and/or ACM digital library
- IEEE Transactions on Software Engineering
- ACM Transactions on Software Engineering Methodology
- Springer Software and Systems Modeling
- IEEE Software
- ICSE, the International Conference on Software Engineering (http://www.icse-conferences.org/)
- ASE, IEEE/ACM International Conference on Automated Software Engineering (http://www.ase-conferences.org/)
- FSE/ESEC, the ACM SIGSOFT Symposium on the Foundations of Software Engineering (FSE) (http://www.sigsoft.org/fse20/)
- OOPSLA, Object-Oriented Programming, Systems, Languages & Applications, (http://www.oopsla.org/oopsla-history/)
Class Rules

- No cell phone
- No walk in and out in class
- No leave earlier
- Zero tolerance for any plagiarism activities
- Active participation (instead of passive listening) is strongly encouraged. Due to the nature of this advanced class, the delivery of this class highly depends on your active participation. So please ask questions, respond to my questions, and make comments
Review of Software Engineering

Elevator Example
Software Engineering Activities

- System specification, requirement specification – functional requirement, non-functional requirement
- analysis, use cases, sequence diagram
- system design, system architecture
- object design, design patterns
- implementation, programming language
- testing, verification, validation, simulation-based testing, formal methods
- software lifecycle model
- management issue
- modeling and design, FSM, Petri net, DEVS
- UML
- OO, component-based design, Service-oriented computing
The heart of object-oriented problem solving is the construction of a model.

- **UML** stands for Unified Modeling Language.
What do you have to do right now?

- Form a team of 2 members.
- Decide your topic for advanced topic review. Email me your topic by Jan. 22.
Self-introduction to help team formation