CS 3773 Software Engineering Lecture 1

Dr. Mark Robinson
Office: NPB 3.350

Me

- Dr. Mark Robinson : Mark.Robinson@utsa.edu
- Practicing *unlicensed* software engineer since 1993
- Domain: speciality insurance market
- Platforms: service-oriented applications (web app, server processes, web services)
- Agile: small teams; fast, iterative development

This Class

- Good understanding of the Software Engineering process
- Useful tools, techniques (methods), and experience
- Academic and Industry value

Schedule

Week 1: Jun 1	Intro to SE	
Week 2: Jun 6/8	Principles,Processes,Agile	A1 out
Week 3: Jun 13/15	RE and the SRS	
Week 4: Jun 20/22	UML	A1 in
Week 5: Jun 27/29	Planning/Scheduling	A2 out
Week 6: Jul 6	Mid-term Wed. 5/6	
Week 7: Jul 11/13	Architecture	
Week 8: Jul 18/20	Design Patterns	A2 in, Project out
Week 9: Jul 25/27	Quality and Risk Mgmt.	
Week 10: Aug 1/3	Testing	
Week 11: Aug 8/10	Final 1:30p to 4p	Project in

Textbook(s)

- **Software Engineering Theory and Practice**, 4th Ed., by Shari Lawrence Pfleeger and Joanne M. Atlee ISBN 978-0-13-606169-4
- Optional: UML 2 and the Unified Process, 2nd Ed., by Jim Arlow and Ila Neustadt ISBN 0-321-321127-8

Exams

- Midterm: Wednesday 7/6 (partial review week before)
  - Worth 25%
- Final Exam: Friday 8/12 1:30pm
  - Covers 2nd half of course only
  - Worth 25%

Assignments

- 2 assignments
- Each worth 10%
- Available on a Mondays, due 2 Sundays after assigned by midnight

Project

- User interface prototyping project
- Worth 30%
- Due midnight on day of final exam

What is Software Engineering?

- “[the use of] tools, techniques, procedures, and paradigms to enhance the quality of their software products.”
- IEEE defiition: “The application of a **systematic**, **disciplined**, **quantifiable** approach to the development, operation, and maintenance of software; that is, the application of engineering to software.”

Different from Comp. Sci.?

- Computer Science: “the science that deals with the theory and methods of processing information in digital computers, the design of computer hardware and software, and the applications of computers.”
  - from http://www.dictionary.com/browse/computer-science
- Software Engineering uses the products of computer science as tools to build software (to solve higher-level problems)

Software

- A computer program that directs the operation of a computer to accomplish a specific task
    - http://www.dictionary.com/browse/software
- In SE: the primary product of the development effort. May or may not involve actual programming (usually does)

Enterprise Software

- Software that is built for a specific business process; has characteristics like
  - Persistent data (usually via a database)
  - Accumulate a lot of data (database size can grow quite large)
  - Multiple, concurrent users and lots of screens
  - Integrates with other enterprise systems and data
  - Involves behavior that is specific to the business process
- Can be large, medium, or small software systems (e.g., lines of code, database size, # of users)

A Software System

- Software products usually does not operate in isolation (i.e., without interaction with external entities)
  - Even games today interact with external data/programs
- **Entity**: a human or other software/system that interacts/ interfaces with a software program
- **Boundary**: determines what is included in the development of the software product (entities outside the boundary are not part of the development effort but produce input for the system and/ or consume its output)

More Software System Terms

- **Behavior**: something the software does (function, use case, etc.)
- **Objects**: data used by the activities; can be records, classes/templates, fields, etc.
- **Relationships**: which objects are used in which activities
- Note: It is important to know which behaviors and objects lie inside the system boundary and which do not

Breaking Down Complexity

- Software can be extremely complex. Two very powerful techniques used to engineer software:
    - **Abstraction**: a simplification that allows focus on some pertinent aspect of the software; a model (e.g., a blueprint of a home showing electrical wiring)
    - **Modularization**: divide and conquer; break entire behavior into independent, logical parts; design/build/test/integrate each part (sometimes in parallel)

SE Stages/Activities

- Activity: a type of work performed during software development:
    1. requirements analysis and definition
    2. system design
    3. program design
    4. programming
    5. unit/integration/system testing
    6. delivery
    7. maintenance

SE Process

- A process is an approach for the engineering of a particular software product
    - I.e., a specific configuration of the Activities
- SE encompasses the approach AND the tools/ methods that are within the approach to build the software
- The process should suit the project, SHs, budget, etc.

Stakeholders

- Anyone who benefits from the software being produced (i.e., has a stake in the success of the software)

Who Are Stakeholders?

- Clients
- Investors (client and developer)
- Developers:
  - Programmers
  - Testers
  - Analysts
  - Designers/Architects
  - Managers
  - Trainers
  - Other Support Staff
- End-users

Quality

- A very important characteristic of good SE
- Lots of different ways to judge software quality
  - User, Manufacturing, Product, Value views
  - We will define this in more detail later
- Ultimately, quality is not boolean and should a customer-dictated requirement

NASA Shuttle Launch System

- Completely software-controlled
- Involves billions of $, human lives, and global reputation
- 420 KLOC
  - 17 errors found in 11 versions
  - Commercial equivalent would have at least 1000 bugs

http://www.fastcompany.com/28121/they-write-right-stuff

Not Perfect, But Ultra-high Quality

- 1/3 of development occurred before any code
- 40,000 pages of specifications
- Adding new GPS feature caused 2,500 more pages
- Specifications are almost pseudo-code

Remember: a good design leads to good implementation

NASA Approach to Bugs

- Fix what caused the bug
  - Unclear API: improve document quality
  - Insufficient tests: re-do test coverage
  - Improper use of tools: re-train dev
- Validate/Review at all levels
  - 85% of bugs found BEFORE testing even started

NASA Cost

- 260 people
- $32 million
- 1 year development
- Is this the level of quality ALL software should have?

Bug Terminology

- **Fault/Defect/Bug**: a human error while performing a software engineering activity
  - in code: “x += 1;”   instead of   “x -= 1;”
- **Failure**: incorrect software behavior due to a fault
  - the “x += 1;” defect in a rocket launch system causes a rocket to explode during launch
- Does every fault cause a failure? Vice versa?