COSC 425: Software Engineering I

(some slides courtesy Ralph Johnson and Darko Marinov)

What is (not) SE?

- Not just software programming
  - Individual vs. team focus
  - Delivering value
- Not just a process
  - Field that studies several different processes

Some definitions of SE

"The application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software."

- IEEE Standard 610

01-3

Some definitions of SE

"Software engineering is the establishment and use of sound engineering principles in order to obtain economically software that is reliable and works efficiently on real machines."

- Peter Naur and Brian Randell (1968) at the first NATO conference of software engineering

01-4

Some definitions of SE

"Software engineering is the part of computer science which is too difficult for the computer scientist."

- Friedrich Bauer

01-5

Developing Software

- Product
- Process
- People

01-6

Things we study in SE

- Process
- Tools
- Techniques
- Models (of software development)
- Modeling (the systems we develop)
- The complete life cycle: requirements through maintenance and support

01-7

Activities in the Process

Requirements, architecture, design, documentation, management, planning, configuration management, testing, metrics, debugging, reverse engineering, refactoring,…

01-8

Software Varies

- Size
- How humans interact with it
- Requirements stability/knowledge
- Need for reliability
- Need for security
- Portability
- Cost

01-9

Microsoft PowerPoint

- Size: large
- Interactiveness: high
- Requirements: frequent new features
- Reliability: moderate
- Security: low (at least used to be)
- Portability: high
- Cost: high

01-10

Space shuttle software

- Size: moderate to large
- Interactiveness: low
- Requirements: stable
- Reliability: very high
- Security: low
- Portability: low
- Cost: high

01-11

EBay online software

- Size: moderate
- Interactiveness: high
- Requirements: frequent new features
- Reliability: moderate
- Security: high
- Portability: low
- Cost: low

01-12

Your example

- Size:
- Interactiveness:
- Requirements:
- Reliability:
- Security:
- Portability:
- Cost:

01-13

Software process?

- Pressman: "A framework for the tasks that are required to build high-quality software"
- IEEE 1074: "A set of activities performed towards a specific purpose"
- Johnson: "The steps a particular group follows to develop software"

01-14

Activities of IEEE 1074

- Project Management
  - Project initiation
  - Project monitoring and control
  - Software quality management
- Development
  - Requirements
  - Design
  - Implementation

01-15

IEEE 1074

- Post-development
  - Installation
  - Operation and support
  - Maintenance
  - Retirement
- Integral processes
  - Verification and validation
  - Software configuration management
  - Documentation development

01-16

Defined processes

- eXtreme Programming (XP)
- Rational Unified Process
- Scrum
- Crystal Clear
- Cleanroom
- Bazaar
- …

01-17

XP Process

- Roles
  - XP: Customer, Developer, Coach
- Activities
  - XP: Write stories, planning game, test-first, pair programming, continuous integration, refactoring
- Work products
  - XP: User stories, tests, code

01-18

Purpose of course

- Learn, modify, and follow a process (XP more or less)
- Learn and execute modern SE techniques and practices (not just XP):
  - Requirements, planning, SCM, testing, metrics, documentation, UML, designing, refactoring, automated build, pair programming, IDE
- Learn how to follow a process
- Learn how to change/improve a process

01-19

Project

- Group projects (~5 students)
- Various topics
- Opportunity to practice
- Process should start with XP… which parts? modifications?
- Must document the process you use
- Must convince me you follow the process you documented

01-20

Problems with projects

- Too vague
  - Can't tell if task has been done
    - "Study xx"
  - Tasks too big
  - Too much creativity
- Too small
  - Make bigger, but prioritize tasks so you can reduce scope

Project lifecycle

- Propose project
- Form team
- Develop a project
- Deliver code, tests, documentation
- Graded on process during development + quality of product you deliver = process (a bit less) + product (a bit more)

Project requirements

- Must manage "customer" requirements
- Must practice proper configuration management (e.g., version control, etc.)
- If necessary, must have a transition plan for how to keep it going after course ends
- Demonstrate use of test first approach
- Demonstrate use of UML to document where applicable (e.g., use of object-oriented languages)

01-23

Generic ideas

- Web site or tool for a customer
- App for cell phone (Android or iPhone)
- Eclipse plugin
- Adding feature to open source system
- Math or CS educational tool

01-24

Project Description

- General Description
- Motivation
- Languages, libraries, frameworks, platform
- Risks/challenges
- User stories, use cases (a few examples)
- Required/actual skill sets!!

01-25

Team Creation

- Present proposals
- Form teams (with input of instructor)
- Project success largely depends on the people!

01-26

Next: Planning

- Start working on your project
  - Initial project description
  - Initial user stories
  - Identify required resources, tools, software, etc. --- begin training as necessary
- Time tracking starts next week
- Peer evaluations start next week

01-27

Academic year outline

- Project
- Homework
  - Learn tools and practices for the project
  - Develop important skills (e.g., Presentations)
- Exams
  - Midterm
  - Final Presentations
- Grade: homework assignments, project, midterm, presentations… emphasis is on project and its components

01-28