The Role of Software Engineering

Brief overview of relationship of SE to managing DSD risks

Outline
- Role of software engineering
  - Purpose of software engineering
  - Software engineering areas addressing DSD risks
    - Process management
    - Requirements negotiation and specification
    - Design for collaborative development
- Project Introduction: Mobile Face Recognition System
- In class assessment questionnaire

Objectives
- Key property distinguishing DSD
  “The key phenomenon of DSD is coordination over distance.” – J. Herbsleb (2007)
- Overall goal is to produce the right software on time with available resources
- Implies the need to coordinate the work among distributed team members such that
  - Every task gets done
  - Tasks are done in the right order
  - Resources are allocated so tasks are done on time by the right people
Purpose of SW Engineering

- The purpose of software engineering is to gain and maintain intellectual and managerial control over the products and processes of software development.  
  - "Intellectual control" means that we are able make rational technical choices based on an understanding of the downstream effects of those choices.  
  - Managerial control means we are able to make rational choices about development resources (budget, schedule, personnel) to deliver software on time and within budget.  

*Memorize this!*

Meaning of Intellectual Control

- Software development progresses through a sequence of decisions:  
  - Decisions about requirements (e.g., tradeoffs, priority)  
  - Design decisions (e.g., first decomposition)  
- Earlier decisions affect the difficulty of later decisions.  
- Ensuring that we end up with the desired properties requires making the right decisions in the right order.  
  - E.g., cannot add properties like security late in the game, Windows demonstrates this vs. OSX (Unix private address space).  
- Being in control means we can:  
  - Decide in advance the functional and non-functional requirements the software should satisfy  
  - Proceed systematically through the steps of software development to produce a system meeting those requirements.

Meaning of Managerial Control

- Managerial control means we are able to make rational choices about development resources.  
- Real projects have finite set of resources: time, people, money  
- Must choose where, when, and how much resources are allocated.  
- Being in control means we can:  
  - Decide in advance the level of resources needed to deliver software meeting requirements  
  - Deliver that software on time and within budget.
Analogy to Driving a Car

- Driving a car
  - In control: decide in advance where we want to go, how long, it should take, how much fuel it will take
  - Out of control: end up at a different destination, takes twice as long as expected, uses twice as much fuel, etc.
- Managing a software project
  - In control: decide in advance what capabilities and properties the software will have, how long it will take, how much effort
  - Out of control: software is delivered with less or wrong capabilities, delivered late, over budget
- Many software projects are out of control in this sense
  - Perfect control is not possible
  - Requires constant feedback and correction

Relevant SE Areas

- Risks: in DSD communication difficulties and context difference lead to coordination and control problems
- Approach: apply software engineering processes, methods, and tools to mitigate risks
- In this course, we will focus on a few key areas
  - Process: How should we managing distributed resources?
  - Requirements: How do we ensure everyone is building the right system?
  - Software Architecture: How do we design for distributed development?
- Quality assurance: How do we check our control?

Example: Requirements Risks and Goals

- Requirements risks in DSD
  - Many different stakeholders with different goals and different understanding
  - Risk of building the wrong system
  - Risk that cannot manage change effectively
- Requirements goals
  - Negotiate a common set of requirements among distributed stakeholders
  - Communicate requirements to all the developers so there is a common view of what should be built
  - Plan for change throughout development
  - Processes in place to manage change
  - Design for ease of change
Example: Design Risks and Goals

- Relationship between system design and communication overhead
  - Must decompose the system into work assignments for distributed teams
  - System components depend on one another
  - Risk: More dependencies require more need for communication and coordination
    - i.e., Coupling ↔ communication
- Software architecture goals
  - Decompose the system into work assignments that are as independent as possible
  - Work can proceed concurrently and independently
  - Little need for communication
  - Little need for one team to wait for another
- How do we design the software architecture to have these properties?

How should components be distributed among teams?

Goal: distribution of work on components requiring least inter-team communication

Topics & What You Should Learn

- SE learning goals
  - Learn how to use SE methods to address specific development problems
  - Be able to explain why specific choices or decisions are the right ones
- Software processes and project planning
  - Which life-cycle model to use
  - What kinds of activities and roles support distributed development
  - How to allocate work to distributed teams
- Software architecture
  - How to use architecture to manage dependencies
  - Which design principles and methods support concurrent development
  - Role of interfaces as contracts
- Software requirements
  - How to ensure mutual understanding of expected behavior and system qualities
- Software verification and validation
  - The role of feedback in maintaining control
  - Effective reviews
  - Distributed testing
Summary

- DSD presents certain specific problems that make it difficult to control software development
  - Create the desired system
  - Maintain budget and schedule
- Purpose of software engineering is to provide technology and techniques for maintaining control
- We will cover some specific techniques that address key DSD problems
  - Only a subset of useful methodologies

Project Introduction:
The Face Recognition System (FRS)

Proposal

- We propose a system to recognize and retrieve information about individuals from their pictures.
- To keep the system portable and appealing to the on-the-go user
  - The system is implemented on a smartphone device such as the iPhone and/or Android
  - The picture may be taken then and there
  - May supply context-relevant information (allow different databases to be accessed)
Basic Application

1. Take a picture of John Smith
2. The system searches a pre-established database of faces
3. If recognized, retrieves relevant information

Architectural Concept

- Take advantage of existing web services
- Support a family of possible applications

Assignments

- Assessment
- Review Concept of Operations (ConOps)
Questions?