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
• Teams and Project Planning
• To Do List

Objectives

• What distinguishes 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

• Defn: 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
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 intellectual control means we can systematically build the right software:
  - 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 recourses 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 manage 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 of requirements changing during development
- 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: Greater dependencies require more communication and coordination
- 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
Q: Properties should the components have?

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
Schedule and Deliverables

• PKU students are proposing to develop Windows Phone apps.
  – C#, Xna, Silverlight
• Different from the past where we have pre-defined the application
  – Opportunity: opportunity to choose application, define requirements as a team
  – Risk: need to converge quickly to allow sufficient development time

Example Project from 2011: Notable Faces Service (NFS)

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 data-bases to be accessed)

Basic Application

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

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

Anatomy of a Good Project

- Supports independent development
  - Clear division into work assignments
  - Few dependencies
- Complex enough to support expansion/contraction (useful subsets, increments)
- Complex enough to require some effort in all the parts of development (requirements, design, testing, etc.)
- Simple enough to finish in ten weeks

To Do

Team 1
- Feng, Zeyu
- Harris, Riley
- Nguyen, Christine
- Rogers, Erick

Team 2
- Ceccarelli, John
- Elliott, David
- Spore, Aaron
- Zhang, Ran

Team 3
- Cerny, Austin
- Feldman, Michael
- Liu, Jiazhang
- Neill, Morgan

Questions?