Introduction to Distributed Software Development
CIS 423/510

Stuart Faulk
University of Oregon

Lian Yu
School of Software and Microelectronics
Peking University

Outline

• Rationale for Distributed Software Development (DSD)
• Software engineering challenges of DSD
• Course structure and goals
  — Course plan
  — Student resources

Growth of DSD

• Software development by globally distributed teams used by companies of all but the smallest size
• Outsourcing growth by an order of magnitude over past few years
• Over half of the Fortune 500 companies
  — Software industry: Google, Microsoft, IBM, etc.
  — Other industry: telecom, cell phone, etc.
• If you work in industry, you will likely work in a distributed team
  — Also true of academic research!

Rationale for DSD

• Expanded pool of trained workforce
• Getting closer to customers and using local expertise to acculturate products
• National policy (regulatory locality requirements)
• Difference in development costs
• Promise of round-the-clock development that could lead to shorter intervals
**Evolution of DSD**

Global Sourcing is Here to Stay and is Evolving

<table>
<thead>
<tr>
<th>Evolving From...</th>
<th>Developing To...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Offshore labor primarily in India</td>
<td>Labor is multiple geographies around the globe</td>
</tr>
<tr>
<td>Offshoring provides cheaper labor</td>
<td>Offshoring provides efficient access to larger talent pool and leading edge technologies</td>
</tr>
<tr>
<td>Numerous tactical vendors</td>
<td>3.5 strategic outsourcing vendors</td>
</tr>
<tr>
<td>Predominantly maintenance</td>
<td>New development lifecycle, design and test</td>
</tr>
<tr>
<td>Predominantly technical programming</td>
<td>Executing, business process outsourcing, T&amp;M and infrastructure</td>
</tr>
<tr>
<td>Global delivery is a strategy</td>
<td>Global delivery (chain) is the standard</td>
</tr>
</tbody>
</table>

*IBM Global Initiative*

**Benefits Come with Risks**

- With DSD benefits come increased risks compared to similar co-located developments
- Schedule delays – same work takes longer
- Higher risk of failure
- Reduced product capabilities
  - Decreased functionality, qualities
  - Doesn’t meet some customer requirements
- Increased cost
  - May cost more in spite of lower labor costs
  - Schedule delays and rework increase costs

**Observed Difficulties (1)**

- Nature of a software project
  - Software development produces a set of interlocking, interdependent work products
  - E.g. Requirements -> Design -> Code
  - Implies dependencies between tasks
  - **Implies dependencies between people**
- Successful development requires effective coordination between people and tasks!
  - Must coordinate work (need product A to produce product B)
  - Must coordinate schedule (must finish A before starting B)
  - Must coordinate people (person P has expertise need to produce A but is busy)

**Working Definition**

- **Distributed Software Development (DSD):** teams in geographically distant locations collaborate to produce the work products of a software development
  - Synchronize in phases of the life cycle
  - Collaborate on artifacts from requirements to code
  - Coordinate activities among members of distributed teams
Observed Difficulties (2)

- Key property distinguishing DSD from co-located development
  
  “The key phenomenon of DSD is coordination over distance.” — J. Herbsleb (2007)

- All software projects require coordination
- Suggests that coordination at a distance is different
- Managing these differences is a central issue in DSD

Informal Communication Pathways

- In co-located projects, people build up informal ways of coordinating work
  - Shared process view (implicit or explicit)
  - Common vocabulary, viewpoint
  - Clear idea of expertise, responsibility
  - Free flow of information through informal channels
  - Common language, culture, backgrounds help avoid misunderstanding
  - Relatively good understanding of relationships
    - People to tasks
    - Task dependencies
    - Professional and social
- Consider your 422/522 experience
DSD is Different...

- In DSD many of the mechanisms for coordinating work are absent or disrupted
  - Much less communication
    - Temporal distance
    - Socio-cultural distance, e.g., language
    - Spontaneous communication declines rapidly with distance
  - Less effective communication
    - Fewer overlapping work hours
    - Low bandwidth links (e.g., email and other asynchronous)
  - Lack of awareness
    - Lack context hence knowledge of history, relationships
    - What people are doing day to day, concerns, availability
  - Incompatibilities
    - Differ in tools, processes, work products
    - Leads to confusion, misunderstandings, inconsistencies

Software Development Problems

- Manifests as problems in coordination and control of software development
  - Difficulty establishing requirements (eliciting, understanding, negotiating)
  - Difficulty effectively distributing work
  - Difficulty detecting and correcting conflicting assumptions
  - Difficulty detecting and correcting slips in schedule
  - Difficulty managing change (especially requirements)
  - Difficult managing development resources (schedule, personnel, budget)
- Similar to traditional SE problems, but more intense
  - Work takes longer
  - Requires more effort

Useful to View as Risks

- Examples of increased project risks due to communication and control difficulties
- Risk of building the wrong software (behavior or qualities)
  - Misunderstand the requirements
  - Miss requirements or fail to address them
  - Functions needed by distributed team members not implemented or implemented incorrectly
- Management risks (budget, schedule, personnel)
  - Balancing workload
  - Developing common understanding schedule, sequencing
- Fundamental issue we will address is how to mitigate DSD risks

Software Engineering Emphasis

- Emphasis on life-cycle management and teamwork in a globally distributed context
- Understand coordination and communication challenges of distributed development
- Participate in planning and coordination of distributed teams
- Learn effective software engineering techniques to address DSD risks
- Learn effective communication and collaboration skills supporting distributed teamwork
Course Structure

Approach and Rationale

• Gain real experience with problems by simulating an industrial DSD project
  – Similar to way in-class project simulates co-located development
• Collaborate with students at Peking University on a software project
  – Each team will be roughly half UO, half PKU
  – Work together on a software application
  – Collaborate over the web to create, review, and present the results of development
• Learn to apply SE principles, methods and tools to support long-distance collaboration

Remote Collaboration

Weekly Schedule

• Three class meetings a week
  – Mix of lectures, discussions, group exercises
  – Some lecture times or parts thereof will be used for team meetings and project discussions
• Meetings with the instructor
  – Small group discussion of SE and project ideas
  – Design reviews
  – Progress reviews
  – Course assessment
**Resources**

- Class web page:
  - Syllabus: updated with assignments, lecture slides, and class videos
- Project: documents and activities related to the project
- Tools: descriptions of the development and communication tools we will use
- Readings: download reading assignments
- Announcements
- FAQ: ask questions or find answers about the project, lectures, or anything else.
- Assembla workspaces
  - On-line collaboration environment for project teams

**Teams**

- Will try to have 4 teams of UO/PKU students
- Goals and expectations
  - Learn SE skills by encountering and overcoming DSD problems in simulation
  - Learn teamwork across cultural boundaries
  - Active participation with your whole team is critical to this
    - It is expected that you will fulfill the duties of your role(s) in your team by actively collaborating with your teammates
    - Interacting with your teammates at PKU especially important
- Also need student liaison to work with student at PKU (student-to-student)
  - Communicate about course issues
  - Help us provide resources and help needed

**Likely Projects**

- Most likely a Windows 8 phone app
- Negotiate with PKU students
- Client/Server

**Student Evaluation**

- Primarily interested in what you learn about software engineering and teamwork
- Rough decomposition
  - Quality of development artifacts: 35%
  - Quality & functionally of code: 20%
  - Project management & communication: 15%
  - Teamwork and participation: 20%
  - Written evaluations: 10%
Assignments

• Familiarize yourself with the class web pages
• Fill out team formation survey and bring to next class
• Identify Liaison(s)
• Reading (by Friday): Cultural Surprises in Remote Software Development

Questions?