Software Engineering
CS5704: First Class

Instructor: Shawn A. Bohner
Voice: (703) 538-8374
Email: bohner@nvc.cs.vt.edu
Teaching Assistant: Sepna Georges
Voice: (703) 538-8381

Agenda

▲ Introductions
  ● Semester Schedule
  ● Guidelines and Expectations
▲ Chapter 1 – The Software Product
  ● Break
▲ Chapter 2 – The Software Process
▲ Survey of Interests
▲ Homework Assignment
### Spring Semester Timeline

<table>
<thead>
<tr>
<th>Jan</th>
<th>Feb</th>
<th>Mar</th>
<th>Apr</th>
<th>May</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class Begins Product &amp; Process</td>
<td>Analysis, Design, &amp; Architecture</td>
<td>Mid-Term Exam</td>
<td>SW Metrics &amp; Testing Strategies</td>
<td>Final Exam</td>
</tr>
<tr>
<td>PM Metrics &amp; Estimation</td>
<td>Cross-Life-Cycle Process</td>
<td>Testing Techniques</td>
<td>Object-Oriented Development</td>
<td>Advanced SWE Topics</td>
</tr>
</tbody>
</table>

16 weeks, 13 sessions... So much to do and so little time... be prepared to work hard and glean the value!

### Guidelines and Expectations

- **Grading:**
  - Homework/Quizzes (10%)
  - Project, Mid-Term, and Final (30% each)

- With the exception of today, plan to have read the assigned material before class

- Plan for an average of 9 hrs of work a week outside of class

- Once the website is up, check it regularly

  - Handouts, when available, will be on web

- You may work in teams on some assignments – be fair to other team members!

- What you learn in CS5704, you will use in most IT or software endeavors

  - CS5704 is a good investment with ROI
2000+ Technology Headlines

▲ IT budget brakes are still on
  ● Network spending on the rise
  ● IT spending in non-IT budgets
▲ IT shops undergoing radical role change
  ● Increasingly functioning as integrators, solution implementers, and sourcing managers
▲ Increasing packaged applications in IT portfolio
▲ ADM “rigor” eroding with demand for IT agility
▲ IT staff shortage inhibiting key technology delivery

Source: 2000 World Wide Benchmark

Following Y2K, software is critical to the success of most private and public sector organizations

2000+ Business Reality: Increasing IT-Intensive Initiatives

▲ Business-Triggered Technology Initiatives
  (CRM, ERP, E-Business, …)
▲ Event-Triggered Initiatives
  (Y2K, Euro, M&A, Reorgs, …)
▲ Technology Infrastructure Initiatives
  (Network Upgrades, Data Center Consolidation, …)
▲ Modernization Initiatives
  (Legacy Consolidation, Stove-Pipe Integration, …)

Software performance drives need to manage manifold risks induced by the growing portfolio of IT initiatives
Chapter 1
The Product

Purpose

▲ The purpose of this chapter is to introduce software as a “product” designed and built by software engineers for consumers
  ● Software is important because it is used by a great many people worldwide
  ● Businesses today cannot operate without it

▲ Objectives
  ● Outline software as a product
  ● Outline key aspects of software products
  ● Examine software engineering’s role
What is Software?

Software is a set of items or objects that form a “configuration” that includes:

- programs
- documents
- data...

That’s one way to look at it...

▲ Software is part of a computer system that is intended to change
▲ Business changes and its computer systems must respond
▲ Software is engineered, not manufactured
▲ Software is intangible
▲ Software is complex

Software is suppose to change... otherwise it would in the hardware!
Software Doesn’t Wear Out

▲Software doesn’t change with age or “wear out” with use! However, ...

- Software “ages” or becomes “obsolete” with a changing environment
- Software deteriorates or “degrades” with continued changes

Wear versus Deterioration

- Increased failure rate due to side effects
- Change in failure rate over time
Software Design Degradation

The Original Software Design...

- Easy to Understand
- Components well isolated to facilitate change
- Isolation supports change validation

...Plus a few “Changes”

- Increased size and complexity
- Reliability of system degrades, errors creep in
- At some point, it’s unmaintainable

Information Lose Due to Relentless Change

Baseline       Change       Change       Change
Code           Set 1            Set 2            Set N
Design         Spec’s
The Cost of Change

- Definition: 1x
- Development: 1.5-6x
- After release: 60-100x

Software is Complex

▲ Applications Gridlock
  ● 1980s Spaghetti code became the 1990s spaghetti integration
▲ Total replacement represents enormous business risk
▲ Increasing costs for maintaining and simply reconfiguring applications
Software Applications

▲ Business Applications
▲ Real-Time Systems
▲ Engineering/Scientific Applications
▲ Embedded Systems
▲ AI/Knowledge-Based Systems
▲ Web applications
▲ PC/Midrange/Mainframe Applications
▲ System Software
▲ ...

Software “Schizophrenia”

▲ Who is the Audience for Software?
  ● The Computer
  ● Subsequent software engineers
  ● Users

▲ Software is developed or engineered, NOT manufactured
  ● Effort in creation not production
Software is an Important Asset

▲Provides valuable service/resource
  ● Important part of most systems
  ● Embodies important domain and system knowledge
  ● Requires maintenance to remain serviceable

▲Costly to develop
  ● Redevelopment must be amortized
  ● Stern-wave of development often brings with it a large bow-wave of maintenance costs

Software Crisis on the Horizon?

U.S. Software Development Productivity Trends

Source: 1989-2000 Worldwide Benchmark
Chapter 2
The Process

Purpose

▲ The purpose of this chapter is to introduce several software process models used to manage large-scale software projects
  ● While no software process works well for every project, every project needs to conform to some systematic process
▲ Objectives
  ● Outline software as a process
  ● Outline key aspects of software process
  ● Examine how to assess software process maturity
Business Agility is Driving Software

▲ “IT Agility” needed for businesses to rapidly exploit new markets and adapt to ever changing business conditions

▲ Change tolerance essential for evolving software at the pace of today’s business

▲ Reduce time to:
  ● Market for new products and services
  ● Operations for internal software systems
  ● Financial break-even point

▲ Software is the cornerstone for change

Software Still Stuck in Construction

▲ Software Development evolved largely out of constructive engineering principles

▲ However, software was meant to change (or else it would be in the hardware!)

▲ “Change” is what software does not do readily unless the principles are intentionally applied
A Layered Technology

Software Engineering

- tools
- methods
- process model
- a “quality” focus

A Common Process Framework

Common Process Framework

- Framework activities
  - work tasks
  - work products
  - milestones & deliverables
  - QA checkpoints
  - Umbrella Activities
Umbrella Activities  
(AKA Cross-Life-Cycle Activities)

- Software project management
- Formal technical reviews
- Software quality assurance
- Software configuration management
- Document preparation and production
- Reusability management
- Measurement
- Risk management

Process as Problem Solving
The Process Model: Adaptability

▲ Framework activities will **always** be applied on **every** project ... BUT
▲ Tasks (and degree of rigor) for each activity will vary based on:
  ● the type of project (an “entry point” to the model)
  ● characteristics of the project
  ● common sense judgment; concurrence of the project team

Primary Goal:
High Quality

Remember:

*High quality = project timeliness*

Why?

*Less rework!*
The Linear Model

System/information engineering

- analysis
- design
- code
- test

Iterative Models

listen to customer

build/revise mock-up

customer test-drives mock-up

Prototyping

RAD
The Incremental Model

increment 1
analysis design code test delivery of 1st increment
increment 2 analysis design code test delivery of 2nd increment
increment 3 analysis design code test delivery of 3rd increment
increment 4 analysis design code test delivery of 4th increment
calendar time

An Evolutionary (Spiral) Model

Planning
Risk Analysis
Customer Communication
Engineering
Customer Evaluation
Construction & Release
Still Other Process Models

- **Component assembly model**—the process to apply when reuse is a development objective
- **Concurrent process model**—recognizes that different parts of the project will be at different places in the process
- **Formal methods**—the process to apply when a mathematical specification is to be developed
- **Cleanroom software engineering**—emphasizes error detection before testing

SEI Process Program Background

- CBA IPIs and SPAs conducted since 1987 through December 1999 and returned to the SEI by January 2000
  - 1512 assessments
    - 1024 CBA IPIs
    - 488 SPAs
  - 1166 organizations
  - 309 participating companies
  - 283 reassessed organizations
  - 6168 projects

*Source: Software Engineering Institute*
## SEI’s Software Process Capability Maturity Model

<table>
<thead>
<tr>
<th>Level</th>
<th>Focus</th>
<th>Key Process Areas</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>Optimizing</td>
<td>Defect prevention, Technology innovation, Process change management</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Managed</td>
<td>Process measurement and analysis, Quality management</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Defined</td>
<td>Organization process focus, Organization process definition, Peer reviews, Training program, Intergroup coordination, Software product engineering, Integrated software management</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Repeatable</td>
<td>Software project planning, Software project tracking, Software subcontract mgt., Software quality assurance, Software configuration mgt., Requirements management</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>Initial</td>
<td>Heroes</td>
<td></td>
</tr>
</tbody>
</table>

*Source: Software Engineering Institute*

## Summary of the SEI/CMM Levels

1) **Initial**: The software process is characterized as ad hoc, and occasionally even chaotic. Few processes are defined, and success depends on individual effort.

2) **Repeatable**: Basic project management processes are established to track cost, schedule, and functionality. The necessary process discipline is in place to repeat earlier successes on projects with similar applications.

3) **Defined**: The software process for both management and engineering activities is documented, standardized, and integrated into a standard software process for the organization. All projects use an approved, tailored version of the organization’s standard software process for developing and maintaining software.

4) **Managed**: Detailed measures of the software process and product quality are collected. Both the software process and products are quantitatively understood and controlled.

5) **Optimizing**: Continuous process improvement is enabled by quantitative feedback from the process and from piloting innovative ideas and technologies.
Summary Maturity Findings: Organizational Trends

- **Software Quality Assurance** is the least frequently satisfied level 2 KPAs among organizations assessed at level 1.
- **Integrated Software Management, Training Program and Organization Process Definition** are the least frequently satisfied level 3 KPAs among organizations assessed at level 2.
- **Higher maturity** has been reached among those organizations reporting reassessments.

*Source: Software Engineering Institute*
Process Improvement Maturity Levels

Defined
▲ Process metrics focus
▲ High predictability
▲ Process => Success
▲ Medium Risk

Repeatable
▲ Project metrics focus
▲ Low variability/Medium predictability
▲ PM => Success
▲ Medium-High Risk
Ad hoc
▲ High variability/Low predictability
▲ Heroes => Success
▲ High Risk

More Traction at Upper levels...

Optimized
▲ Value metrics
▲ High predictability
▲ Agility => Success
▲ Low Risk

Managed
▲ Product and Process metrics
▲ High predictability
▲ Managed Process => Success
▲ Low-Medium Risk
Survey of Background and Interest

▲ Name  ▲ Why did you take CS5704?
▲ Student ID  ▲ What are your objectives for this course?
▲ Address  ▲ Which of the following have you created?
▲ Phone #  ● Requirements
▲ Email  ● Specification
▲ Program (CS, MIS, MIT)  ● Architecture Design
   ● Number of Hours  ● Detailed Program Design
   Completed in Program  ● Source Code
▲ Programming/Software  ● Test Plans
   Engineering Experience  ● Test Cases
   ● # of years  ● Maintenance Plan
   ● Current position  ● Project Plan
▲ Languages/Environments  ● Business Case for Project
▲ Languages/Environments  ● Software Methodology

Homework Assignment for 1/26/01

▲ Read Pressman Chapters
   ● 1.2
   ● 2.1-2.7
   ● 3.1-3.8
   ● 4.1-4.3
▲ Email survey information to:
   bohner@nvc.cs.vt.edu
▲ Have a great week!