Senior Design Lab -- PACCAR 106

Each team has space within the lab …

- In general, two computers (monitors) per team
- Spaces are labelled (CPSC 1, CPSC 2, etc.)
- Plus shared workspace and a printer

To check-out equipment …

- See Jason Schnagl (HERAK 103E)
- Computers and additional equipment for checkout

Please see the lab rules, ask if you have questions …

- By using the lab, you are agreeing to the rules
Successful vs Challenged vs Failed

A **successful project**
- develops the “correct” product **on time** and **within budget**

Two notions of “correctness” in software engineering
1. **Verification**: was it built correctly? (to specification)
2. **Validation**: was it the correct thing to build? (to users needs)

A **challenged project**
- late, over budget, or partially incorrect (partially useful)

A **failed project**
- canceled or not adopted (not useful)

How well do Software Projects do?

<table>
<thead>
<tr>
<th>MODERN RESOLUTION FOR ALL PROJECTS</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
</tr>
<tr>
<td><strong>SUCCESSFUL</strong></td>
</tr>
<tr>
<td><strong>CHALLENGED</strong></td>
</tr>
<tr>
<td><strong>FAILED</strong></td>
</tr>
<tr>
<td>2011</td>
</tr>
<tr>
<td>29%</td>
</tr>
<tr>
<td>49%</td>
</tr>
<tr>
<td>22%</td>
</tr>
</tbody>
</table>

The Modern Resolution (OnTime, OnBudget, with a satisfactory result) of all software projects from FY2011–2015 within the new CHAOS database. Please note that for the rest of this report CHAOS Resolution will refer to the Modern Resolution definition not the Traditional Resolution definition.

* Standish Group, 2015
How well do Software Projects do?

Leading project **success factors** (most important to less important)

1. Executive management support
2. User involvement
3. Optimization (of scope ... limit to high value features)
4. Skilled resources
5. Project management expertise
6. Agile process
7. Clear business objectives
8. Emotional maturity (bad: over-ambition, prestige, absence, ...)
9. Execution (planning, measuring progress, managing change)
10. Tools and infrastructure (limit unnecessary overhead)

The Quality Triangle

**Scope**: amount of work team has to do
- e.g., the features that need to be implemented

**Time**: how much time you have
- may be different than how much time you need!

**Budget**: amount of money you have to spend
- in software, typically translates to the size of the team
The Quality Triangle

Related to the saying “fast, cheap, good; pick 2”

For your project:
- You have fixed cost and time ... so high quality depends on scope
- You also need a useful product ... so find & focus on the MVP!
- Also need to develop your skills (technical & project management)

Development Processes (aka Dev. Lifecycles)

Define development steps/phases and order to carry them out

The main steps (phases) found in most process models:
- Requirements ... what to build
- Architecture & Design ... technologies, components, detail
- Implementation ... primarily coding
- Testing ... unit, system, integration, user
- Deployment ... installation, training
- Maintenance ... updates, bug fixes, upgrades
Steps

1. Start with a general idea of what to build
2. Code like hell
3. Maybe use some combination of (optional)
   ○ informal design, debug/test methods
4. Stop when you have a product ready to release