Presentation History

- Prepared and first presented at Korea Polytechnic University
- Missing bullet detailing Quality² added to "Balance" slide

Predictable Software Development

By Matthew Von-Maszewski

November 9th, 2001

Software Development Problem in Industry

- Time usage in Industry
  - 1/3 Planning
  - 1/6 Coding
  - 1/4 Component Test [Unit Test]
  - 1/4 System Test

[from The Mythical Man-Month, by Frederic P. Brooks, Jr.]

- ~100% Time in College spent on Coding

College Coding Focuses on Function not Form

- Test Question: Which "if" statement is best?

```
void function(char * value)
{
        if ( *value ) {/* some action */};
        if ( *value != 0 ) {/* some action */};
        if ( 0 != *value ) {/* some action */};
        if ( '\0' != *value ) {/* some action */};
        return;
}       /* function */
```

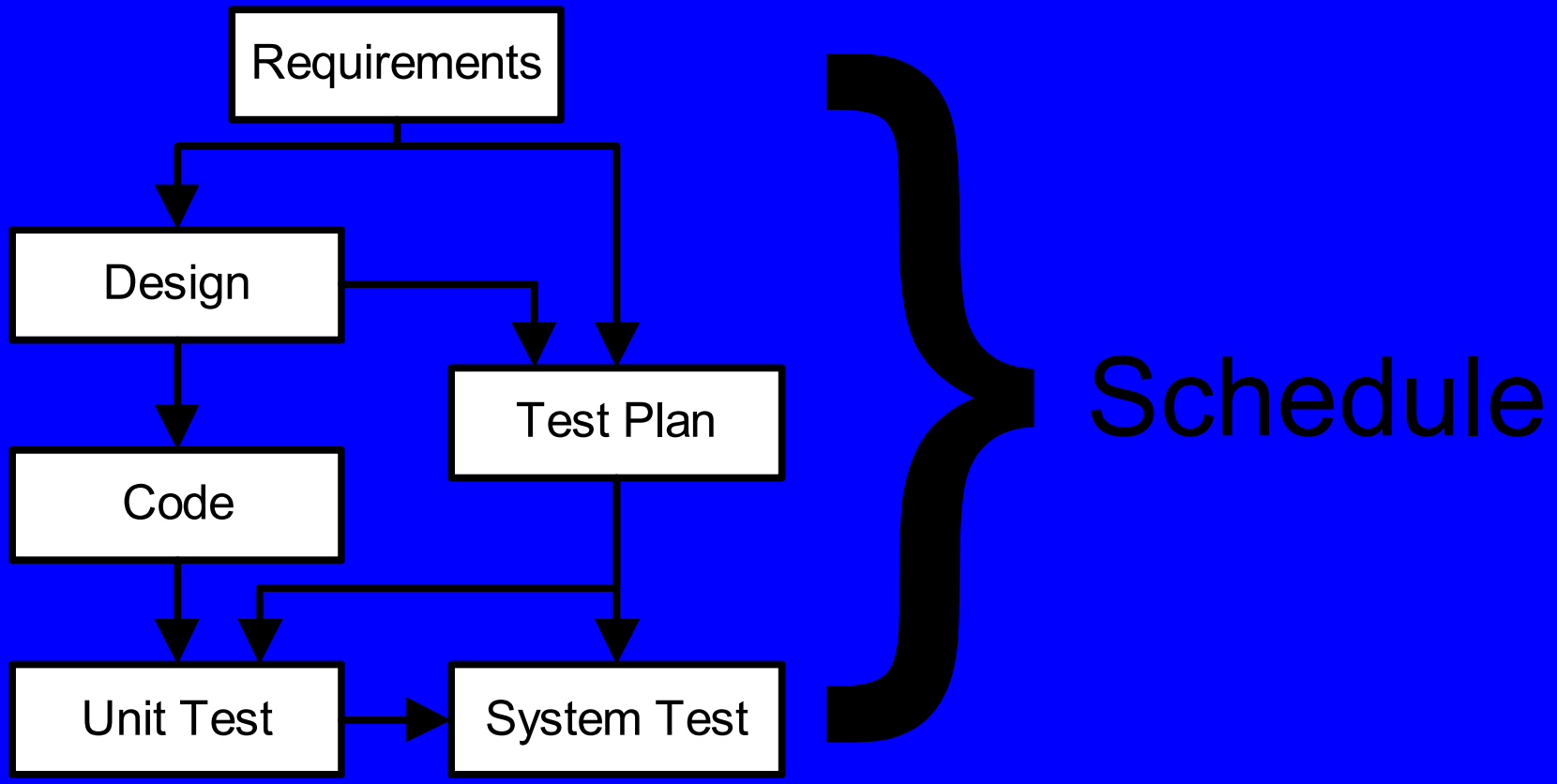
You can change this problem by grasping four Industry concepts

- Software Products have many releases
- Each product release has a plan
- A product plan has balance
- You have responsibilities

Software Products have many Releases

- Fewer features achieve earlier release
  - More customers sooner
  - More revenue sooner
- Customers explain better features for next release
  - Each release adds more features
  - Each release adds more customers
- Never delay a release for an extra feature

Each Product Release has a Plan



A Product Plan has Balance

$$\text{Schedule} = \frac{\text{Features} \times \text{Quality}^2}{\text{Staff}}$$

- This formula is not "mathematical", it is a rule of experience
- Quality^2 means the quality of this product plan and the previous product plan impact this product's schedule
- Staff can only be increased over the long term
- Features, Quality, and Schedule can be changed over the short term
- But try to never change schedule

You have Responsibilities (1)

- To the Product and its future releases:
  - Always remember that there is another release
  - Listen to the current customers to find the future customers
- To the plan:
  - Know all parts of the plan
  - Contribute actively to other parts, not just coding
- To the balance
  - Understand that Schedule is paramount
  - Help your team balance Features and Quality

You have responsibilities (2)

- To yourself: After college, continue your education
  - Goal: read at least one technical book per quarter
    - Design Patterns, by Gamma, Helm, Johnson, and Vlissides
    - Effective C++, by Scott Meyers
  - Goal: learn about coding as an art form
    - Writing Solid Code, by Steve Maguire
    - The Practice of Programming, by Kernighan and Pike
  - Goal: find, practice, and share good software development techniques
    - The Capability Maturity Model, from Software Engineering Institute (Carnegie Mellon University)
    - Software Testing in the Real World, by Edward Kit