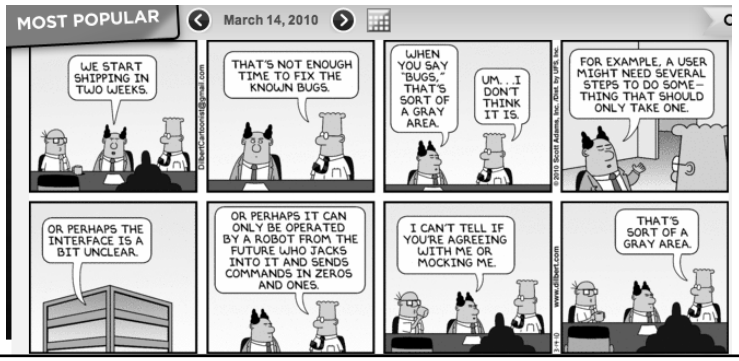
Quality Assurance I

Project presentations

QA Basics

Need for a plan



1

Project 1 Presentations

- Project 1 presentations
  - Will be limited to 8 minutes apiece (practice your timing)
  - Make sure you can connect to the projector (in advance)
  - Test your demo on the computer you plan to use
- Not enough time for all the teams to present during Friday class
  - Random one or two teams will present Monday

Project Submission

- All Project 1 materials are due at class time on Friday
- Make sure that **all project deliverables** are available on your Assembla pages with links from the Home page
  - Include source code as a downloadable package
  - Include any executable and test cases
  - Include presentation slides
  - Provide explicit instructions how to download, install and run your software!

Steps to Academic Integrity

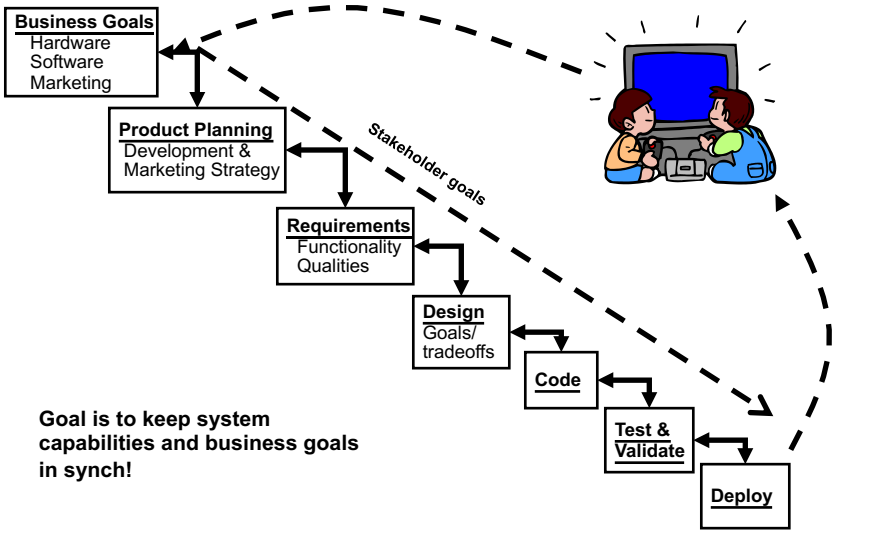
- Reminder: unattributed use of material you did not produce is plagiarism
- Basic steps to ensure safety
  - Any work from another source must contain a reference to that source
  - It must be clear what is and is not original work
  - Any submission must be “substantially” original work (i.e., think 90%)
- Areas to be careful
  - OK to use prior work as a *model* but not copy the work itself
  - OK to include non-original code if a) it is clearly marked and b) most is by your own effort

QUALITY ASSURANCE BASICS

Purpose of SE

- The *purpose of Software Engineering* is to *gain* and *maintain* intellectual and managerial control over the products and processes of software development.
  - **Intellectual control**: able to make rational development decisions based on an understanding of the downstream effects of those choices.
  - **Managerial control** means we likewise control development *resources* (budget, schedule, personnel).

Product Development Cycle

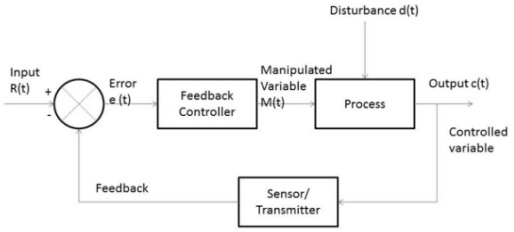


Goal is to keep system capabilities and business goals in synch!

Requires Feedback-Control

- Uncertainty means we cannot get everything under control then run on autopilot
- Rather control requires continuous feedback
  1. Define ideal
  2. Make a step
  3. Measure deviation from ideal
  4. Correct direction or redefine ideal and go back to 2

Role of QA



Basic QA Questions

- For this to work, must define notions like "ideal" and "measure" for products and processes
  - What defines the "ideal?"
  - What should we measure?
  - How can we measure it?
  - When should we measure it?
  - Who should do the work?

Example: System Requirements

- What happens if we get requirements wrong?
- What qualities should a "good" requirements specification have (ideally)?
- How should we evaluate the qualities of the requirements specification?
- What is the right time for these activities?
- Which roles should be responsible?

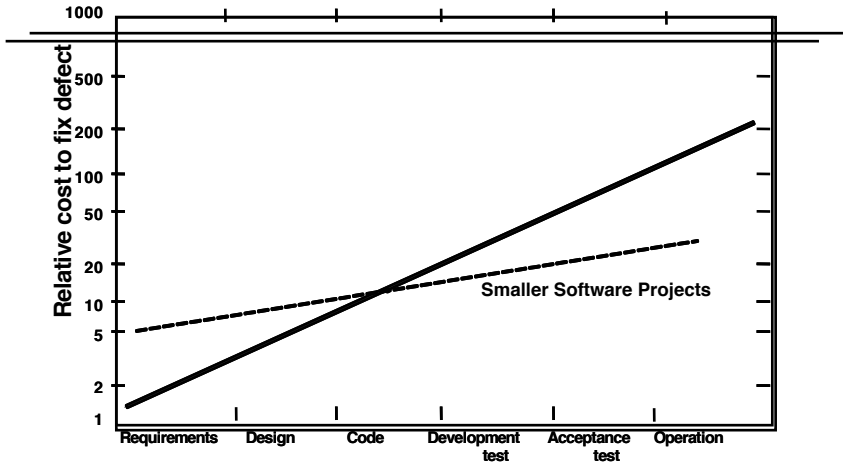
QA Questions

- Properties of a good requirements spec
  - Relevant: capture what the stakeholders want?
  - Complete: capture all the stakeholder requirements (functional and quality)?
  - Consistent: not inconsistent with one another?
  - Unambiguous: avoid multiple interpretations?
  - Precise: clearly distinguish acceptable from unacceptable implementations?
  - Verifiable: can it be tested?
- How could we evaluate these properties?
  - What could we actually *measure*?
  - Hard problem

Example: System Requirements

- What happens if we get requirements wrong?
- Ideal: which qualities should a “good” requirements specification have?
- How should we evaluate the qualities of the requirements specification?
- *When is the right time for these activities?*
- Which roles should be responsible?

Increase in Software Cost-to-fix vs. Phase (1976) *



* Barry Boehm - A View of 20th and 21st Century Software Engineering

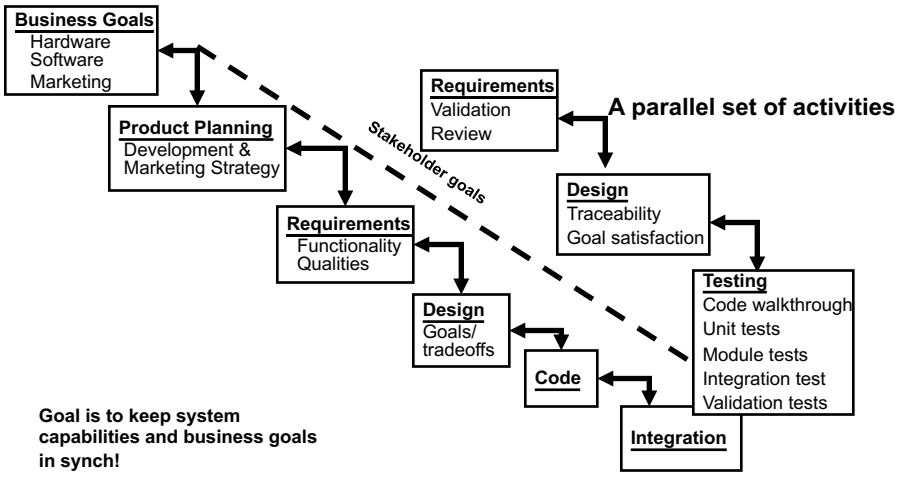
Quality is Cumulative

Requirements Analysis	• Are the requirements valid? • Complete? Consistent? Implementable? • Testable?
Architectural Design	• Does the design satisfy requirements? • Are all functional capabilities included? • Are qualities addressed (performance, maintainability, usability, etc.?
Detailed Design	• Do the modules work together to implement all the functionality? • Are likely changes encapsulated? • Is every module well defined
Coding	• Implement the required functionality? • Race conditions? Memory leaks? Buffer overflow?

We need a plan!

- QA activities are
  - Critical to control (and project success)
  - Part of every phase of the project
  - Time consuming, labor intensive and expensive
    - NIST Study: *~80% of development costs are consumed by software developers identifying and correcting defects*
  - Cannot do everything, need to choose
- Suggests need to plan QA activities to:
  - Detect issues as early as possible
  - Target highest priority/risk issues for project
  - Support cost-effective use of resources

Product Development Cycle



QA Plan

- Purpose: synchronize QA activities with project deliverables such that:
  - Artifacts satisfy quality goals
  - Delivered code is consistent with stakeholder needs
- The plan should answer the question “How will the project will meet its quality goals?”
  - The overall QA objectives, strategy, and methodologies
  - The kinds of QA activities that should occur
  - Roles that will carry out the activities
  - When the activities should occur

Example QA Plan

- See example provided with Assembla pages

1. Purpose
2. Methods
   1. Prototypes
   2. Reviews
   3. Testing, etc.
3. Schedule and Resources
4. Measures: metrics collected
5. Acceptance criteria
   1. Review issues
   2. Code defects
   3. Quality variation (e.g., performance variation), etc.
6. Responsibilities

Verification and Validation

Validation and Verification

- *Validation*: activities to answer the question – "Are we building a system the customer wants?"
  - Familiar activity: customer review of prototype
- *Verification*: activities to answer the question – "Are we building the system consistent with its specifications?"
  - Most familiar verification activity is functional testing
- Both are processes, both have many variations

V&V Methods

- Most applied V&V uses one of two methods
- Review: use of human skills to find defects
  - Pro: applies human understanding, skills. Good for detecting logical errors, problem misunderstanding
  - Con: poor at detecting inconsistent assumptions, details of consistency, completeness. Labor intensive
- Testing: use of machine execution
  - Pro: can be automated, repeated. Good at detecting detail errors, checking assumptions
  - Con: cannot establish correctness or quality
- Will discuss methods for each of these in coming weeks

Summary

- Quality Assurance activities provide the *feedback* in controlling development
- Effective QA requires that we
  - Can define what we want (the ideal)
  - Can evaluate work products against the ideal
- QA activities consume substantial resources, require planning

  …But, done well, pay for themselves