Code Quality

Sang Shin
www.jPassion.com
"Learn with jPassion!"

Topics

- Characteristics of quality code
- Software defects
- How to improve code quality
- Code quality checking
- Coding principles
- Commenting

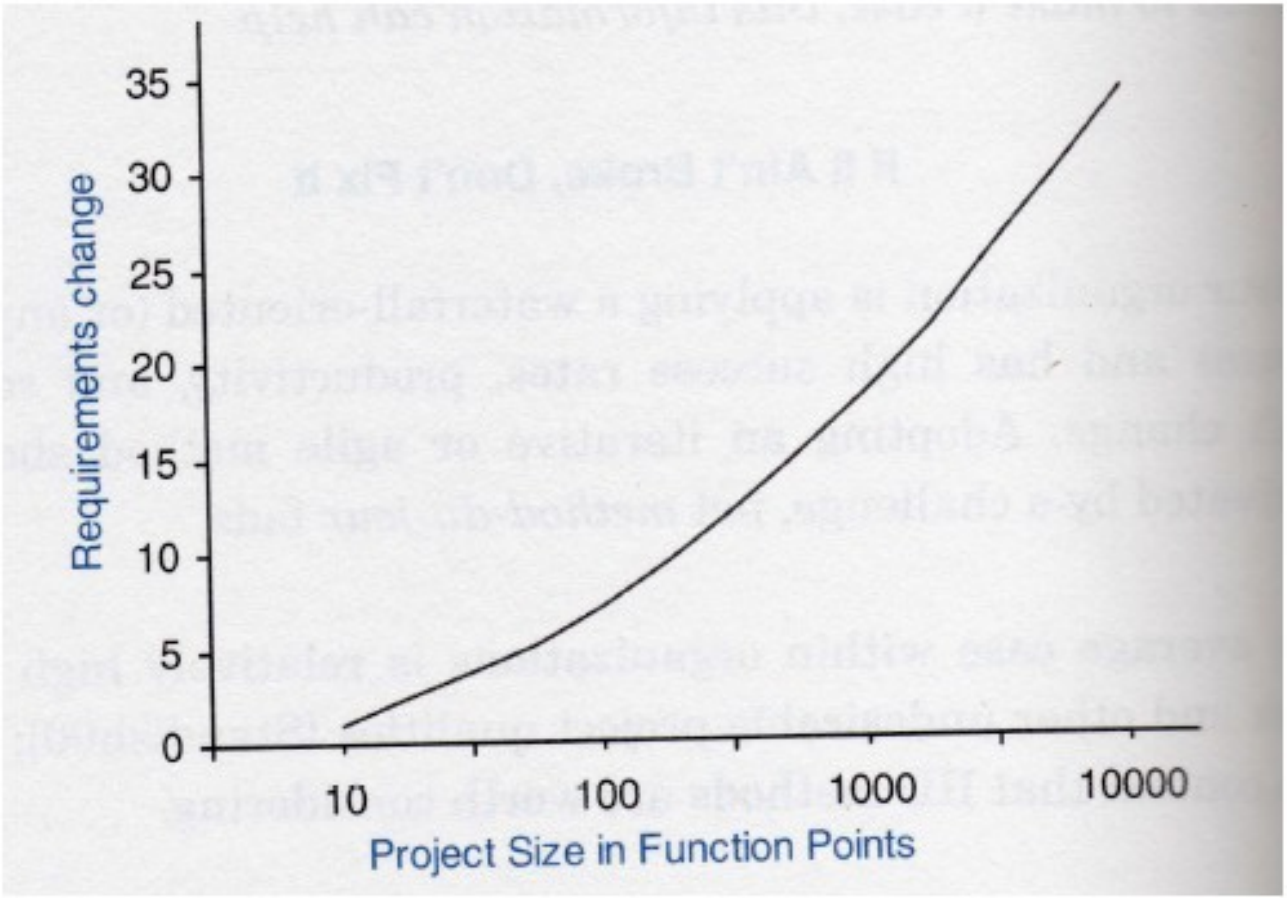
Why we should care Code Quality?

Something to think about

“Programs must be written for people to read, and only incidentally for machines to execute”

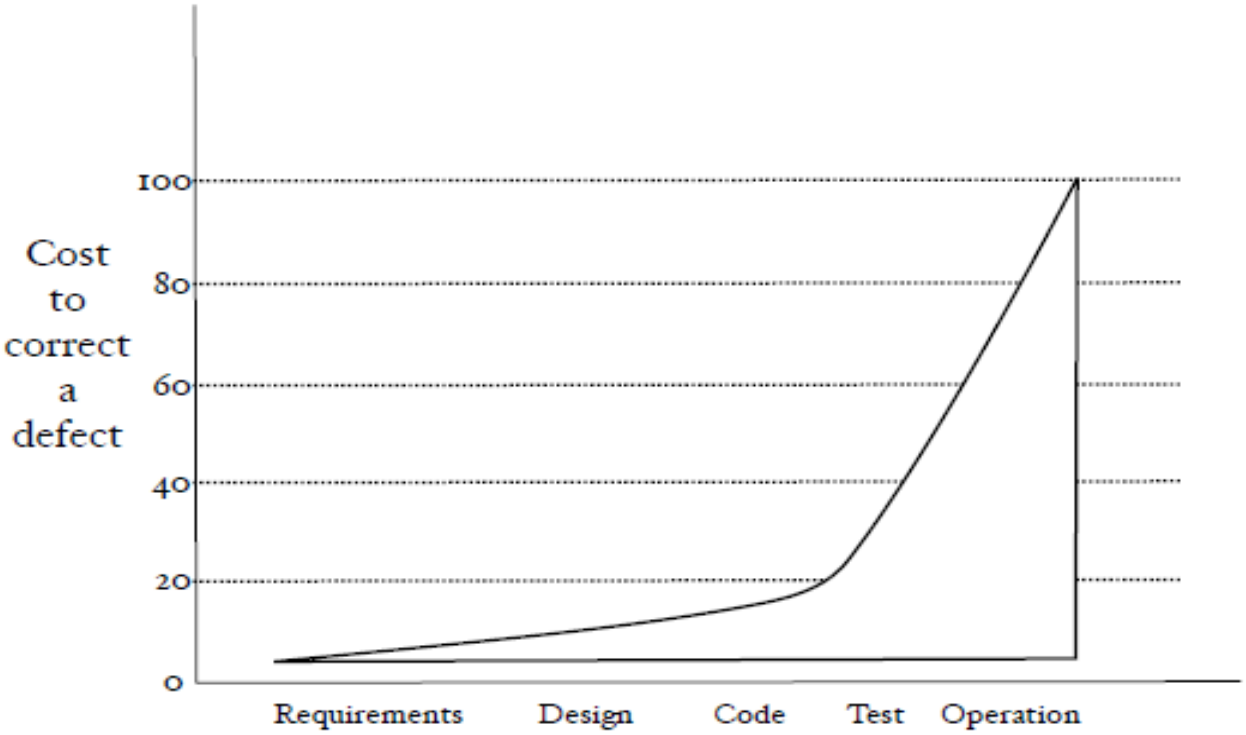
- Abelson and Sussman

Change in Requirements



Cost of Defect

- The earlier the better for dealing with defects



[BOEH01]

Software Defects

Software Defect Statistics

- Finding, fixing problems in production is 100 times more expensive than during requirements/design phase
- 40-50% of typical project work is “avoidable rework”
- ~80% of avoidable rework comes from 20% of defects
- ~80% of defects come from 20% of modules
- ~90% of downtime comes from at most 10% of defects
- Peer reviews catch 60% of defects
- Perspective-based reviews catch 35% more defects than non-directed reviews (Perspective-based reading is a reading technique that gives each reader a specific viewpoint to use, such as "user" or "tester.")
- Disciplined personal practices can reduce defects by up to 75%

But, still...

- The evidence is overwhelming, but still..
- We never seem to have time to do it, but always seem find time to “redo” it??

What about QA?

- Can't QA take care of quality? Why should developers care?
  - Because QA should not care about quality of design and implementation
  - Instead they should care about “testings that matter”: functionality, acceptance, performance, usability, etc.
  - Give them a better quality software so they can really focus on “testings that matter”

Technical Debt

- “Technical debt” are activities (like refactoring, upgrading a library, conforming to coding standart, ..) that you've left undone
- “Technical debt” will hamper your progress if left undone

How to Improve Code (Software) Quality?

Ways to Improve Software Quality

- Start early
- Don't compromise
- Schedule time to lower your technical debt
- Make it work; make it right (right away)
- Requires monitoring and changing behavior
- Be willing to help and be helped

Individual Effort

- Care about quality of your code
  - Good names for variables, methods,
  - Short method, smaller classes,
- Keep it simple
- Write tests with high coverage
- Run all your tests before check-in
- Check in frequently
- Ask for feedbacks

Team efforts

- Avoid shortcuts
- Talk collective ownership – team should own the code
- Promote positive interaction
- Provide constructive feedback
- Constant code review

Broken Window Problem

- Study shows broken windows lead to vandalism
- Code that no one cares for deteriorates quickly
- Fix code that is not elegant or looks broken
- Keep your code always releasable

Treat Warnings as Errors

- Don't say “that's only a warning”
- Warnings may have hidden problems
- If unavoidable, suppress (selectively)

Strive for Higher Cohesion

- Cohesive code is focused, narrow, and small
  - It does one thing and one thing well
  - Single responsibility principle
- Higher cohesion leads to “Lower Cyclomatic Complexity” (Cyclomatic complexity/conditional complexity indicates complexity of a program by providing measurement on the number of linearly independent paths through a program's source code)
  - Less expensive to maintain
  - Change in low cohesive code will break something else
- Cohesiveness applies to all levels
  - method, class, component, and subsystem

Code Quality Checking

Code Coverage

- How much (%) of your code is covered by test?
- How about paths through your code?
- Is there code that deserve not to be tested?
- Code coverage tools can tell you which and how much code is covered
  - Clover
  - Cobertura
  - JaCoCo
- Some tools delete code that have no test
  - Guantanamo
  - Ashcroft

Cyclomatic Complexity

- Cyclomatic Complexity Number (CCN)
  - Number of distinct paths through code
- Strive for lower CCN
- Tools can show CCN
  - PMD
  - CheckStyle

Code Duplication

- Duplicated code is expensive to maintain
  - Duplicated code is common
- Tools that detect code duplication
  - PMD

Code Analysis

- Analyzing code to find bugs
  - Logic errors
  - Coding guideline violations
  - Synchronization problems
  - ..
- Tools
  - PMD
  - FindBugs
  - JLint

Coding Principles

Coding Principles

- Simplicity
- Clarity
- Brevity
- Humanity

Commenting Code

Commenting Code

- Comments should say “Why” or “Purpose” not “How”
- Don't comment what a code does - I can read the code
  - Keep it DRY
- Don't keep documentation separate from code

Thank you!

Check JavaPassion.com Codecamps!
http://www.javapassion.com/codecamps
"Learn with Passion!"