CompSci 230
Software Design and Construction

Software Quality 2015S1

What is software quality?

Introduction to your lecturer

- Diana Kirk

  - 1987-1994: C programmer - Minder Systems and ECONZ
  - 1994-1998: Shareholder-Director - Foley International
  - 1998-2003: Software Quality Manager – Dialogic, later Intel NZ
  - 2003-2006: PhD Computer Science, The University of Auckland
  - 2006-2008: Post-doc – Research Fellow at Griffith University
  - 2008-2010: Consultant (software process and practice)
  - 2008- : Contract research and teaching positions (UoA, AUT, Unitec, Massey)

Lecture plan

Week 1: *No class - Anzac Day*
What is software quality?
Some key developer practices (version control, testing).

Week 2: Black box testing.
White-box testing.
Myers' testing principles.

Week 3: Traditional approach to testing (Waterfall).
Agile approach to testing (XP).
Famous failures.

Readings

- **Myers**, Glenford. *The Art of Software Testing*, 3rd edition, Wiley, 2012.
  - This is an e-book, on an unlimited-user viewing licence in our library. This means that 3 people can access at any one time.
  - The number of pages you can download at any one time is limited.
  - You *may* have to be on-campus to read this online.

- **Vogel, Lars.** JUnit – Tutorial, version 2.5, sections 1, 3, 4 and 5.
  - Please do this before Friday.

- Supplementary reading :
  - If you're using Eclipse, you should complete Vogel's tutorial (Sections 3.4 through 6) before attempting Assignment 3.
  - If you're using DrJava, read Chapter 8 (Testing Using Junit) in the DrJava User Documentation.

Learning goals

- Demonstrate a theoretical understanding of software quality
  - Quality models and characteristics
  - Quality themes, origins and application to software processes
- Demonstrate an understanding of testing
  - theoretical (black box, white box, unit test, component test, system test, acceptance test, …)
  - practical (competently perform some testing tasks using JUnit)
- Demonstrate a basic understanding of how traditional and agile methodologies approach software testing
- Develop an understanding of version control
  - theoretical (best practices)
  - practical (tutorial and assignment)

Learning goals for today

- Understand what we mean by software quality.
- Relevance of
  - quality models
  - process

Questions

- **What do we mean by 'software quality'?**
- How can we achieve 'software quality'?

What is software quality?

- Some ideas:
  - Does it do what the specification states?
    - The idea of 'conformance to specification' was introduced in the field of manufacturing (widgets must be 4mm +- .1mm)
    - Directly measurable (establish by inspecting or testing) BUT
    - User may expect this as a minimum and view a quality product as one that offers more
      - e.g. word processor may have all expected functions but may be difficult to use

What is software quality?

- Some ideas:
  - Fitness for use
    - How well does the product perform its intended functions?
    - Varies according to user group (concept of 'grade')
      - In a car, I want fuel-efficiency and safety
      - You may prefer a car that looks good and has powerful acceleration
    - For software, a user may care about, for example, security, usability, reliability, etc.

What is software quality?

- So the answer to 'what is software quality?' is 'it depends'
- Some products to consider
  - Control software for a patient monitoring system
    - reliability? accuracy?
  - Game to support children learning maths
    - usability? child-age appropriate?
  - Mobile app
    - resource usage? usability? developer can easily modify to upgrade frequently?

Quality models

- A number of quality models have been developed that describe the *quality characteristics* to be considered when developing a quality product.

- As a developer, you should understand which of these apply to the product you are developing.

Quality models

- ISO/IEC 25010 – System and software quality models

BS ISO/IEC 25010:2011
ISO/IEC 25010:2011(E)

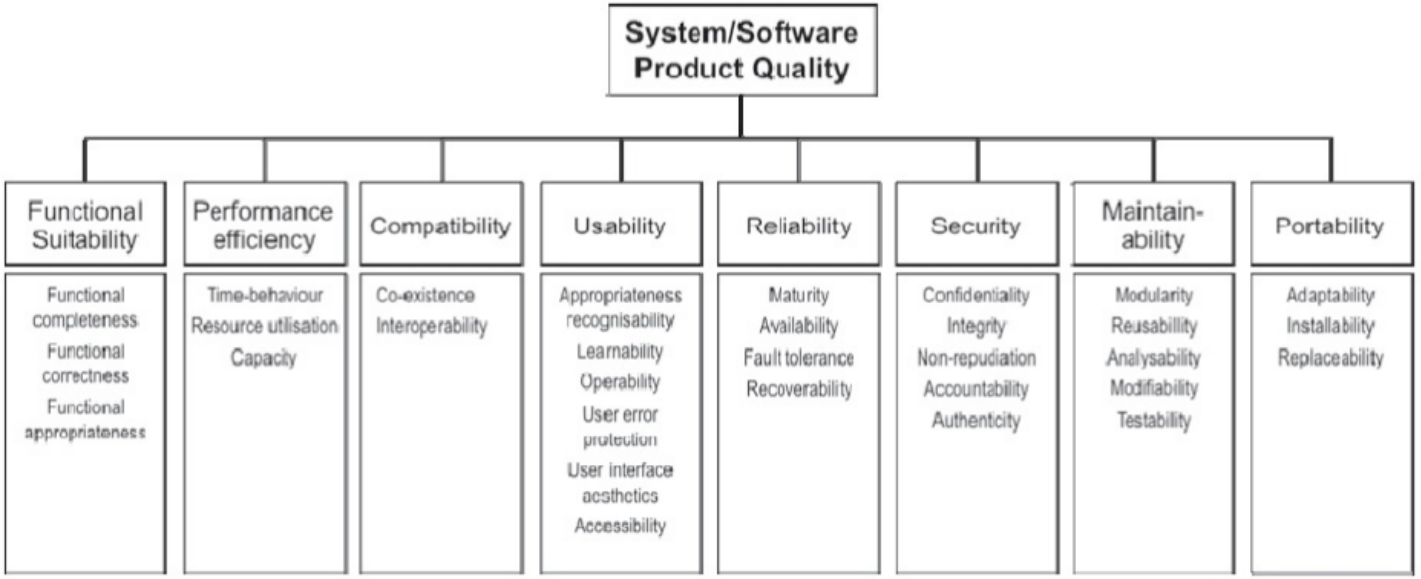


Figure 4 — Product quality model

Questions

- What do we mean by ‘software quality’?
- **How can we achieve ‘software quality’?**

Historical perspective

- First look at where many of the software quality ideas come from.
- Origins are in manufacturing.

- Brief historical look at manufacturing:
  - US
  - Japan

Historical perspective US

- Brief historical look at manufacturing: (US)
  - 13th century: craftsman guilds

  - mid-18th century: factory system emphasising product inspection (conformance to specification)

  - World War 11: quality becomes critical (bullets) - Shewart's statistical process control techniques applied
  - 1950s: post-war economic growth - abandon quality learnings (assumption: quality == $$)

  - 1970s: US industrial sectors broadsided by competition from Japan
  - Late 1900s: Quality systems based on Japanese practices - *TQM, CMM, Six-Sigma, Zero Defects, Lean manufacturing*

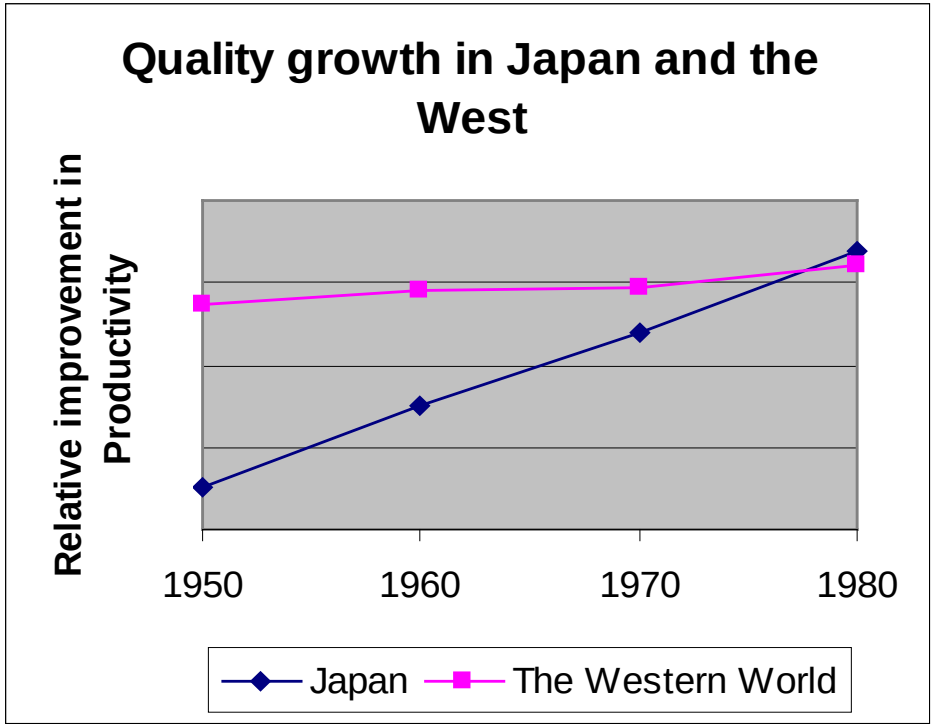
Historical perspective Japan

- Brief historical look at manufacturing: (Japan)
  - 1920-1930s: under-developed industrial technology (scrap high, reliability low)

  - late 1940s:
    - war with US destroys Japanese production capacity
    - Sarasohn (US) posted to Japan to support Japanese communications industry
    - Deming and Juran brought from US to train Japanese managers
  - 1950 - ??? Japanese embrace and expand quality ideas and make quality production profitable

Historical perspective

- Brief historical look at manufacturing:
  - adapted from Juran 1981



Quality themes

- What were some of the quality ideas that produced such success?
- There were many – I will present those most relevant for our investigation into *software* quality.

Quality themes

- Deming:
  - Quality is conformance to specification
    - *Get the specs right and build according to them. Basis of the 'waterfall' approach to software development.*
  - You cannot inspect quality into a product – it must be built in from the start
    - *Inspecting (testing) finds what is wrong with the product, which then has to be reworked. Too late. For developer, it's your job to make sure what is passed on to testing team or another developer is of high quality.*
  - Most problems are a result of the system (process) – don't blame the workers, it's a management problem
    - *Consider: team of 5 developers, 4 experienced and 1 new. The new one is in charge of keeping the specification wiki up to date. Hmmm…*

Quality themes

- Juran:
  - Quality is fitness for use (issues of 'grade')
    - *Seen in agile approaches to software development i.e. work closely with the client to make sure you deliver what (s)he really wants.*
    - *Developer must understand required quality characteristics.*
  - Workers must be empowered by support of top management
    - *Basis of TQM initiatives.*
    - *HUGE problem in many (most?) software development initiatives.*

http://www.toyota.co.jp/ev/vision/production_system

Quality themes

- Pareto:
  - Only a few factors are responsible for most of the problems
  - *If you want to improve your software process, identify the small number of areas that will make the biggest difference.*

- Toyota:
  - Eliminate waste (inventory, processing steps)
  - *Basis of the lean approach to software development.*

http://www.toyota.co.jp/ev/vision/production_system

Quality themes

- Western initiatives resulting from Japanese successes, all applied to software:
  - TQM (Total Quality Management):
    - Management approach involving customer focus and system-wide continuous improvement based on measurement
  - Six Sigma:
    - Management strategy aimed at reducing process variability i.e. removing 'common causes'
  - Lean:
    - Reduce waste
  - ISO 9000 series (Quality systems):

http://www.isixsigma.com/library/content/c021230a.asp
http://europe.isixsigma.com/library/content/c051214b.asp

Quality themes and SE

- Waterfall
  - quality is conformance to specification
  - inspections and reviews to avoid defect injection (can't test quality in)
  - phase 'gates' (test outputs from each stage before passing to next)
- XP
  - shared vision
  - iterations for customer feedback (quality is 'fitness-for-use')
  - pair programming and test-first to avoid defect injection
  - empower employees
- Lean software development
  - reduce waste

Quality themes and SE

- Underlying belief is : Good process -> quality product
  - This idea is behind all the (heated) discussions about software process. For example, should an organisation implement a waterfall approach? XP? Perhaps Lean Software Development is the way to go?
  - Each of these approaches implements some of the ideas from above.
  - Current thinking is that practices must be adapted to the particular circumstances of the project.

Quality themes and SE

- Good process -> quality product
  - In this course, our focus is software development. There are some basic practices advocated by all methodologies that are relevant within development and crucial for a quality outcome. These should be understood and implemented by the developer.
    - Managing changes to the codebase (version control)
    - Ensuring you deliver high quality code (unit testing)
  - We will talk about these over the next few sessions.