Unit 1. Software Quality: Overview and Basics

1. Introduction
2. Software Quality Attributes
3. Software Quality Engineering
4. Quality Standards

Reading: TB-Chapters 1 (1.1-1.6), 17 (17.1-17.2), 18 (18.1-18.2)

1. Introduction

The Need for Quality Software Products

–2014 (April): Heartbleed vulnerability
- Caused by bug in OpenSSL implementation
- Several agencies impacted, e.g., Revenue Canada

–2013 (Oct): Failed Affordable Care ("Obamacare") site launch
- Difficult to navigate for users
- Frequent crash due to unexpected heavy load
- Poor responsiveness

–1996: Ariane 5 was destroyed 40 seconds after take off due to a crash in a software component controlling the rocket (1996), cost ~$8 billion

Achieving Quality

-Achieving a high level of product or service quality must be the objective of any organization.

-No longer acceptable to deliver poor quality products, and fix deficiencies after they have been shipped to the customer.

-However, software quality is a complex notion:

- Quality product is expected to comply with customer requirements; unfortunately such requirements are often incomplete for software.
- Difficult to precisely specify and check certain quality requirements (e.g., usability, maintainability, reusability etc.)

-Quality management: defines procedures and standards to be used during software development, and checks that these are followed by all engineers.

2. Software Quality Attributes

Notions

***Quality**: the totality of features and characteristics of a product, process or service that bear on its ability to satisfy stated or implied needs.*

***Quality attributes**: the expected quality features and characteristics of a software product.*

- **Quality attributes:**

-*Used early in the development process* to identify user and system quality requirements.

-Each system has specific and unique quality needs, which are a function of the purpose of the application.

-Are assessed using ***quality control*** techniques and, when possible, appropriate software measurements or ***metrics***.

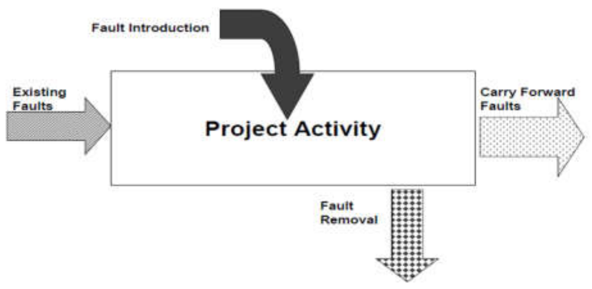
Typical Software Quality Attributes

Lifecycle	Product Transition	Product Operations	Product Revision
Quality Attributes	Portability Interoperability Reusability	Reliability Robustness Efficiency Usability Safety Security Fault-tolerance	Maintainability

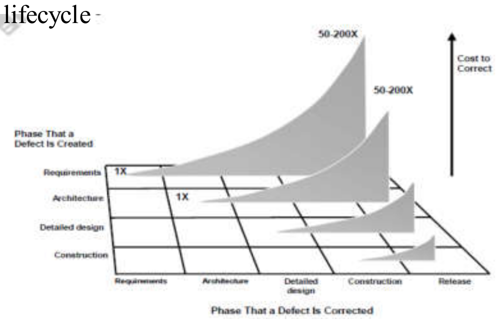
3. Software Quality Engineering

Software Quality Foundation

-Defects are introduced in software by human activities through Software lifecycle



-Quality improvement cost increases as we progress in the software lifecycle



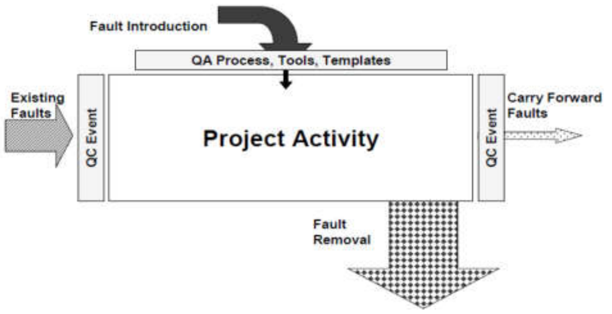
-Project success depends on positioning the project team to detect mistakes **early** so that they can correct them quickly and easily (=> **Quality Control**)

-Project success depends also on using effective and efficient methods by the project team (=> **Quality Assurance**)

QA vs. QC

-**Quality Assurance (QA)** is fault **prevention** through process design and auditing

- **-** Creating processes, procedures, tools, jigs, etc. to prevent faults from occurring
- - Examples: Templates, checklists, guides
- - Main goal: prevent as much as possible defect injection



-**Quality Control (QC)** is fault/failure **detection** through static and/or dynamic testing of artifacts

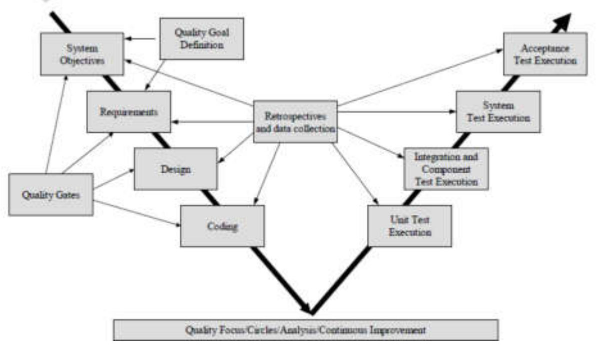
-Examining the artifact against pre-determined criteria to measure conformance

-Quality control methods

- Reviews
Personal, peer, pair, management, QA, independent
-Testing
structural, functional, integration, stress/performance, regression, field, acceptance
-Simulations
Prototypes, models
-Field Trials
Prototypes, beta testing
-Mathematical
Proofs of correctness

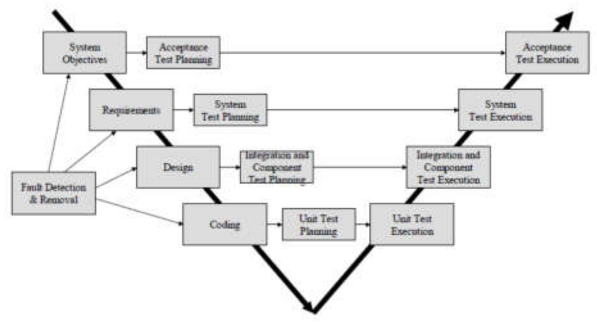
Quality Practices in the Software Lifecycle

QA practices


System Objectives
Quality Goal Definition
Acceptance Test Execution
Requirements
Retrospectives and data collection
System Test Execution
Design
Integration and Component Test Execution
Quality Gates
Coding
Unit Test Execution
Quality Focus/Circles/Analysis/Continuous Improvement

Quality Practices in the Software Lifecycle (ctd.)

QC practices



4. Quality Standards

-Several well-established and proven quality standards and procedures are available.

-Two basic premises underlie existing quality standards:

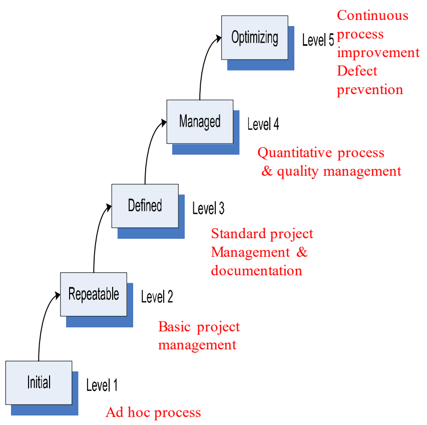
1. Process vs. Product: The quality of a product is largely based on the quality of the process that leads to that product.
2. Conformance vs. Improvement: techniques can be developed to improve the quality of product or process, or alternatively to ensure they conform to specified quality requirements.

Examples of quality standards

	Conformance	**Improvement**
Product	ISO 9126	‘best practices’
Process	ISO 9001 SQA	CMM SPICE Bootstrap

Example of Process Standard: Capability Maturity Model (CMM)

- •Developed by the Software Engineering Institute (SEI) at Carnegie-Mellon University
- •Defines five levels of software maturity for an organization and provides a framework for moving from one level to the next.
- •Guides developers through activities designed to help an organization improve its software process, with the goal of achieving repeatability, controllability, and measurability.
- •Provides a framework for process improvement that consists of "key process areas," or organizational activities that have been found, through experience, to be influential in various aspects of the development process and resultant software quality.


Optimizing
Level 5
Continuous process improvement Defect prevention
Managed
Level 4
Quantitative process & quality management
Defined
Level 3
Standard project Management & documentation
Repeatable
Level 2
Basic project management
Initial
Level 1
Ad hoc process