Software quality is conformance to

1. explicitly stated functional and performance requirements,

2. explicitly documented development standards, and

3. implicit characteristics that are expected of all professionally developed software.

♣ Error – human interaction which produce an incorrect result.

♣ Fault – representation of an error.

♣ Failure – occurs when a fault executes.

♣ What could go wrong?

1. Faulty requirements definition by the client

2. Logical design errors

3. Coding errors

4. Documentation errors

5. User interface and procedure errors

♣ Why did it go wrong?

1. Inadequate knowledge/skills

2. Client-developer communication failures

3. Deliberate deviations from software requirements

4. Non-compliance with documentation and coding instructions

♣ The uniqueness of software products

— High complexity: millions of operation possibilities.

— Invisibility of the product

— Limited opportunities to detect defects during the product development and production process

Requirement document is one of the most important elements for achieving software quality!

Software Quality Factors

Product operation factors

♣ Correctness: output specifications

— output mission, accuracy, completeness of required output

— up-to-date, availability of the information

— standards for coding and documenting the system

♣ Reliability

— minimize failure rate

— failure recovery

♣ Efficiency

— Resources needed to perform software function (processing, data storage, communication, battery consumption, etc.)

♣ Integrity

— software system security, access rights

♣ Usability

— ability to learn, perform required task/training

Product revision factors

♣ Maintainability:

— Deal with the complete range of software maintenance activities: θ corrective maintenance, θ adaptive maintenance and θ perfective maintenance.

— Modularity, simplicity, coding and documentation guidelines, document accessibility, etc.

♣ Flexibility: adaptive and perfective

— degree of adaptability (to new customers, tasks, etc.) ♣ Testability

— support for testing (e.g. log files, automatic diagnostics, etc.)

Product transition factors

♣ Portability

— adaptation to other environments (hardware, software)

— Platform independency

♣ Reusability

— use of software components for other projects

♣ Interoperability

— ability to interface with other components/systems

— compatibility

Software quality assurance is:

— A systematic, planned set of actions necessary to provide adequate confidence that the software development process or the maintenance process of a software system product conforms to established functional technical requirements as well as with the managerial requirements of keeping the schedule and operating within the budgetary confines.

QA Components

♣ Pre-project

♣ Project life cycle activities assessment

♣ Infrastructure error prevention and improvement

♣ Software quality management

♣ Standardization, certification and SQA system assessment

♣ Organizing for SQA - the human components

Assurance vs. control

♣ Quality control: detection/product

— A set of activities designed to evaluate the quality of a developed or manufactured product.

— Main objective: withhold any product that does not qualify.

— Completed before the product is shipped to the client.

♣ Quality assurance: prevention/process

— Main objective: minimize the cost of guaranteeing quality.

— Prevent the cause of errors; detect and correct them early in the development process

— performed throughout the software life cycle

♣ Software CM is a discipline for managing the evolution of software systems throughout all stages of the software life cycle.

“SCM is a service provider in that it supports people and controls data.”

♣ Configuration baseline is a fixed reference configuration established by defining and recording the approved SCI at a milestone event or at a specified time.

♣ Each configuration baseline serves as a point of departure for future SCI changes.

♣ Configuration change control makes sure the product is in a consistent state and enforces access control

— functional configuration audit (FCA)

θ validate the system against the requirement

— physical configuration audit (PCA)

θ whether the design and reference documents represent the software that was built.

♣ Versioning models:

— Lock-Modify-Unlock: SourceSafe, RCS, Subversion

Only one person is allowed to change a file at a time.

— Copy-Modify-Merge: SourceSafe, Subversion, Git

♣ Branching: the duplication of an object under revision control so that modifications can happen in parallel along both branches.