Software Quality

Keunhyuk Yeom
Dept. of Computer Engineering
Pusan National University
yeom@pusan.ac.kr

Software Qualities

- Definitions
  - DoD, 1985, ``The degree to which the attributes of the software enable it to perform its intended end use.''
  - ISO, 1986, ``The totality of features and characteristics of a product or service that bear on its ability to satisfy specified or implied needs.''
  - Kitchenham, 1989, ``Fitness for needs.''
    - Conformance to its specification:
      - Is it a good solution?
    - Fitness for its intended purpose:
      - Does it address the right problem?

Some Insights about Quality

- Quality is not absolute
- Quality is multidimensional
- Quality is subject to constraints (people, money, time, tool)
- Quality is about acceptable compromises
- Quality criteria are not independent

Classification of Software Qualities

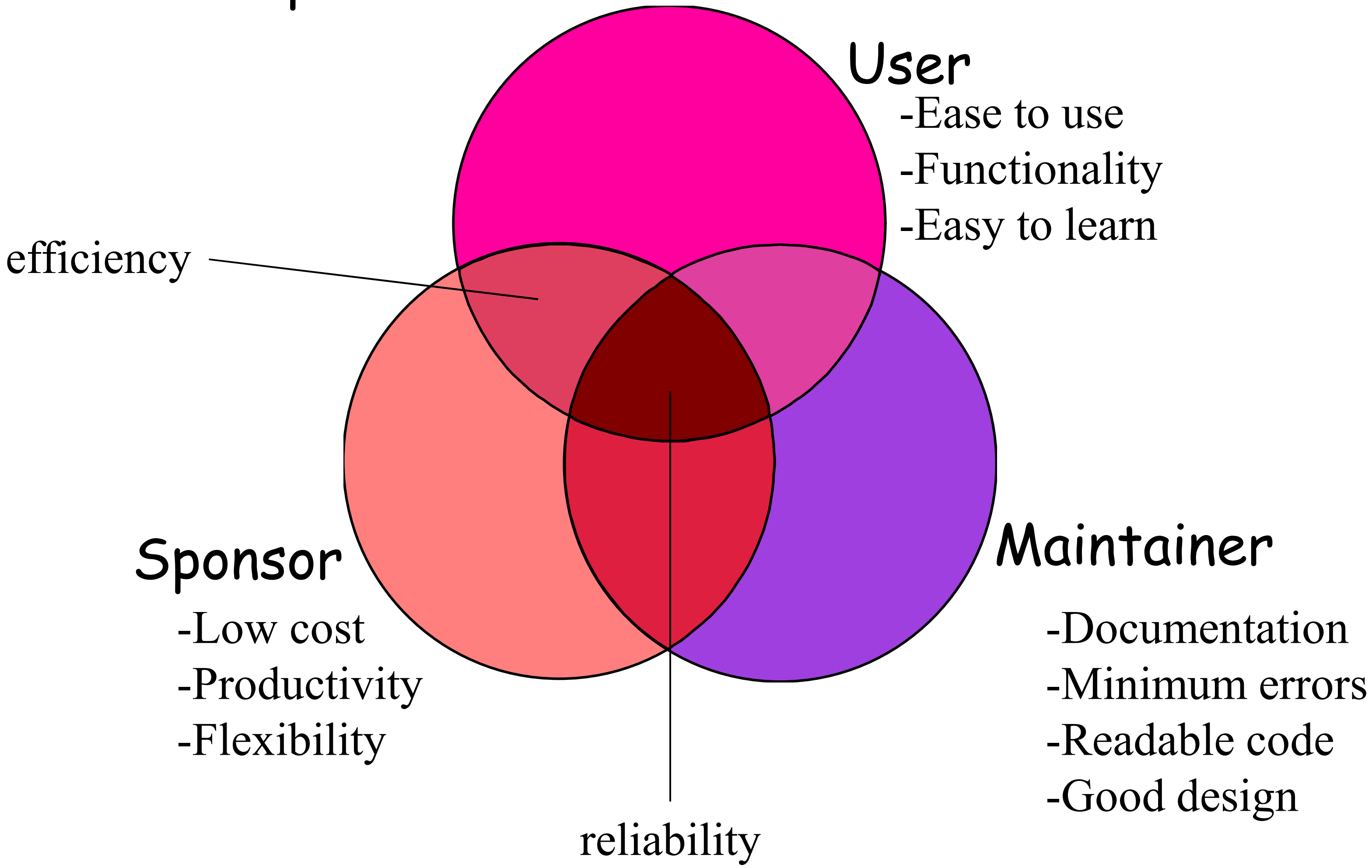
- External Quality vs. Internal Quality
  - Distinction is not sharp
  - External quality: Visible to the users of the system.
  - Internal quality: Concerns the developers of the system.
  - Internal quality help developers achieve the external quality.
    - e.g. verifiability in internal is necessary for achieving reliability in external
- Product vs. Process Qualities

Product and Process Qualities

- Closely related: use a process to produce the software product.
- Product quality:
  - Functionality
  - Usability
  - Efficiency
  - Reliability, etc
- Process quality:
  - Productivity
  - Effectiveness of methods, tools
  - Use of standards
  - Management, etc

Quality Factors

External aspects



User
- -Ease to use
- -Functionality
- -Easy to learn

efficiency

Sponsor
- -Low cost
- -Productivity
- -Flexibility

Maintainer
- -Documentation
- -Minimum errors
- -Readable code
- -Good design

reliability

Why software quality is different from other types of quality?

- Software has no physical existence.
- The lack of knowledge of client needs at the start.
- The change of client needs over time.
- The rapid rate of change in both h/w and s/w.
- The high expectation of customers, particularly w.r.t. adaptability.

Representative Qualities

- Correctness, Reliability and Robustness
- Performance
- User Friendliness
- Verifiability
- Maintainability
- Reusability
- Portability
- Understandability
- Interoperability
- Productivity
- Timeliness
- Visibility

Correctness, Reliability and Robustness

- Open used interchangeably, meaning that the degree of the application's performing its functions as expected.

Correctness

- Correct if the program behaves according to the specification of the functions.
- Assumption :
  - A specification of the system is available.
  - It is possible to determine unambiguously if a program meets the specification

Reliability

- = dependability: reliable if the user can depend on it.
- A measure of the frequency and criticality of product failure
- Failure: An unacceptable effect or behavior under permissible operating conditions.
- Can define in terms of statistical behavior: MTTR, MTTF, MTBF
- Relative, (Q: What about correctness?)

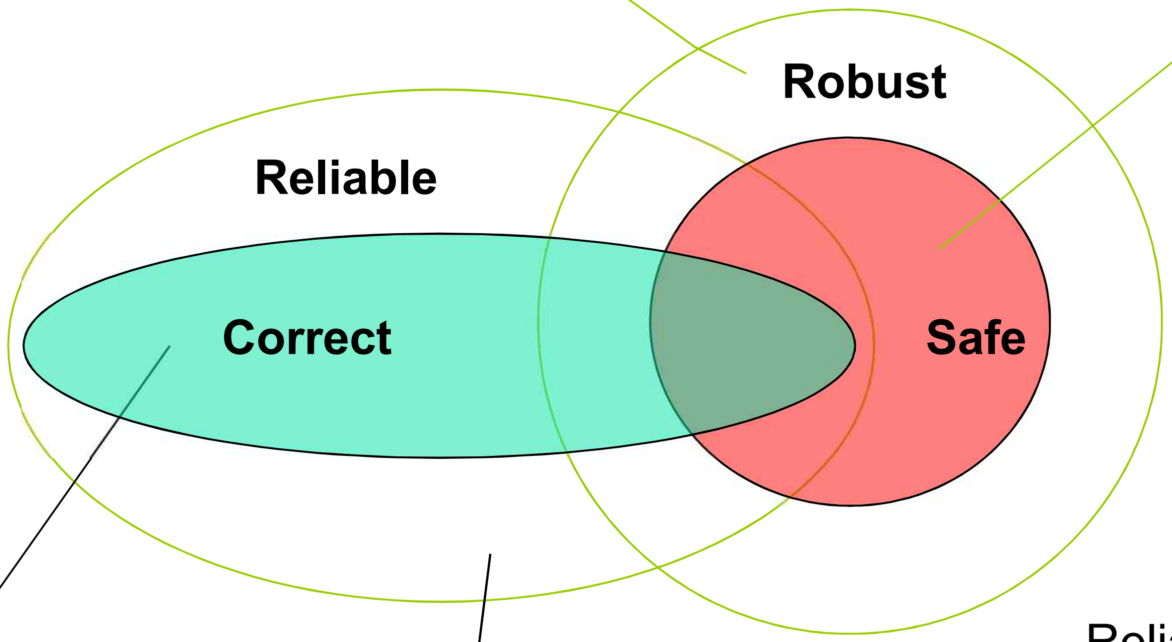
Robustness

- Robust if it behaves reasonably, even in circumstances that were not anticipated in the requirements specification.
- A function of a number of factors such as
  - The range of operating conditions.
  - The possibility of unacceptable results with valid input.
  - The acceptability of effects when the product is given invalid input.
- Q: Relation to reliability

Relationship between Quality Factors

Robust but not safe: catastrophic failures can occur

Safe but not correct: annoying failures can occur



Reliable but not correct: error conditions occur rarely

Correct but not safe or robust: the specification is inadequate

Performance

- Equates performance with efficiency(space, time).
- Affects the usability of the system.
- Evaluation
  - Measurement (monitoring)
  - Analysis
  - Simulation
- Q: When do we need to estimate performance?
- Application of performance to process

  ⇒ Productivity

User Friendliness

- Ease to use
- Ease with which the system can be configured and adapted to the *hardware environment.*

Verifiability

- Verifiable if its properties can be verified safely.
- Performed either by formal analysis methods or through testing.

Maintainability (1/2)

- Maintenance
  - Corrective
  - Adaptive
  - Perfective
- Software evolution (instead of maintenance)
- Repairability and Evolvability

Maintainability (2/2)

- Repairability
  - Repairable if a software system allows the correction of defects with a limited amount of work.
  - Improved through the use of proper tools: HL PL, CASE, etc.
  - Achieved by modularization
- Evolvability:
  - Evolvable if it allows changes that enable it to satisfy new requirements.
  - Modified over time
    - To provide new functions.
    - To change existing functions.
  - Restructuring or re-engineering

Reusability

- Use existing components to build a new product.
    - Examples: Scientific libraries, Motif, Unix shell, etc
- Reuse levels
    - People
    - Requirements
    - Design
    - Code
    - Domain analysis, etc.
- Object-oriented technology, Components, Services, etc.
- Application to process
    - Software methodology
    - Life cycle model

Portability

- Portable if it can run in different environments
  - hardware platforms
  - software platforms

Understandability

- An internal product quality.
  - internal quality factor - it helps in achieving many of other quality.
  - Maintenance easily
  - from an external point of view, understandable if it has predictable behavior.
- Object-oriented paradigm claims ease to understand

Interoperability

- Ability of a system to coexist and cooperate with other systems
- Open system concept

Productivity

- A quality of the software production process: to measure the efficiency of process
- Difficult to measure: simple metric: LOC
- Tools increase the productivity

Timeliness

- Process-related factor
  - Ability to deliver a product on time.
- Requires
  - careful scheduling,
  - accurate work estimation and
  - clearly specified milestones.
- Use incremental delivery to achieve it

Visibility

- Process-related
- Visible if all of its steps and its current status are documented.
- Allows to weigh the impact of their actions and thus guides them in making decisions.

Quality Requirements in Specific Application Areas

- Information Systems
- Real-time Systems
- Distributed Systems
- Embedded Systems

Information Systems

- Storage and retrieval of data
- Business area requirements
  - Examples: banking systems, library-cataloging systems, etc.
  - Quality requirements
    - Data integrity, Data availability, Transaction performance
    - Security
    - User interface

Real-time Systems

- Respond within a predefined and strict time periods
  - Examples: factory-monitoring systems, missile guidance systems, mouse-handling software
- Control-oriented
- Scheduling
  - Deadline
  - Priority
- Hard and soft real-times
- Quality
  - Respond time requirements (correctness criterion)

Distributed Systems

- The degree of distribution
  - Data
  - Control
  - Hardware
- Examples: middleware in client/server systems, groupware, etc.
- Qualities
  - System availability
  - parallelism
  - task allocation
  - partitioning
  - fault tolerant, …

Embedded Systems

- Software is one of many components.
- Often has no interface to end-user.
- Examples: Airplanes, robots, microwave ovens, dishwashers, automobiles, etc.