**Vietnam and Japan Joint**
**ICT HRD Program**

ITSS Software Development

# Chapter 2. Quality of Software Product

Nguyen Thi Thu Trang
trangntt-fit@mail.hut.edu.vn



## Content

# Content

1. What is Quality of Software Product
2. Quality model
3. Quality Evaluation Process



# 1. What is Quality of Software Product?

- Software Quality is explained as
  - "…..the ability to satisfy stated and implied needs" in "ISO/IEC 9126-1:2001 Software engineering -- Product quality -- Part 1: Quality model"
- Software Quality is classified following three kinds of qualities depending on by whom and when they are evaluated
  - Internal Software Quality: The quality evaluated by the developer in Analysis, Design, and construction process.
  - External Software Quality: The quality evaluated by the developer, or tester in testing process
  - Quality in Use: The quality evaluated by the user after product delivery

# Software product quality evaluation

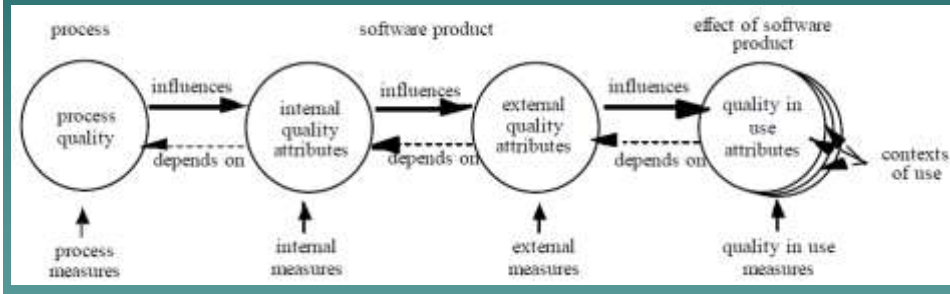
- Software product quality can be evaluated by measuring [2]
  - Internal attributes: typically static measures of intermediate products
  - or by measuring external attributes: typically by measuring the behavior of the code when executed
  - or by measuring quality in use attributes.

[2]: Session 5.1; pp. 3

5

# Quality in the lifecycle

- The objective is for the product to have the required effect in a particular context of use [2]



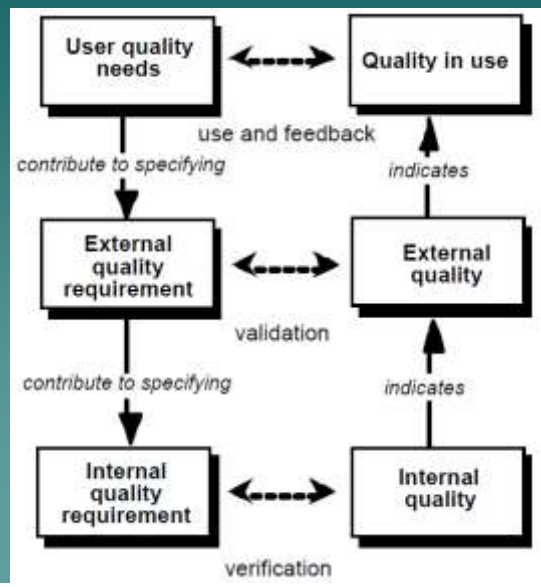

[2]: Figure *Quality in the lifecycle* in Session 5.1; pp. 3

6

## Quality in the software lifecycle

- There are different views of product quality and associated metrics at different stages in the software lifecycle [2]

[2]: Figure *Quality in the software lifecycle* in Session 5.2; pp. 4




## Content

1. What is Quality of Software Product?
2. Quality in use
3. External Software Quality
4. Internal Software Quality
5. Quality Evaluation Process

8

# 2. Quality in Use

- Quality in Use is evaluated as
  - How the software product can satisfy user needs in "Effectiveness", "Productivity", "Safety", and "Satisfaction"?
- Quality in use is the user's view of the quality of an environment containing software, and is measured from the results of using the software in the environment, rather than properties of the software itself [2].


[2]: Figure *Quality model for quality in use* in Session 7; pp. 12



# 2. Quality in Use (2)

- After the shipment of the software product, Quality in Use will most likely be evaluated by questionnaire survey to the user.
- The evaluation results of Quality in Use is a feed back material for next version or generation of the product
- Achieving quality in use is dependent on achieving the necessary external quality, which in turn is dependent on achieving the necessary internal quality[2]

  → In order to get good evaluation of quality in use of the user, it is necessary to get high score of Internal & External Software Quality during the development period

[2]: Session 7; pp. 12

# 2. Quality in use (3)

- Four characteristics
  - **Effectiveness**[2]
    - The capability of the software product to enable users to achieve specified goals with accuracy and completeness in a specified context of use
  - **Productivity**[2]
    - The capability of the software product to enable users to expend appropriate amounts of resources in relation to the effectiveness achieved in a specified context of use.

[2]: Session 7.1; pp. 12



# 2. Quality in use (5)

- Four characteristics (cont.)
  - **Safety**[2]
    - The capability of the software product to achieve acceptable levels of risk of harm to people, business, software, property or the environment in a specified context of use
  - **Satisfaction**[2]
    - The capability of the software product to satisfy users in a specified context of use

[2]: Session 7.1; pp. 13

# Content

1. What is Quality of Software Product?
2. Quality in use
3. External Software Quality
4. Internal Software Quality
5. Quality Evaluation Process



# 3. External Software Quality

- External Software Quality has the following 6 viewpoints to evaluate, named “quality characteristics”
  - Functionality: Ability to provide the functions to meet the needs
  - Reliability: Ability to maintain its performance level
  - Usability: Ability to be understood, to be acquired, and to be used easily, and to be attractive
  - Efficiency: Ability to provide high performance comparison with amount of resource used
  - Maintainability: Ability to be maintained easily
  - Portability: Ability to be transferred from one environment to another

## 3. External Software Quality (2)

- Each external quality characteristic could be also considered as a kind of requirements such as
  - **Functional requirements**: What functionality the software has?
    - Pretty well-known requirements!! It is not so easy to analyze, define and evaluate it
  - And there are 6 kinds of **Non- functional requirements**
    - **Reliability requirements**
    - **Usability requirements**
    - **Efficiency requirements**
    - **Maintainability requirements**
    - **Portability requirements**
- Can you image how to define and evaluate them of Software Application?

15

## 3. External Software Quality (3)

- Every quality characteristics has some sub-characteristics on which developers can build the quality into the software and evaluate it
- The concept of sub-characteristics for the quality is introduced base on “ISO/IEC 9126-1:2001_Software engineering -- Product quality -- Part 1: Quality model”

16

# 3.1. Functionality

- We declare in slide of 3.4.1 in Chapter 1: "**The functional and the interface requirements external to the software are analyzed and defined using Use Case Diagram and Use Case Scenarios**".
- Typical sub-characteristics of Functionality are:
  - **Suitability**: Ability to provide the suitable functions to meet the needs.
  - **Accuracy**: Ability to provide the accuracy functions to meet the needs.
    - e.g. Number of significant figures of calculation.

17

# 3.1. Functionality (2)

- There, not only **Suitability**, **Accuracy**, but also the following sub-characteristics are important for the Functionality
  - **Interoperability:** Ability to interact with other systems.
    - e.g.: To ensure interoperability with other system within the University Management System.
  - **Security:** Ability to protect against unauthorized access.
    - e.g.: Only registered user can log in the system. Only users logged in the system can access each facility such as the Course Registration System, and …
  - **Functionality Compliance:** Ability to adhere the standards or conventions related to function.
    - e.g.: To adhere "personal information protection law"

18

# 3.1. Functionality (3)

- Then, next question is how to evaluate and build-in the functional and the interface requirements?
- In upper stream of the development, we must conduct some kind of reviews such as External and Internal Design Review, Coding review…
- And in testing, it is also important based on functionality

19

# 3.2. Reliability

- Four sub-characteristics of the Reliability
  - **Maturity:** Ability to avoid the failure caused by implicit faults in software.
    - e.g.: It is often required by MTBF(Mean Time Between Failures).
  - **Fault tolerance:** Ability to maintain the specified performance when failures occur in the software.
    - e.g.: "The system shall be available 24 hours a day, 7 days a week, with no more than 10% down time."
  - **Recoverability:** Ability to reestablish the specified performance and recover the data when failures occur in the software.
    - e.g.: It is often required by MTTR (Mean Time To Restore).
  - **Reliability Compliance :** Ability to adhere the standards or conventions related to reliability.

In Requirements Analysis process, Reliability requirements are defined and in Testing Process, the requirements are evaluated

20

# 3.3. Usability

- The sub-characteristics of the Usability
  - **Understandability:**
    - Ease of understanding the concepts/applicability of the software
    - e.g.: The new registered student can understand the Course Reg. Sys. It is required unified concepts and applicability as University Mngr. Sys
  - **Learnability**
    - Ease of learning the operation and usage of the software
    - e.g. The tutorial to learn the operation within 10 minutes, is required. The help system for new users to assist all main operations is required



# 3.3. Usability (2)

- The sub-characteristics of the Usability (cont.)
  - **Operability**
    - Ease of operation and operation management of the software
    - e.g.: The new registered student can operate it without operation manuals

    →Recommend to have the same operation style as IE with tool chips
  - **Attractiveness**
    - Ability to be attractive software for the users
  - **Usability compliance**
    - Ability to adhere the standards or conventions related to usability

# 3.3. Usability (3)

- How to build-in and evaluate the required usability?

- Before that, developers have to make an agreement with the customer.

→ **Prototyping** of "the user screens and, if possible, major operations", in the **Software Requirement Analysis process**.

**"User Interface Prototyping"**

- In **the design processes**, they are built-in and evaluate with reviews.
- And in **the testing process**, internal usability evaluation testing is effective

23

# 3.4. Efficiency

- The sub-characteristics of the Efficiency
  - **Time behavior**
    - To show how response time, processing time and, throughput rate is taken to perform the function
    - E.g. *The system shall support up to 50 users to operate simultaneously "Course Registration" operation and all users can receive the result of the operation within 10 seconds*
  - **Resource behavior**
    - Ability to use adequately the amount of resources such as memory/disc, and communication resource
  - **Efficiency compliance**
    - Ability to adhere the standards or conventions related to efficiency

24

# 3.4. Efficiency (2)

- The previous slide said as an example of Time behavior
  - *The system shall support up to 50 users to operate simultaneously "Course Registration" operation and all users can receive the result of the operation within 10 seconds*
- However, this is also system requirement. This means
  - In system design, the ability of severs, storages, networks and system architecture are designed and determined!
  - Software development is usually achieved under the system conditioned.
- Next question is when and how to build-in the requirements and evaluate it?

25

# 3.4. Efficiency (3)

- Software performance* design and evaluation
  - *): "Performance" is general term, is strongly related to "time efficiency" of Quality sub-characteristics
  - In the **software architecture design process**
    - the developer designs or selects the software architecture that meets to the requirements
    - the developer evaluate the design is met to the requirements or not!!

    → How to design and how to evaluate? → Later
  - In the program/software test of **Software Integration process**
    - the developer evaluate and confirm the software to meet requirements
    - How to test and how to confirm?

26

# 3.5. Maintainability

- The sub-characteristics of the Maintainability
  - **Analyzability**
    - Ease of diagnosis deficiencies or cause of failure
    - Ease of identification of the place to be modified
    - E.g. Documentations for maintenance are required
  - **Changeability**
    - Ease of change, for example, base on design change
    - E.g. Java is required to develop the software. (Because there are many Java programmers)
  - **Stability:** Ability to avoid unexpected effects of modification
  - **Testability:** Ease of Testing
  - **Maintainability compliance:** Ability to adhere the standards or conventions related to efficiency

27

# 3.6. Portability

- The sub-characteristics of the Maintainability
  - **Adaptability:** Ease of adaptation to different specified environment
  - **Installability:** Ease of install in a specified environment.
    - E.g. It is often required like “The operator in charge can install the software less than 2 hours”.
  - **Co-existence:** Ability to co-exist with a specified software.
  - **Replaceability (Compatibility):** Ability to replace a specified software and to be used with the same environment.
    - E.g. “It can be replaced the previous product without data conversion”.
  - **Portability compliance:** Ability to adhere the standards or conventions related to portability.

28

# 3.7. Summary

- All characteristics and sub-characteristics are important. However, they can be weighted depending on kind of the developed software,  the user, and its environments the software are running on.
- → In “Course registration system”, Reliability, Usability, Efficiency (Time behavior) are taken up as important non-functional requirements/quality-characteristics.
- → It could be thought that these quality-characteristics are specified by the customer

29

# 3.7. Summary (2)

- Not only Functional requirements, as for also non-functional requirements, developers have to be analyzed, defined and make agreement with customers. And developers have to build-in and evaluate the requirements/quality characteristics
- → Sometimes, it have to be considered as activities on the software development process such as the case of Reliability, Usability, Efficiency (Time behavior) in the “Course registration system”.

30

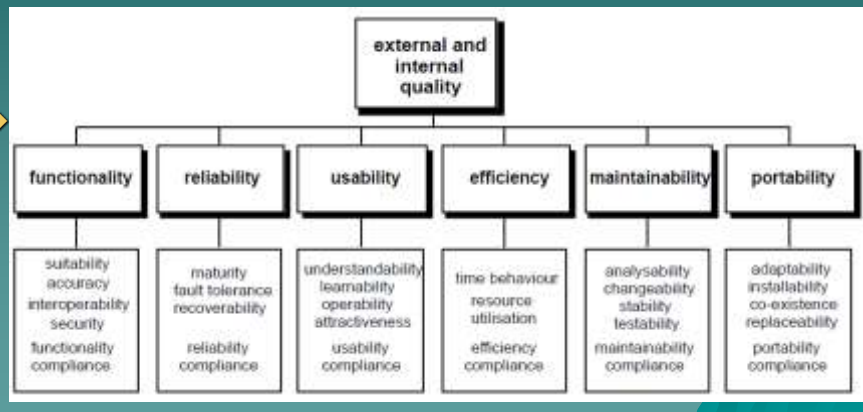
# Content

1. What is Quality of Software Product?
2. Quality in use
3. External Software Quality
4. Internal Software Quality
5. Quality Evaluation Process



# 4. Internal Software Quality

- Characteristics and sub-characteristics are the same as those of External Software Quality[2].




[2]: Figure 4 - *Quality model for external and internal quality* in Session 6, pp.7

# 4. Internal Software Quality (2)

- The difference between Internal & External Software quality is **when** they are evaluated. As a result, the metrics, which means measurement, rating and evaluation way, are different.
- → Internal Software Quality is required and evaluated within form software requirements analysis process to software integration by software developer.
- → The metrics are internal metrics such as “number of failure pointed out internal design review (Reliability)”, “number of internal call to database from a specified major use case operation (Time behavior)”



# Content

# 5. Quality Evaluation Process

- Quality Evaluation Process based on ISO/IEC 14589-1 “Information technology - Software product evaluation – Part 1 General Overview, 1999”

1. Establish the evaluation requirements: Aim of evaluation and, who and in what scene each evaluation will be done are decided

↓

2. Establish the evaluation specification: Measurement methods, criteria for evaluations are established

↓

3. Design evaluation: Create the evaluation plan

↓

4. Achieve evaluation:

: 1. collect evaluation value, 2. compare with certified value, 3. over-all judgment

35

# 5.1. How to build and evaluate the quality

- Select the important quality sub-characteristics for the project
- For our Case Study “Course Registration System”, the following quality sub-characteristics are selected and should be built and evaluated
  - As for **Functionality:**
    - In general, the common review policies are existed and external (Function) test is available

    → Experience and lecture in latter classes
  - As for **Security**: Based on system standards
  - As for **Reliability**: Systematic Testing and Implicit Fault Evaluation are available as basic methods → Latter

36

## 5.1. How to build and evaluate the quality (2)

- For our Case Study “Course Registration System”, the following quality sub-characteristics are selected and should be built and evaluated (cont.)
  - As for Usability: Prototyping method is available → Experience and lecture in latter classes
  - As for Performance: Investigations and prototyping if necessary
  - As for Maintenance: Depending user or system equipments.

37

## 5.2. How to build and evaluate the performance

- Investigates some development cases based on Struts to confirm Struts based development meets to the performance requirements such as
  - *“The system shall support up to 50 users to operate simultaneously “Course Registration” operation and all users can receive the result of the operation within 10 seconds.”*
- If we cannot find out the suitable development case report, Prototyping to confirm Struts based development meets to the performance requirements is very effective. In this case, this prototyping should be achieved in **Software requirements analysis** or **software architectural process**

38

# References

[1] ISO/IEC FDIS 12207, *Systems and software engineering — Software life cycle processes*.

[2] ISO/IEC 9126-1:2001, *Software engineering – Product quality – Part 1: Quality model.*