CSE 321Software Engineering

Software Quality Concept

Md. Shamsul Haque

Quality ?

Quality – A Pragmatic View

User view

Quality in terms of an end-user's specific goals.

If a product meets those goals, it exhibits quality.

This is fitness for purpose.

Quality – A Pragmatic View

- *Product View*

Tied to inherent characteristics (e.g., functions and features) of a product.

Microsoft Lumia 435

Key features

Highlights :

windows 8.1 os with lumia denim
2 mp primary camera
0.3 mp secondary camera
4-inch touchscreen
dual sim (gsm + gsm)

Quality – A Pragmatic View

- *Manufacturer's view*

Defines quality in terms of the original specification of the product. If the product conforms to the spec, it exhibits quality.

Processor	1GHz
RAM	512MB
Internal storage	4GB
Expandable storage	Yes
Expandable storage type	microSD
Expandable storage up to (GB)	32

Quality – A Pragmatic View

- *Transcendental view*

You immediately recognize, but cannot explicitly define.

I can't define it, but I know when I see it.

Quality – A Pragmatic View

- *Value-based view*

  How much a customer is willing to pay for a product.

Quality

- Quality encompasses all these views:
- *transcendental view*
- *user view*
- *manufacturer's view*
- *product view*
- *value-based view*


htc
4
07
PM
23°C

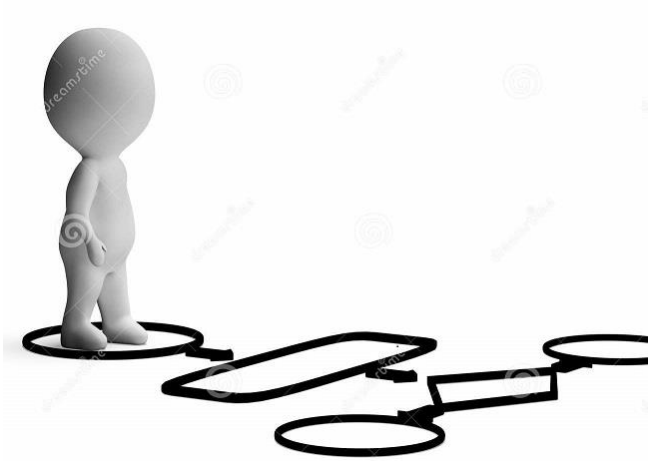

NOKIA

Software Quality

- An **effective software process** applied in a manner that creates **a useful product** that provides measurable **value for those who produce it and those who use it.**

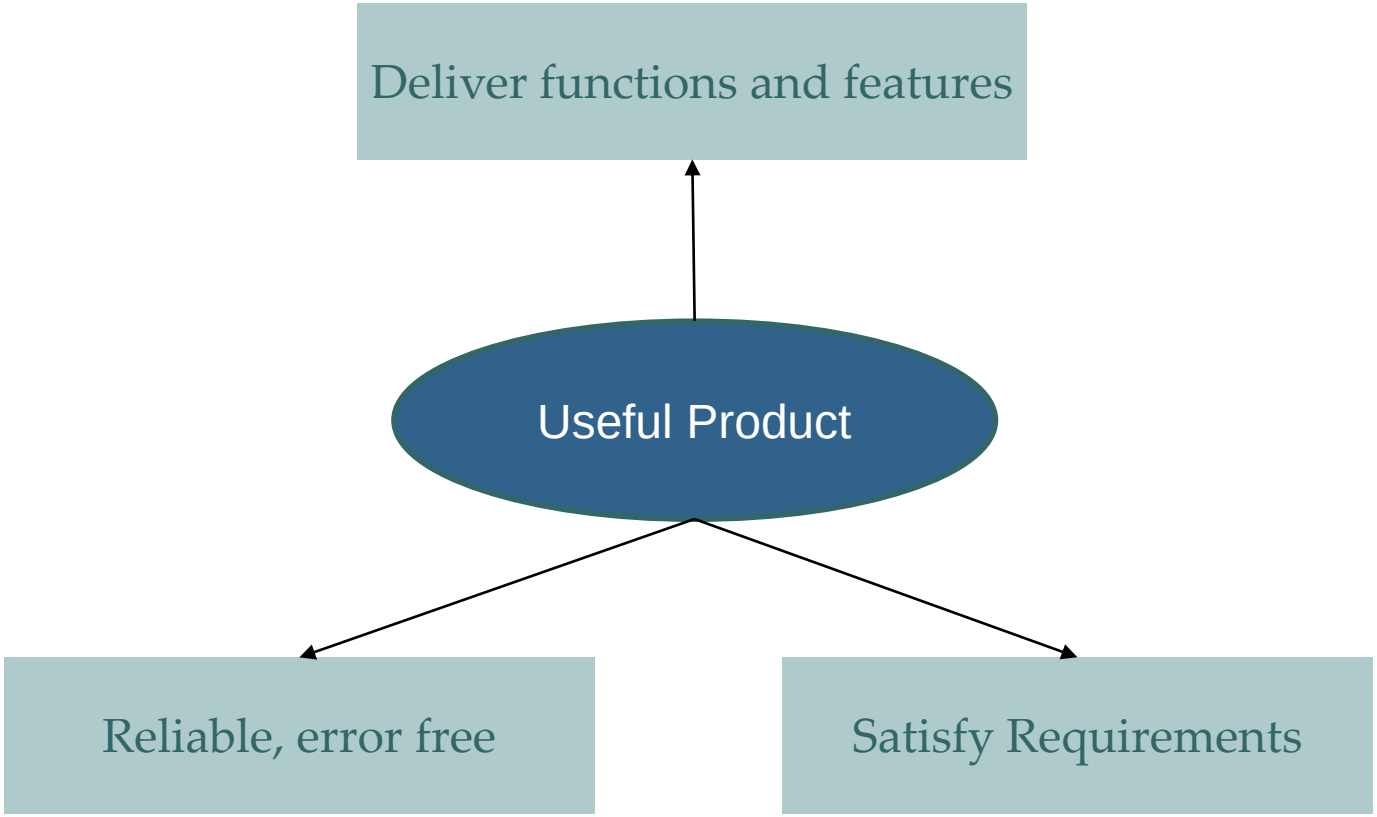
- **Effective software process**
- **Useful product**
- **Value for those who produce it and those who use it.**

Effective Software Process



- The management aspects of process create the checks and balances that help avoid project chaos.
- Software engineering practices.
- Umbrella activities such as change management and technical reviews must performed, Continuous Improvement

Useful Product



Value for both producer and user

The software organization gains added value :

Less maintenance effort

Fewer bug fixes

Reduced customer support.

The user/customer gains added value :

Greater software product revenue

Better profitability when an application supports a business process

Improved availability of information that is crucial for the business

Garvin's Quality Dimensions

- David Garvin's Eight dimensions of Quality:
  - **Performance Quality.**
    Does the software deliver all content, functions, and features that are specified as part of the requirements model in a way that provides value to the end-user?
  - **Feature quality.**
    Does the software provide features that surprise and delight first-time end-users?
  - **Reliability.**
    Does the software deliver all features and capability without failure? Is it available when it is needed? Reliability is the likelihood that a product will not fail within a specific time period

Garvin's Quality Dimensions

- **Conformance.** Does the software conform to local and external software standards that are relevant to the application? Does it conform to de facto design and coding conventions?
- **Durability.** Can the software be maintained (changed) or corrected (debugged) without unintended side effects ? Durability measures the length of a product's life
- **Serviceability.** Can the software be maintained (changed) or corrected (debugged) in an acceptably short time period. Can support staff acquire all information they need to make changes or correct defects?

Garvin's Quality Dimensions

- **Aesthetics.** Most of us would agree that an aesthetic entity has a certain elegance, a unique flow, and an obvious "presence" that are hard to quantify but evident nonetheless.

- **Perception.** Perception of the quality of the product in the mind of the consumer.
  Honda cars, Sony Walkman and Rolex watches are perceived to be high quality items by the consumers.

McCall's Quality Factors

influence
ability to change
the software product

influence
the ability to
adapt the software to
new environments

Basic
operational
characteristics that influence
the extent to which the
software fulfils its
specification

McCall's Quality Factors

Quality Factors	Definitions
Correctness	The extent to which a program satisfies its specifications and fulfills the user's mission objectives.
Reliability	The extent to which a program can be expected to perform its intended function with required precision.
Efficiency	The amount of computing resources and code required by a program to perform a function.
Integrity	The extent to which access to software or data by unauthorized persons can be controlled.
Usability	The effort required to learn, operate, prepare input, and interpret output of a program.
Maintainability	The effort required to locate and fix a defect in an operational program.
Testability	The effort required to test a program to ensure that it performs its intended functions.
Flexibility	The effort required to modify an operational program.
Portability	The effort required to transfer a program from one hardware and/ or software environment to another.
Reusability	The extent to which parts of a software system can be reused in other applications.
Interoperability	The effort required to couple one system with another.

McCall's Quality Factors

Quality Categories	Quality Factors	Broad Objectives
Product Operation	Correctness	Does it do what the customer wants?
	Reliability	Does it do it accurately all of the time?
	Efficiency	Does it quickly solve the intended problem?
	Integrity	Is it secure?
	Usability	Can I run it?
Product Revision	Maintainability	Can it be fixed?
	Testability	Can it be tested?
	Flexibility	Can it be changed?
Product Transition	Portability	Can it be used on another machine?
	Reusability	Can parts of it be reused?
	Interoperability	Can it interface with another system?

ISO 9126 Quality Factors

Functionality:

The degree to which the software satisfies stated needs as implied

Reliability:

The amount of time that the software is available for use

Usability:

The degree to which the software is easy to use.

ISO 9126 Quality Factors

Efficiency.

The degree to which the software makes optimal use of system resources

Maintainability.

The ease with which repair may be made to the software

Portability.

The ease with which the software can be transposed from one environment to another

The Software Quality Dilemma

- “Good Enough” Software

Good enough software delivers high quality functions and features that end-users desire, but at the same time it delivers other more obscure or specialized functions and features that contain known bugs

Cost of Quality

Prevention cost

?

?

Appraisal Cost

?

?

Internal failure cost

?

?

External failure cost

?

?

Cost of Quality

Prevention cost

- Cost of management planning activities for QA and QC
- Cost of tools for test equipment
- Cost of test planning
- Cost of Training

Appraisal Cost

- Cost of conducting technical reviews for work products
- Cost of data collection and metrics evaluation
- Cost of testing and debugging

Internal failure cost

- Rework
- Repair

External failure cost

- Complaint resolution
- Product return and replacement
- Help line support
- Warranty work

Quality and Decisions

- *Quality **depends** on the decisions made while developing the project*
  - *Estimation decisions*
  - *Scheduling decisions*
  - *Risk-oriented decisions*

Negligence and Liability

- Work begins with the best of intentions on both sides,
  but by the time the system is delivered, things have gone bad.

- The system is late, fails to deliver desired features and functions
  is error-prone, and does not meet with customer approval.

Achieving Software Quality

- Critical success factors:
  - **Software Engineering Methods**
  - **Project Management Techniques**
  - **Quality Control**
  - **Quality Assurance**

Chapter 14 Review

CHAPTER 14 QUALITY CONCEPTS 398

Thank You