Software Quality Assurance & Testing
Lecture Note 1

Quality Engineering

Quality engineering -quality management- is a process that evaluates, assesses, and improves the quality of software.

Software Quality Engineering Process

Requirements and Specification

Functional requirements

- Actions that the product must do to be useful to its users

Non-functional requirements

- Such as performance, usability, or security.
- They are called quality attributes.

Specification

- It is a document that specifies in a complete, precise, verifiable manner, the requirements, design, behavior, or other characteristics of a system

Quality Assurance

«Systematic, planned set of actions necessary to provide adequate confidence that the software development and maintenance process of a software system product conforms to established specification as well as with the managerial requirements of keeping the schedule and operating within the budgetary confine».

Daniel Galin, the author of the book Software Quality Assurance (2004)

ISO/IEC 25000
SQuaRE
(System and Software Quality Requirements and Evaluation)

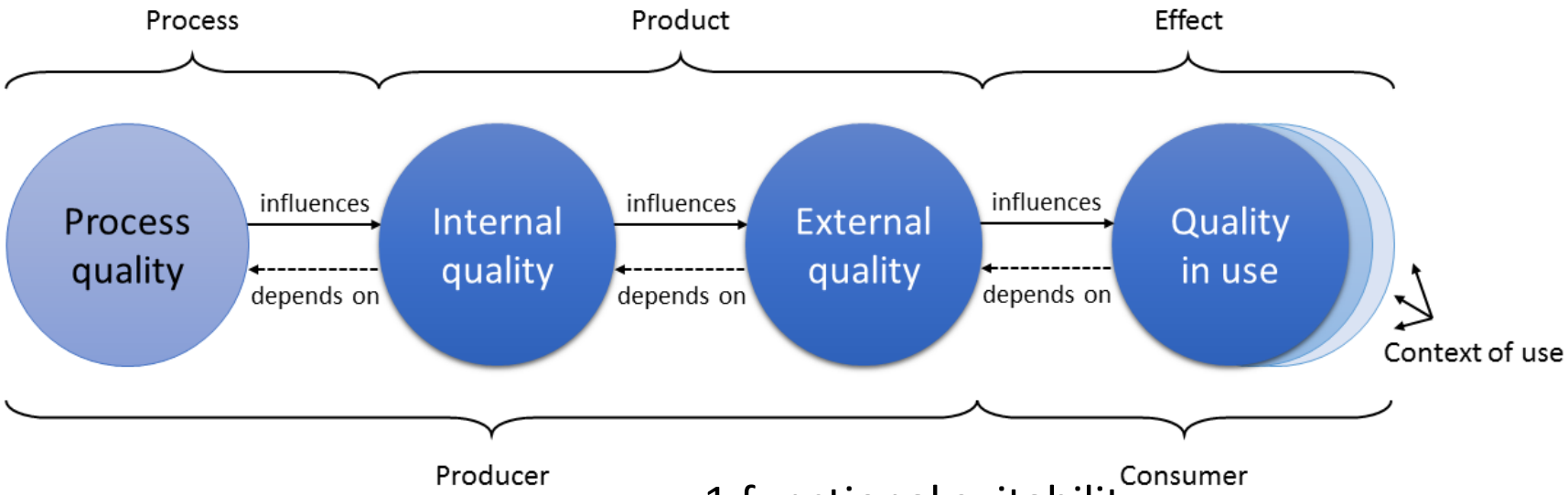
The ISO/IEC-2500 quality reference model distinguishes different views on software quality:

Internal quality: This concerns the properties of the system, that can be measured without executing it.

External quality: This concerns the properties of the system, that can be observed during its execution.

Quality in use: This concerns the properties experienced by its consumer during operation and maintenance of the system.

ISO/IEC-2500 Product Quality Reference Model



The quality model of ISO/IEC-25000 divides the product quality into eight top-level quality features:

1.functional suitability
2.performance
3.compatibility
4.usability
5.reliability
6.security
7.maintainability
8. portability

Functional Suitability

- Represents the degree to which a product or system provides functions that

  meet stated and implied needs

  when used under specified conditions.

Performance efficiency

- Represents the performance relative to the amount of resources

  used under stated conditions.

Compatibility

- This is the degree to which a product, system or component

  can exchange information with other

  products, systems or components

  and/or

  perform its required functions

- A product, system or component

shares the same hardware or software environment.

Usability

- This is the degree to which a product or system

  can be used by specified users

  to achieve specified goals with

  effectiveness, efficiency, and

  satisfaction

  in a specified context of use.

Reliability

- This is the degree to which a system, product, or component

  performs specified functions

  under specified conditions

  for a specified period of time.

Security

- This is the degree to which a product or system protects information and data so that persons or other products or systems have the degree of data access appropriate to their types and levels of authorization

Maintainability

- This represents the degree of effectiveness and efficiency with
  which a product or system can be modified to
    improve it
    correct it
    adapt it
      to changes in environment and in requirements

Portability

- This is the degree of effectiveness and efficiency with

  which a system, product, or component can be

  transferred from one hardware, software, or

  other operational or usage environment

  to another

Other Quality Characteristics : Effectiveness, Efficiency and Satisfaction

Effectiveness: This is the accuracy and completeness with which users achieve specified goals.

Efficiency: These are the resources expended in relation to the accuracy and completeness with which users achieve goals.

Satisfaction: This is the degree to which user needs are satisfied when a product or system is used in a specified context of use

Verification and Validation (V&V)

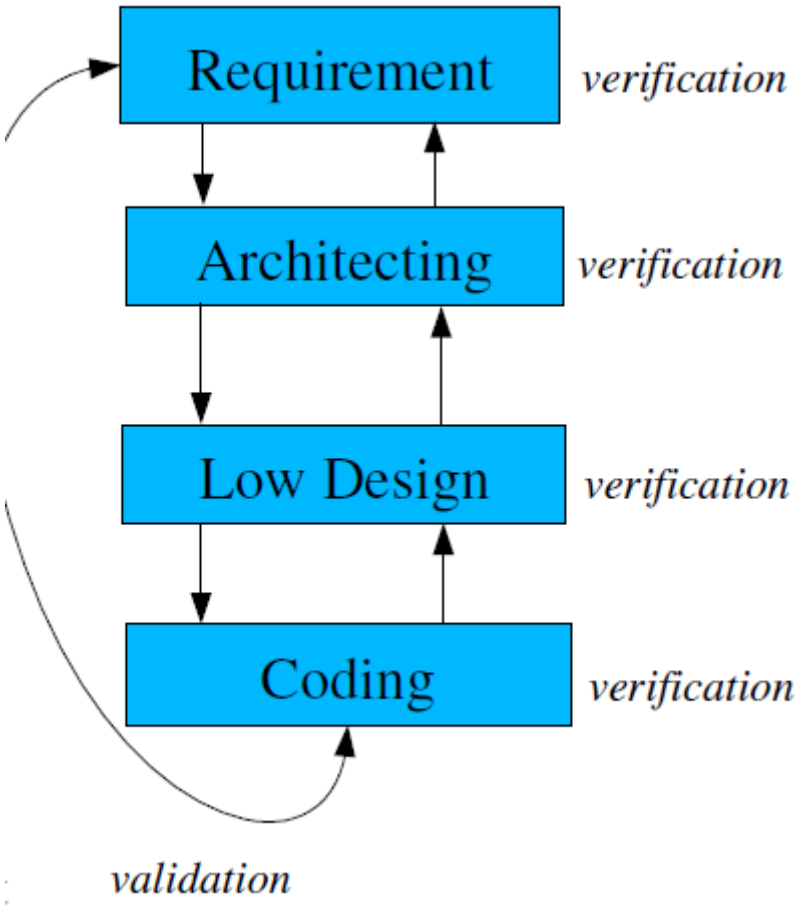
- V&V (Software Quality Control) is concerned with evaluating that the software being developed meets its specifications and delivers the functionality expected by the consumers.
- These checking processes start as soon as requirements become available, and continue through all stages of the development process.
- Verification is different to validation

Barry Boehm Expressed the Difference (in 1979)

- Verification: are we building the product right?
  - The aim of verification is to check that the software meets its stated functional and non-functional requirements (that is, the specification).
- Validation: are we building the right product?
  - The aim of validation is to ensure that the software meets consumer's expectations.
  - It is a more general process than verification
    - ✓Specifications do not always reflect the real wishes or needs of consumers.

V&V

- V&V activities include a wide array of QA activities.
- Although software testing plays an extremely important role in V&V, other activities are also necessary.
- Within the V&V process, two techniques of system checking and analysis may be used:
  - Software Testing(Dynamic Testing) Testing is an execution-based QA activitity.
  - Static Analysis: does not require execution of the software, works on a source representation of the software and may also be used as automated software analysis (the source code of a program is checked for patterns that are known to be potentially erroneous).


Requirement
verification
Architecting
verification
Low Design
verification
Coding
verification
validation

Software Defects

- Key to the correctness aspect of V&V is the concept of software defects.
- The term defect (bug) refers to a generic software problem.
- The IEEE Standard 610.12 propose the following taxonomy related to software defects:
  - Error
  - Fault
  - Failure

Error

- Human action that produces an incorrect result.
- Errors can be classified into two categories:
  - Syntax error: Program statement that violates one or more rules of the language in which it is written.
  - Logic error :Incorrect data fields, out-of-range terms, or invalid combinations.

Fault

- The manifestation of an error in the software system is known as a fault*.
  - For example, an incorrect step, process, or data definition.

* mismatch between requirements

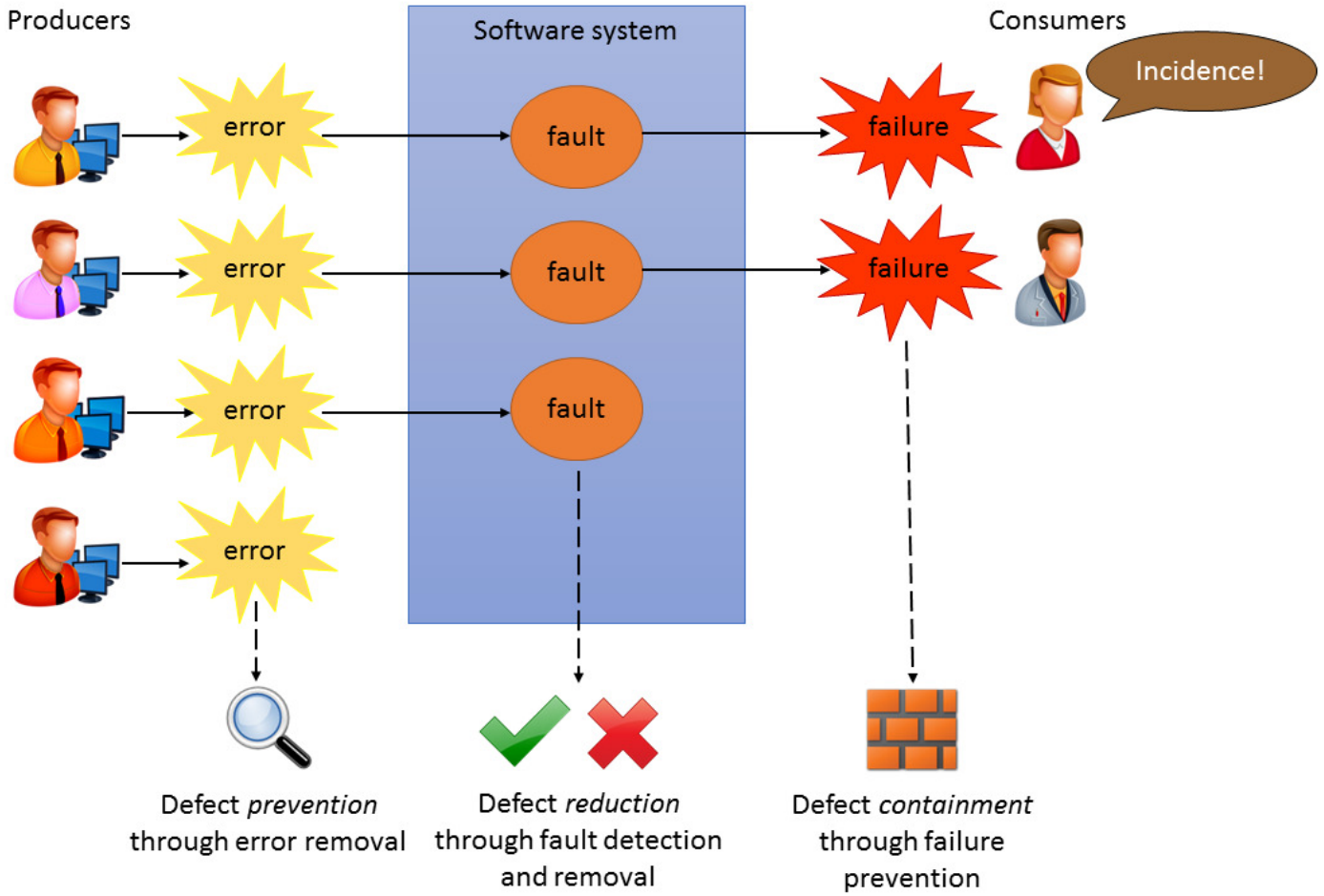
Failure

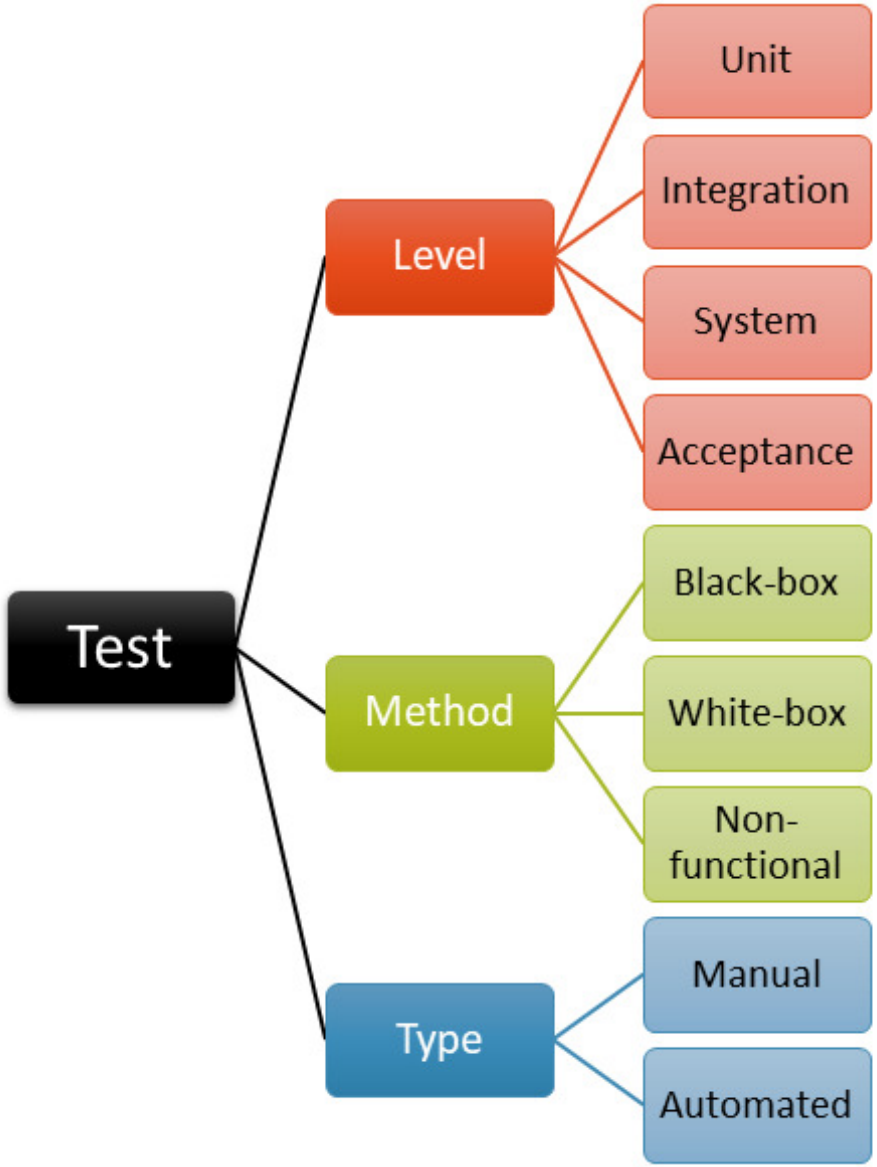
- The inability of the software system to perform its required functions is known as (system) failure.

The Revelation of the Term Bug

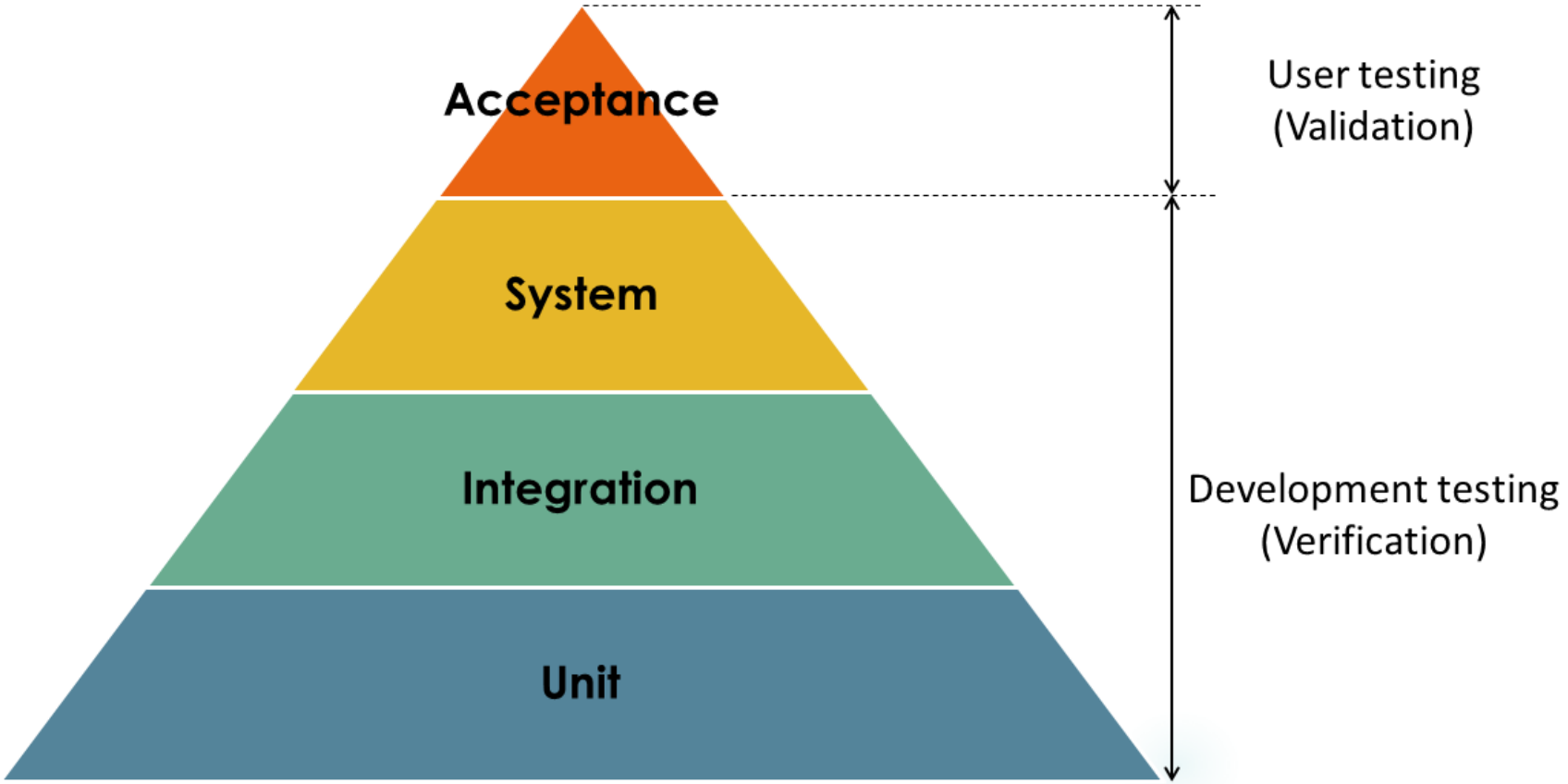
- The term bug was first coined in 1946 by the software pioneer Grace Hooper
- When a moth trapped in rely of an electromechanical computer caused a system malfunction.
- In this decade, the term debug was also introduced, as the process of detecting and correcting defects in a system.

Software defect chain and associated QA activities





Taxonomy of software testing in three categories: levels, methods, and types



Testing levels and its relationship with V&V

Software Testing Taxonomy

- When a final consumer uses a software product to validate if works as expected

  according the taxonomy called as a manual black-box acceptance test.

- Not all possible combination of these three classification of taxonomy is always meaningful.
  - For instance, non-functional tests (example, performance) is typically carried out automatically and at system levels

Faulty and Failure

- If a document with an error in it is used to specify a component, the component will be faulty and will *probably* exhibit incorrect behavior.
- If this faulty component is built into a system the system may fail.
- While failure is not always guaranteed, it is likely that errors in specifications will lead to faulty components