# Unit One

## SQA Concepts

- At the end of this chapter the student will be able to achieve the basic objectives :-
  - To discus about testing ,quality assurance and quality controls
  - To compare an contrast between Detection vs. Prevention.
  - To identify Verification & Validation
  - Discuss about testing.

# Understanding Software QA and Quality Control

- What exactly *software testing* means?
- This concept was introduced as a means of accessing the conformance of software with implicit and explicit expectations.
- implicit expectations :- Things customers expect from a product based on their previous experiences.
- explicit expectations :-These are specific targets customers expect in terms of product quality and performance.

# Understanding Software QA and Quality Control Cont.

- Here are the two types of software testing conducted for this purpose:

❖ **Functional:-** which focuses on the product's functionality from the user's point of view.

- e.g. its ease of use, convenience, UI design, absence of glitches, clear contact forms, price-to-quality ratio, and so on.

❖ **Non-functional (or structural);-** performed to assess the inner components of the software.

- e.g. its code maintainability, scalability to handle increased loads, security measures, compatibility across different platforms and browsers, and so on.

# Understanding Software QA and Quality Control Cont.

- Functional testing is performed through software quality management activities, such as QA, QC, and testing.

# What Is Software Quality Assurance?

- Quality Assurance (QA) or Software Quality Assurance (SQA) is a term used to describe a systematic process of assessing the quality of software solutions and implementing ways to improve them.
- SQA refers to the organizational part of quality management when everyone on the team knows exactly in what order things need to be done and strictly adheres to these rules. This is the basis without which further testing is impossible.

# Processes Involved

- Recognize and identify quality standards applicable in software development.
- Conduct regular quality reviews.
- Carry out processes for recording test data.
- Arrange and maintain documentation for QC measures.

# Methods Used in QA

- In general, there are two types non-functional and functional.
- the phases of non-functional testing, which look like this:

➢ **Vulnerability testing** aims to detect any potential issues that could expose a website or application to security threats such as cyberattacks, data breaches, or unauthorized access.

➢ **Compatibility testing** is conducted to check the product's compatibility with operating systems, browsers, hardware, and software to ensure that it functions correctly and consistently across different environments.

➢ **Usability testing** is executed to ensure that the product's user interface is easy to understand and use so that end users can interact with a website or application intuitively and efficiently.

# Methods Used in QA Cont.

- **Performance testing** allows teams to ensure that the product's performance remains consistent and flawless under various conditions, including high loads, low battery, and so on.

it's time to look more closely at functional testing methods. Here they are:

- **Unit testing** is used to test different parts and components of the software in isolation. Unit tests are written before building a module to test specific functions, classes, or modules in the codebase.

# Methods Used in QA Cont.

- **Integration testing** focuses on evaluating how different components or modules of the software interact with one another when integrated.
  - ✓ It verifies that data flows smoothly between these integrated parts and that they work cohesively as a whole.
- **System testing** extends to the whole framework, enabling test engineers and developers to identify stability issues and spot bugs they haven't noticed before.
- **Acceptance testing** is typically one of the final phases before the product's release, which involves end-users and stakeholders to validate that the product meets requirements and expectations.

# Advantages of Software QA

- The advantages of implementing quality assurance in software development are numerous, including but not limited to the following aspects:
- **Reduced risks of costly errors and rework**. Since most errors are prevented early in software development, less time and resources are spent on major technical reviews in later stages.
- **A high level of motivation**. When everyone understands the product requirements and expectations, they are motivated to contribute their best efforts towards achieving a common goal.

# Advantages of Software QA Cont.

- **A culture of continuous improvement**. With SQA requirements, teams rather collaborate with each other than supervise, which helps cultivate a culture of constant improvement and boosted productivity.
- **Compliance with industry standards and regulations**. Last but not least, software quality assurance testing helps ensure that products adhere to industry-specific standards and regulatory requirements.
  - ✓ This is particularly important for niches such as healthcare, finance, and manufacturing, where non-compliance may lead to financial consequences.

# Examples of QA

- Here are some examples of QA activities:
  - Project planning;
  - Internal/external audits;
  - Process analysis;
  - Establishing standards;
  - Arranging training courses for the team, and more.

# What Is Quality Control?

- QC refers to the meticulous examination of the software product to identify any deviations from established standards and specifications.
- It's the process of monitoring and inspecting each phase of the SDLC to ensure that the final product meets the desired quality criteria.
- To put it in simpler terms, think of it as having a manager run quality checks on the website's functionality by randomly testing its various components.

# Processes Involved

- Monitor and inspect software at different stages of the SDLC.
- Take precautions to prevent bugs and errors in the initial stages of software development.
- Constantly analyze data to identify areas for improvement.
- Keep records of the actions taken to maintain accountability.

# Methods Used in QC

- There are two common methods used in software quality control testing  inspection and statistical quality control.
- Starting with **inspection**, this approach falls into the following three categories:
  - 100% product inspection and sample inspection.
  - Process inspection.
  - Inspection analysis.

# Methods Used in QC Cont.

- When it comes to statistical quality control, it focuses on three key aspects:
  - Sampling plans and analysis, which allow organizations to measure the liquidity of the product by executing selective tests of product components.
  - Control charts
  - Corrective measures.
- By checking samples of the final product along with control charts showing quality deviations from acceptable limits, teams can decide whether to move forward with product deployment or whether corrective action needs to be taken.

# Advantages of QC

- **Customer loyalty**: Consistent delivery of high-quality software leads to building a loyal customer base.
- **Reduced costs**: No bugs and errors means no additional costs to fix software solutions.
- **Constant improvement**: Tracking common error patterns helps avoid them in the future.

**Examples of QC**

- Inspection;
- Random samples control;
- Verification and validation;
- Manual & automated testing;
- Measurements, etc.

# Prevention versus Detection

- The aim of the quality is to prevent quality defect or deficiencies in the first place and to make the product assessable by **quality assurance measures**.
- The total cost of effective quality management is the sum of four components:
  1- Prevention
  2- inspection
  3- Internal Failure
  4- External Failure
- The greatest payback is with prevention. Increasing the emphasis on prevention costs reduces the number of defects that go to the customer undetected, improved product quality, and reduces the cost of production and maintenance.

# Verification & Validation

- Verification
- "Are we building the product right".
- The software should conform to its specification.
- Verification checks the conformance of software product implementation against its specifications to see if it is implemented correctly.
- Validation
- "Are we building the right product".
- The software should do what the user really wants.
- Validation checks the conformance to quality expectations of customers and users in the form of whether the expected functions or features are present or not.

# Verification & Validation Process

- Verification & Validation are a life-cycle process.
- It should be applied at each stage in the software process.

- Verification & Validation has two principle objectives
  - The discovery of defects in a system;
  - The assessment of whether or not the system is useful and useable in an operational situation.

# Verification & Validation goals

- Verification and Validation should establish confidence that the software is fit for purpose.
- This does not mean completely **free of defects**.
- Rather, it must be good enough for its intended use and the type of use will determine the degree of confidence that is needed.

# Validation Activities

- Validation activities check whether a function needed and expected by the customers is present in a software product.

- An absence of an expected function or feature is clearly linked to a deviation of expected behavior, or linked to a software failure.

- Various QA activities linked with such kind of failures directly observable by software users can be viewed as validation activities

# Validation QA Activities

- QA activities that can be classified as Validation Activities are:

  - Acceptance Testing
  - System Testing
  - Usage-based Statistical Testing
  - Software fault tolerance
  - Software safety assurance activities

# Verification Activities

- Software verification activities check the conformance of a software system to its specifications.
- In performing verification activities, we assume that we have a well defined set of specifications.
- A deviation from the specification is either a fault or a failure.
- When failures are involved in verification activities, we are typically dealing with internal system failures and overall system failures in the form of incorrect behavior.
- When a function or feature expected by the customers is present, the activity to determine whether it performs or behaves expectedly is then a verification activity.

# Validation & Verification Activities

- Therefore, connected to validation activities, there are almost always accompanying verification activities as well.

- For Example:

Various forms of testing as primarily validation activities, they all include corresponding verification components.

# Verification & Validation confidence

- Depends on system's purpose, user expectation and marketing environment

❖ Software function
  - ➢ The level of confidence depends on how critical the software is to an organization.

❖ User Expectation
  - ➢ Users may have low expectations of certain kinds of software.

❖ Marketing Environment
  - ➢ Getting a product to market early may be more important than finding defects in the program.

# Independent Validation and Verification

## Developer

- Understands the system but will test “gently” and, is driven by the “delivery”.

## Independent Tester

- Must learn about the system, but will attempt to break it and, is driven by the “quality”.

# Static and Dynamic Verification

- Software inspections: Concerned with analysis of the static system to discover problems (static verification)
  - May be supplement by tool-based document and code analysis

- Software testing: Concerned with exercising and observing product behaviour (dynamic verification)
  - The system is executed with test data and its operational behaviour is observed

# Program Testing

- Can reveal the presence of errors NOT their absence.
- The only validation technique for non-functional requirements is the software has to be executed to see how it behaves.
- Should be used in conjunction with static verification to provide full V&V coverage.

# Types of testing

- Defect testing
  - Tests designed to discover system defects.
  - A successful defect test is one which reveals the presence of defects in a system.
- Validation testing
  - Intended to show that the software meets its requirements.
  - A successful test is one that shows that a requirements has been properly implemented.

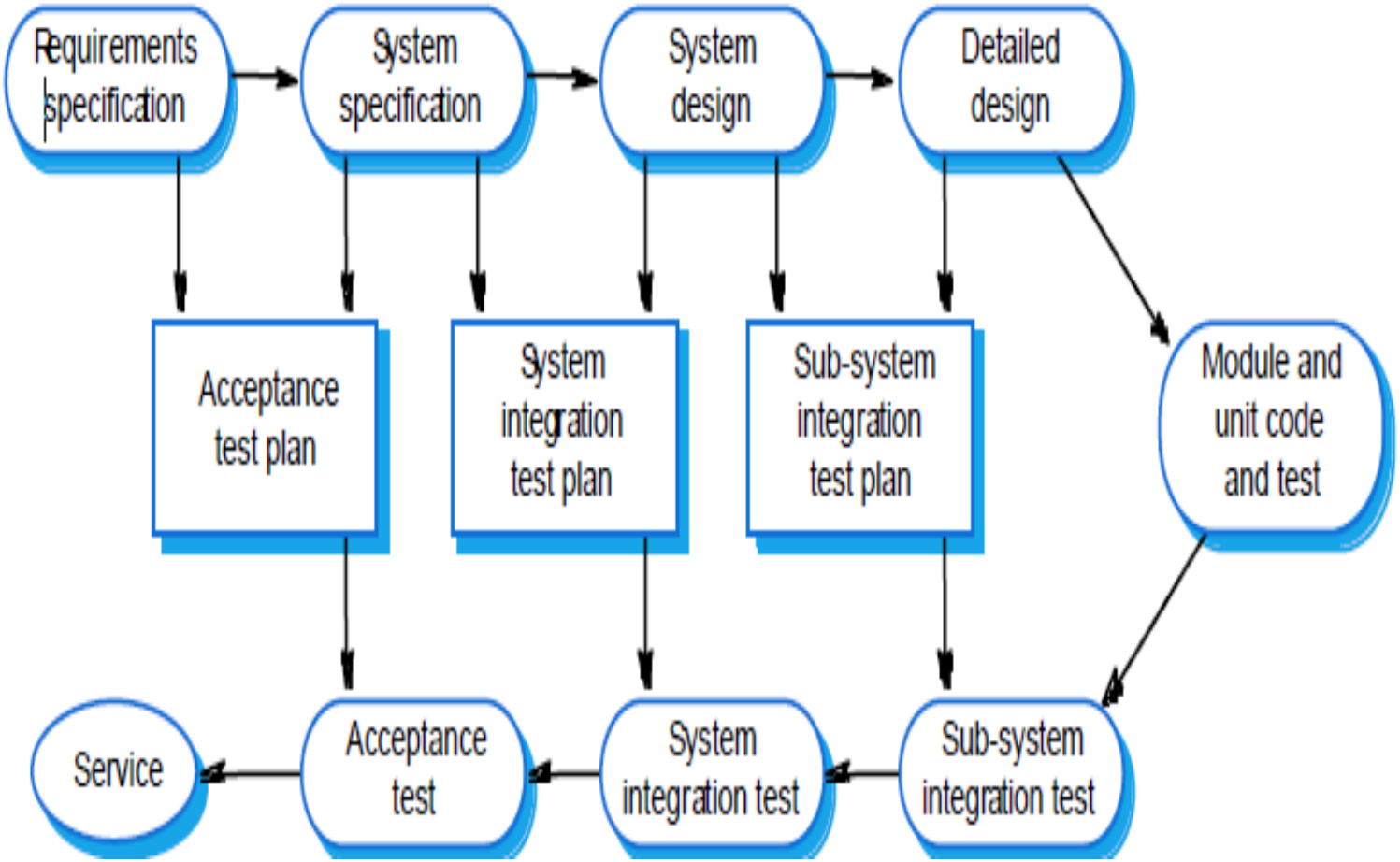
# Testing and debugging

- Defect testing and debugging are distinct processes.
- Verification and validation is concerned with establishing the existence of defects in a program.
- Debugging is concerned with locating and repairing these errors.
- Debugging involves formulating a hypothesis about program behaviour then testing these hypotheses to find the system error.

# Validation & Verification planning

- Careful planning is required to get the most out of testing and inspection processes.
- Planning should start early in the development process.
- The plan should identify the balance between static verification and testing.
- Test planning is about defining standards for the testing process rather than describing product tests.

# V-Model of Development



# What Is Testing?

- Testing is the process of checking a product as a whole and each of its parts separately.
- It can be carried out in many different ways using both manual approach and automated testing tools depending on the scale and goals of a project.

**Processes Involved**

- Develop a strategy outlining what aspects of the product will be tested, how testing will be conducted, and what criteria will define success.
- Create test cases that cover a wide range of functionalities of a product.
- Run test cases, either manually or through automated tools.

# What Is Testing? Cont.

- Document any issues or deviations from expected behavior.
- Repeatedly test the software after each change or update to ensure that new modifications haven't affected the product's performance and functionality.

| | QA | QC | Testing |
|---|---|---|---|
| Approach | Proactive | Reactive | Reactive |
| Focus | Process-oriented | Product-oriented | Focused on finding mistakes and bugs |
| Techniques | Developing strategies, establishing standards, creating test management plans | Verifying the quality of software using testing, code reviews, continuous | Conducting all types of testing through manual or automated tools |
| Timeframe | Throughout the development process | Before the product release | Along the development process |
| Executor | The entire team, including stakeholders | A small, dedicated team | Test engineers, developers |

*a comparison chart that summarizes the key points.*

# Thank You !!!