LECTURE: 5

SYED SAQIB RAZA RIZVI

LECTURE OUTLINE

- **What is Quality?**
- **What is Software Quality?**
- **What is Software Quality Assurance?**
- **Software Testing?**
- **Software Testing Goals?**
- **Defects and Failures?**
- **Static VS Dynamic Testing?**
- **Types of Software Testing?**
- **Software Testing Life Cycle?**
- **Testing and Inspection?**
- **Stages of Testing?**

What is Quality?

"A distinctive attribute or characteristic possessed by someone or something or Degree of excellence achieved"

Oxford Dictionaries definitions

What is Software Quality?

- **Pressman's definition:**

  *"Conformance to stated **functional and performance requirements**"*

- **IEEE Definition:**

*"The degree to which a system, component, or process **meets specified requirements**"*

- **German Industry Standard:**

*"Quality **comprises all characteristics** and significant features of a product or an activity which relate to the satisfying of given **requirements**"*

What is Software Quality Assurance?

- Software quality assurance (SQA) consists of a **means of monitoring the software engineering processes** and methods used to ensure **quality.**

- The function of software quality that assures that the **standards, processes, and procedures are appropriate for the project and are correctly implemented**.

- Specified **standards are used to define the development criteria** that are used to guide the manner in which software is engineered

Software Testing

- **Software testing** is a method of **assessing the functionality** of a **software** program
- Software testing is a process of executing a program or application with the intent of finding the **software bugs.**
- It can also be stated as the **process of validating and verifying** that a software program or application or product
- Meets the **business and technical requirements**
- Testing is intended to show that a program does what it is intended to do and to **discover program defects** before it is put into use.

Software Testing Goals

Verification:

Validation:

Priority Coverage:

Business Requirement Specification:

Detection and Finding of Bugs:

Determine Capabilities and Boundaries of Software Product:

Testing is Risk Reduction:

Defects and Failures

- Not all software **defects** are caused by **coding errors**
- Expensive defects are requirement gaps
- **Non-functional** requirements such as usability, performance, and security.
- A programmer makes an **error** (mistake), which results in a **defect** (fault, bug)
- **Defect** can turn into a **failure**

Static VS Dynamic Testing

- Reviews, or inspections are referred to as static testing
- Actually executing **programmed code** with a given set of test cases is referred to as **dynamic testing**
- Static testing involves **verification**, whereas dynamic testing involves **validation**

Types of Software Testing

- **Compatibility testing**
- **Smoke testing** (basic problems that will prevent it from working)
- **Regression testing** (old bugs that have come back after change)
- **Acceptance testing**(User)
- **Alpha Testing** (simulated or actual operation) and **Beta Testing**
- **Functional Testing** (specific action or function of the code)
- **Non-functional Testing** (behavior under certain constraints)
- **Performance Testing** (responsiveness and stability under a particular workload)

Types of Software Testing

- **Usability testing (**User interface is easy to use and understand**)**
- **Accessibility testing** (Under Certain Code or Standard)
- **Security testing** (Penetration)
- **A/B testing** (effect of one and other)
- **Concurrent testing** (basic operations and basic input to determine basic faults)
- **Many More**

Software Testing Life Cycle

- Total Quality Assurance
- QA in every phase of development V&V Model
- There is a typical cycle for testing:

1. **Test planning**:

Test strategy, test plan. Since many activities will be carried out during testing, a plan is needed. Priority, sensitivity etc.

2. Test development:

Test procedures, test scenarios, test cases, test datasets, test scripts to use in testing software.

Software Testing Life Cycle

3. Test execution:

Testers execute the software based on the plans and test documents then report any errors found to the development team.

4. Test reporting:

Once testing is completed, testers generate metrics and make final reports on their test effort and whether or not the software tested is ready for release.

5. Test result analysis:

Or Defect Analysis, is done by the development team usually along with the client, in order to decide what defects should be assigned, fixed, rejected (i.e. found software working properly) or deferred to be dealt with later.

Software Testing Life Cycle

6. Defect Retesting:

Once a defect has been dealt with by the development team, it is retested by the testing team.

7. Regression Testing:

In order to ensure that the latest delivery has not ruined anything.

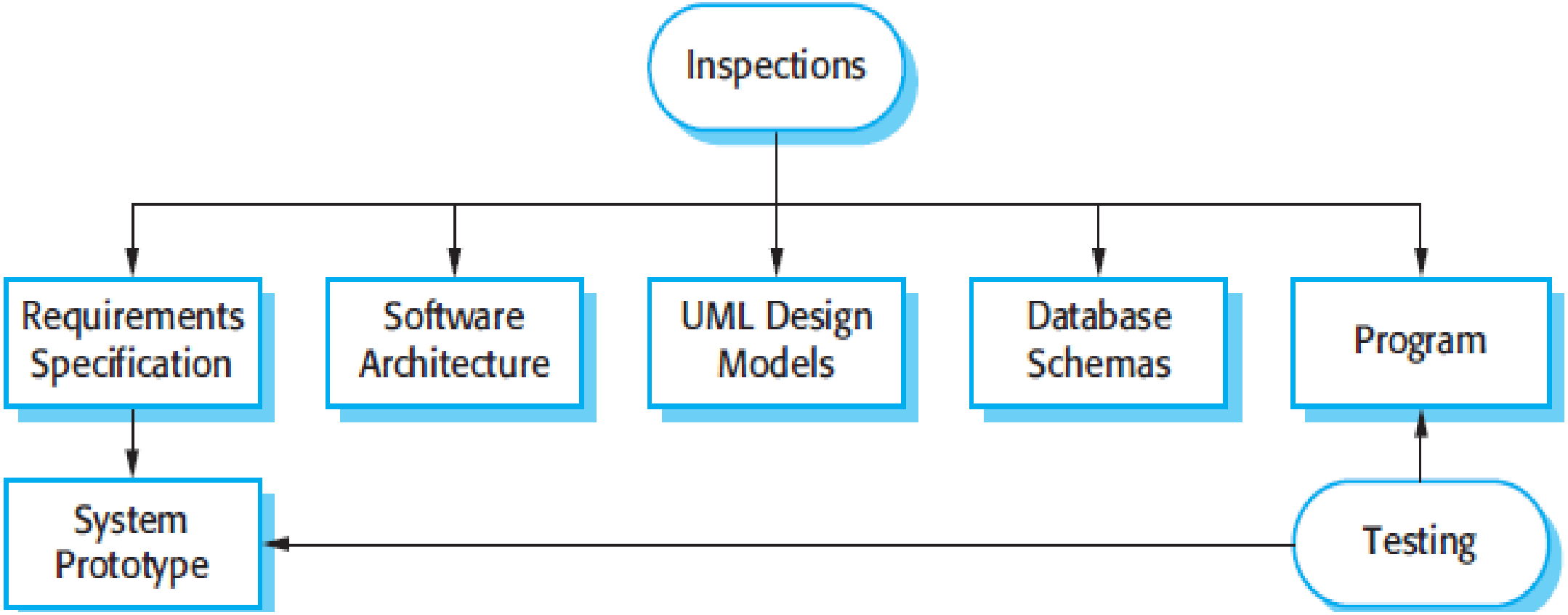
8. Test Closure:

Once the test meets the exit criteria, the activities such as capturing the key outputs, lessons learned, results, logs, documents related to the project are archived and used as a reference for future projects.

Testing and Inspection

- **Software inspections** Concerned with **analysis of the static system** representation to discover problems *(***static verification**)
- Inspections mostly focus on the **source code** of a system but any **readable representation** of the software, such as its **requirements** or a **design model**, can be inspected.
- When you inspect a system, you use knowledge of the system, its application domain, and the programming or modeling language to discover errors.
- **Software testing** Concerned with exercising and observing **product behaviour** (**dynamic verification**)
  - The system is executed with test data and its operational behaviour is observed.

Testing and Inspection



Inspection

- During testing, errors can mask (hide) other errors. Because inspection is a static process, you don’t have to **be concerned with interactions** between errors.
- Incomplete versions of a system can be inspected without additional costs. If a program is incomplete, then you need to develop **specialized test**, to test the parts that are available.
- As well as searching for program defects, an inspection can also consider broader quality attributes of a program, such as compliance with standards, portability and maintainability.
- Inspections and testing are complementary and not opposing verification techniques.
- Both should be used during the **V & V process**.
- Inspections can check **conformance with a specification** but not conformance with the customer’s **real requirements**.

Stages of Testing

- **Development testing**, where the system is tested during development to discover bugs and defects.

- **Release testing**, where a separate testing team test a complete version of the system before it is released to users.

- **User testing**, where users or potential users of a system test the system in their own environment.

1. Development Testing

- Development testing includes all testing activities that are carried out by the team developing the system.
  - **Unit testing**, where individual program units or object classes are tested. Unit testing should focus on testing the functionality of objects or methods.
  - **Component testing**, where several individual units are integrated to create composite components. Component testing should focus on testing component interfaces.
  - **System testing**, where some or all of the components in a system are integrated and the system is tested as a whole. System testing should focus on testing component interactions.

i. Unit Testing

- Unit testing is the process of testing individual program in isolation.
- It is a defect testing process.
- Units may be:
  - Individual functions or methods within an object
  - Object classes with several attributes and methods

Object Class Testing

- Complete test coverage of all the features of the class
  - Testing all operations associated with an object
  - set and check the value of all attributes associated with the object;
  - put the object into all possible states. This means that you should simulate all events that cause a state change.
- Inheritance makes it more difficult to design object class tests as the information to be tested is not localised.

ii. Component or Integration Testing

- Software components are often composite components that are made up of several interacting objects.
- You access the functionality of these objects through the defined component interface.
- Testing composite components should therefore focus on showing that the component interface behaves according to its specification.

iii. System Testing

- **System testing** of software or hardware is testing conducted on a complete, integrated system to evaluate the system's compliance with its specified requirements.
- System testing falls within the scope of black box testing, and as such, should require no knowledge of the inner design of the code or logic.
- System testing tests the overall behavior of a system.

2. Release Testing

- Release testing is the process of testing a particular release of a system that is intended for use **outside of the development team**.
- The primary goal of the release testing process is to convince the supplier of the system that it is **good enough for use**.
- Release testing, therefore, has to show that the system delivers **its specified functionality**, performance and dependability, and that it does not fail during normal use.
- Release testing is usually a **black-box testing process** where tests are only derived from the system specification.

Release Testing and System Testing

- Release testing is a form of system testing.
- Important differences:
  - A separate team that has not been involved in the system development, should be **responsible for release testing**.
  - System testing by the development team should focus on **discovering bugs in the system** (defect testing).
  - The objective of release testing is to check that the system **meets its requirements** and is good enough for external use (validation testing).

.

3. User Testing

- User or customer testing is a stage in the testing process in which **users or customers provide input and advice** on system testing.
- User testing is essential, even when comprehensive system and **release testing** have been carried out.
- The reason for this is that influences from the user's working environment have a major effect on the reliability, performance, usability and robustness of a system. These cannot be replicated in a testing environment.

User Testing

- **Alpha testing (Generic Software)**
  - Users of the software work with the development team to test the software at the developer's site.
- **Beta testing**
  - A release of the software is made available to users to allow them to experiment and to raise problems that they discover with the system developers.
- **Acceptance testing (Particular Company)**
  - Customers test a system to decide whether or not it is ready to be accepted from the system developers and deployed in the customer environment. Primarily for custom systems.

The End!!!