Fair Use Disclaimer Notice

The material used in this presentation i.e., pictures/graphs/text, etc. is solely intended for educational/teaching purpose, offered free of cost to the students for use under special circumstances of Online Education due to COVID-19 Lockdown situation and may include copyrighted material - the use of which may not have been specifically authorized by Copyright Owners. It's application constitutes Fair Use of any such copyrighted material as provided in globally accepted law of many countries. The contents of presentations are intended only for the attendees of the class being conducted by the presenter.

Outlines

- Software Reliability
- Factors Influencing Software Reliability
- SQA Plan
- SQA Group

Software Reliability

- It's a quantitative measure that is useful in assessing the quality of a software is its *reliability*.
- *Software reliability is* defined as the probability of failure-free operation of a software system for a specified time in a specified environment.
- The level of reliability of a system depends on those inputs that cause failures to be observed by the end users.

Software Reliability

- **FAILURE**
  - A failure is said to occur if the *observable outcome of a program execution is* different from the expected outcome.
  - The concept of observable outcome is very broad, and it encompasses a variety of things, such as values produced, values communicated, performance demonstrated, and so on.
  - The expected outcomes of program executions are specified as system requirements.
  - Two characteristics of failures are as follows:
    - failures are associated with actual program executions and
    - failures are observable concepts.

- **A FREQUENTLY FAILING SYSTEM IS CONSIDERED TO BE HIGHLY UNRELIABLE.**

Software Reliability

- FAULT
  - The adjudged cause of a failure is called a fault.
  - While constructing a software system, a designer may mistakenly introduce a fault into the system.
  - A fault can be introduced by having a defective block of code, a missing block of code for an unforeseen execution scenario, and so on.
  - One fault can cause more than one failure depending upon how the system executes the faulty code

Software Reliability

- **Time Interval between Failures**
  - A small time interval between successive failures tells us that the software system is failing frequently, and hence the reliability level is too low.
  - This can happen during system testing or even while the system is in operation.
  - If the time interval between successive failures is long, the reliability is perceived to be high, in spite of the occasional system failure.
  - The three commonly used metrics based on time intervals are the mean time to failure (MTTF), the mean time to repair (MTTR), and the mean time between failure (MTBF).
  - A useful relationship between the three metrics can be stated as MTBF = MTTF+MTTR.
  - In addition to a reliability measure, we must develop a measure of availability.
  - Software availability is the probability that a program is operating according to requirements at a given point in time and is defined as:

$$\text{Availability} = \text{MTTF} / (\text{MTTF} + \text{MTTR}) * 100\%$$

Factors Influencing Software Reliability

1. Size and Complexity of Code

- The number of LOC in a software system is a measure of its size.
- Large software systems with hundreds of thousands of LOC tend to have more faults than smaller systems.
- The likelihood of faults in a large system, because of more module interfaces, is higher.
- The more number of conditional statements the code contains, the more complex the system is considered to be.
- Removing faults from a large system, new faults may be introduced.

Factors Influencing Software Reliability

2. Characteristics of Development Process

- By developing a system under a larger quality control umbrella in the form of embracing the above software engineering techniques and tools, the number of remaining faults in software systems can be reduced.
- Code review techniques have been developed to detect design and implementation faults.
- Test tools are available to assist programmers in their unit-level and system-level testing.
- Formal methods, such as SDL (Specification and Description Language) and UML (Unified Modeling Language), are used to specify the requirements of complex, real-time systems.

Factors Influencing Software Reliability

3. Education, Experience, and Training of Personnel

- Lack of desired skills in personnel can cause a system to have a larger number of faults.

4. Operational Environment:

- Detection of faults remaining in a software system depends upon a test engineer's ability to execute the system in its actual operational environment.
- If a test engineer fails to operate a system in the same manner the users will do, it is very likely that faults will go undetected.
- Therefore, test engineers must understand the ways the users will operate a system.

The SQA Plan

- The SQA plan provides a road map for instituting software quality assurance.
- Developed by the SQA group and the project team, the plan serves as a template for SQA activities that are instituted for each software project.
- Initial sections describe
  - the purpose and scope of the document
  - and indicate those software process activities that are covered by quality assurance.
- The Management section of the plan describes
  - SQA's place in the organizational structure;
  - SQA tasks and activities and their placement throughout the software process;
  - and the organizational roles and responsibilities relative to product quality.

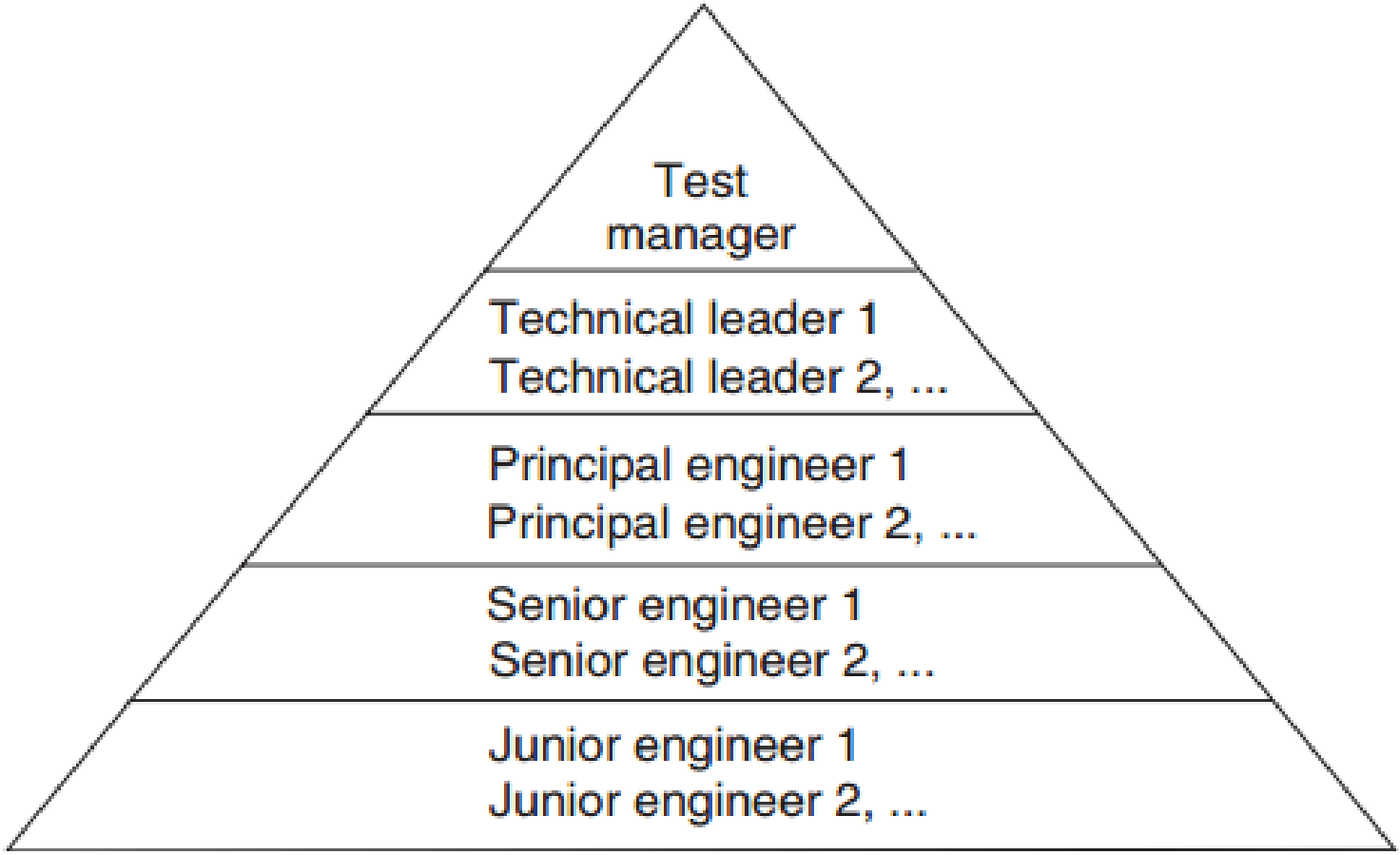
The SQA Plan

- The Documentation section describes(by reference) each of the work products produced as part of the software process. These include:
  - Project documents (e.g., project plan)
  - Models (e.g., ERDs, class hierarchies)
  - Technical documents (e.g., specifications, test plains)
  - User documents (e.g., help files)
- Standards, Practices, and Conventions lists all applicable standards/practices that are applied during the software process.
- The Reviews and Audits section of the plan identifies the reviews and audits to be conducted by the software engineering team, the SQA grond the customer.

SQA Group

- A software quality assurance group has a larger role in ensuring conformance to the best development practices throughout the organization.
- The software quality assurance group should have sufficient authority, power, and standing to work with the entire organization in defining processes and dictating peers to follow the processes
- The group proactively works to drive process improvement initiatives across the organization. The group makes an effort to understand the best practices followed around the world through systematic benchmarking and to amalgamate those practices with the one existing within the organization.

SQA Group


Test manager
Technical leader 1
Technical leader 2, ...
Principal engineer 1
Principal engineer 2, ...
Senior engineer 1
Senior engineer 2, ...
Junior engineer 1
Junior engineer 2, ...

SQA Group

- Test Engineer
  - The administrative responsibilities of a manager include managing the budget, hiring personnel, assigning tasks, training personnel, developing their career, and reviewing their performance.
  - The technical responsibilities of a test manager include
    - developing a test plan, executing a test plan, and handling crises;
    - developing, following, and improving test processes;
    - and defining, collecting, and analyzing metrics.

SQA Group

- Technical Leader
  - A technical leader often needs to coordinate the tasks of many engineers working on complex projects.
  - At this level the engineer should have technical testing skills of a principal engineer as well as negotiation skills, process and project management skills, and communication skills.
  - The technical leader provides the strategic direction for software testing.

SQA Group

- Principal Engineer
  - The principal engineers are test specialists responsible for
    - test planning, test automation, test environment planning, test tool development, procurement of test equipment, performance modeling, reliability engineering, and business acceptance testing.

SQA Group

- Senior Engineer
  - The senior engineers design, develop, and execute test cases.
  - They design and set up test laboratories and environments.
  - Senior engineers assist software developers in reproducing known defects.
  - They participate in test plan review meetings and support maintenance of test laboratories, automated test suites, and test tools.

SQA Group

- Junior Engineer
  - New, inexperienced test engineers are put at a junior level.
  - They gain experience by assisting the senior and principal engineers in charge of test execution, setting up of test beds, and developing scripts for test automation.