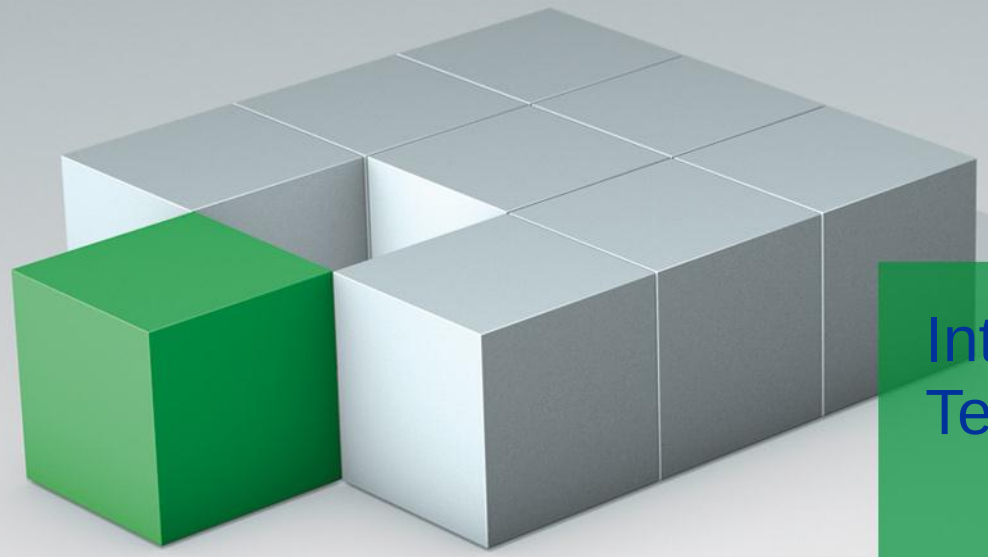
Quality Governance
Transforming IT into Business Value

SQS
SOFTWARE QUALITY SYSTEMS



International Software Testing Qualifications Board

Foundation Course for Software Testing

SQS Group Limited

ISTQB coverage

- Session 1- Fundamentals of testing (K2)
- Session 2- Software development models (K2)
- Session 3- Static techniques (K2)
- Session 4- Test design techniques (K3)
- Session 5- Test organization (K3)
- Session 6- Tool support for testing (K2)

ISTQB Levels of knowledge

Cognitive levels are given for each session

- **K1 : remember, recognize, recall; - 50% questions**
- **K2 : understand, explain, give reasons, compare, classify, categorize, give examples, summarize; - 30% questions**
- **K3 : apply, use ; – 20% questions**

Session 1

Fundamentals of testing (K2)

Fundamentals of testing

Session coverage

- Why is testing necessary (K2)
- What is testing (K2)
- General testing principles (K2)
- Fundamental test process (K1)
- The psychology of testing (K2)

Fundamentals of testing

Session coverage

- **Why is testing necessary (K2)**
- What is testing (K2)
- General testing principles (K2)
- Fundamental test process (K1)
- The psychology of testing (K2)

Standard Testing terms

- Testing terminology is contained in the ‘ISTQB glossary of terms’ provided with the handout

Why is testing necessary?

- Who develops software?
- Is software developed free from defects?
- Why?
- Is it necessary to find defects ?
- Why? What is the cost of defects?
- How do we find these defects?
- Multiplication of defects reduced
- Defects are less costly to correct the early they are found

Testing is necessary !!

- Software is not defect free
- Defects cause failures
- Unreliable software can cause failures
- Failures have associated costs like loss of business
- Testing helps to find defects and learn about reliability of software

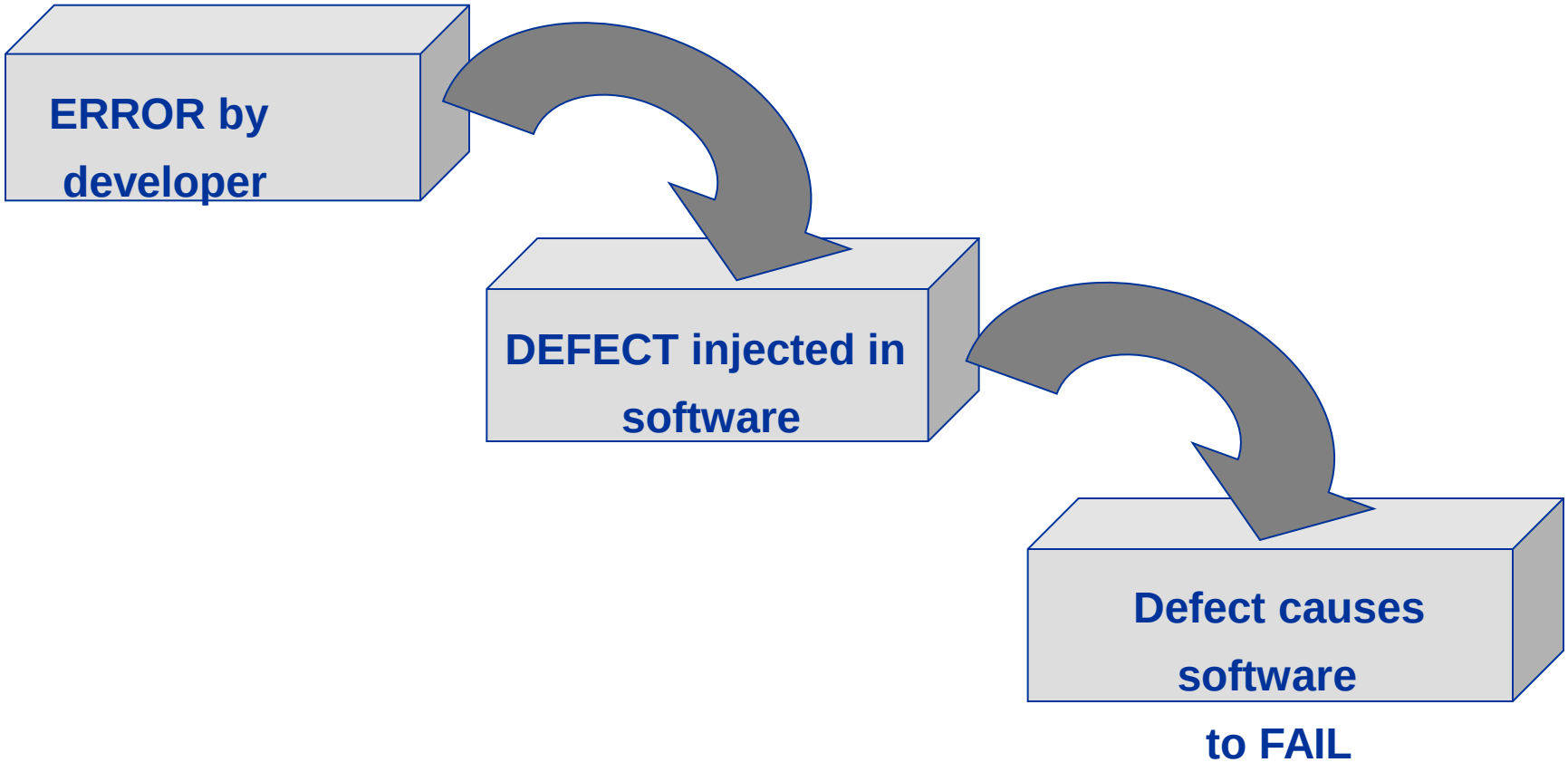
Software systems context

- Software systems are an increasing part of life
- Software does not work as expected
- Incorrect software leads to many problems

Causes of S/W defects

- Human action producing an incorrect result : **error/mistake**
- Errors are inevitable
- Manifestation of error in software: **defect**
- Defects can occur in the source code, documentation – specifications, manuals, etc)
- Defects are also known as *bugs*
- Deviation of the software from its expected delivery or service: **failure**
- Failures are caused by defects in the software

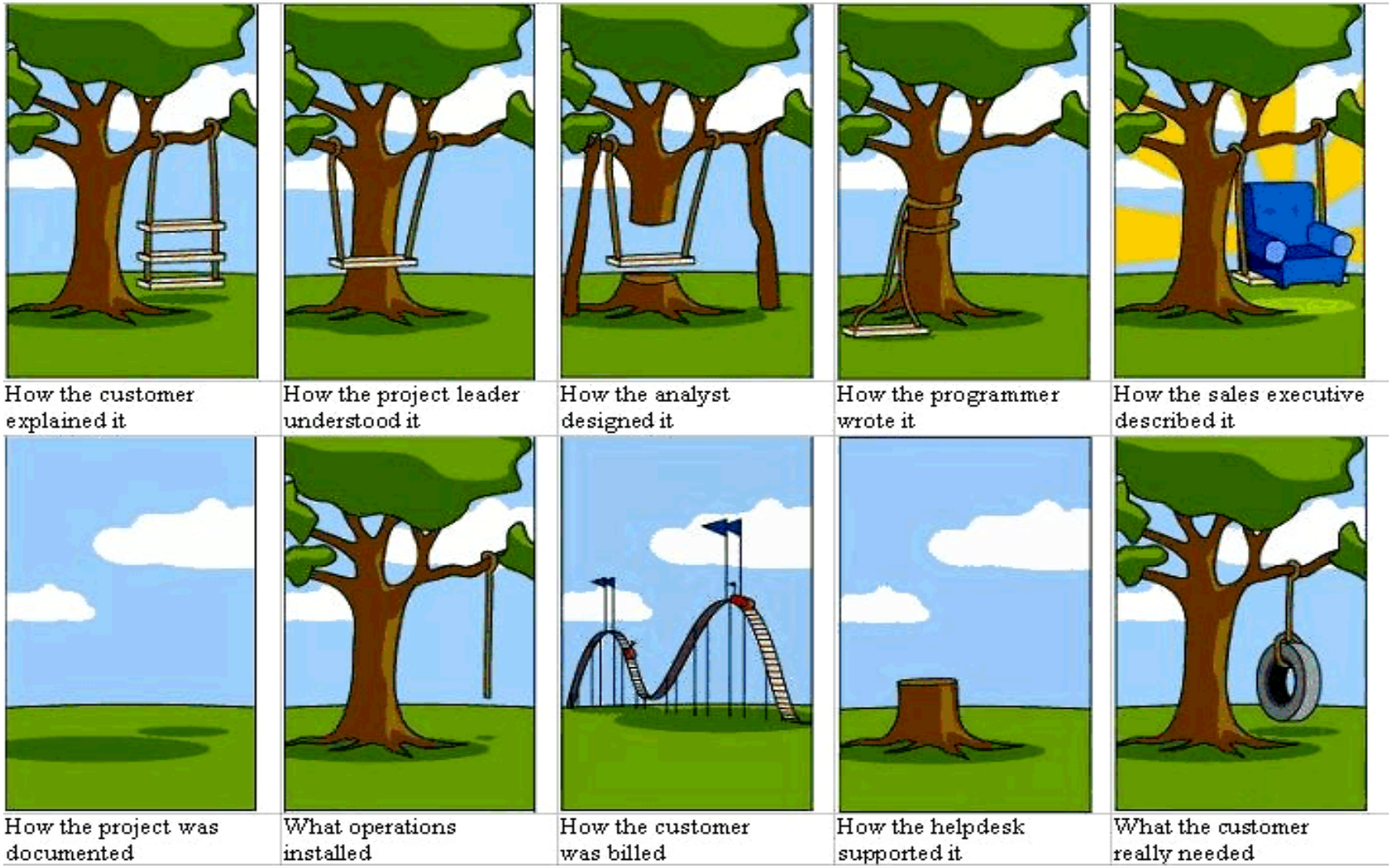
Error → Defect → Failure



Causes of software defects contd..

- Communication
- Software complexity
- Programming errors
- Changing requirements
- Time pressures
- Egos
- Poorly documented code
- Software development tools
- Environmental conditions

Communication or miscommunication



Role of testing in software development, maintenance and operations

- Helps to reduce the risk of problems occurring in an operational environment
- Can contribute to the quality of the software system
- To meet contractual or legal requirements, or industry-specific standards

Testing and quality

- Quality is
  - Meeting customer's needs and requirements
- Testing is
  - Finding out gaps between customer’s requirements and system to be delivered

“Testing improves the quality” How ?

Testing improves quality

- Finding defects and measuring quality in terms of defects
- Building confidence
- Preventing defects
- Reducing risk

How much testing is enough?

- Factors deciding how much to test
  - Level of Risk
    - Technical Risk
    - Business product risk
    - Project Risk
  - Project Constraints
    - Time
    - Budget

Is it possible to test everything?

Exhaustive testing is testing with all combinations of inputs and pre-conditions

"Though not impossible, certainly impractical"

How much testing is enough? Contd..

- Use Risk to decide where to place emphasis while testing
  - Decide what to test first
  - Decide what to test most
  - Decide what not to test

“Prioritize Tests” to do the best testing in the available time

Fundamentals of testing

Session coverage

- Why is testing necessary (K2)
- **What is testing (K2)**
- General testing principles (K2)
- Fundamental test process (K1)
- The psychology of testing (K2)

What is testing?

- **Testing** is a process used to help identify the correctness, completeness and quality of developed computer software

What is testing? Contd…

- *Definition as per ISTQB standard glossary:*
  - The **process** consisting of all **life cycle activities,** both **static** and **dynamic**, concerned with **planning**, **preparatio**n and **evaluation of software products** and related work products to **determine** that **they satisfy specified requirements**, to demonstrate that they are **fit** for **purpose** and to **detect defects**

Objectives of testing

- Finding defects early in the SDLC
- Gaining confidence about the level of quality and providing information
- Preventing defects

Methods of testing

- Static testing
  - Non-execution testing method
- Dynamic testing
  - Test execution testing method

Testing and debugging

- **Testing** (done by tester)
  - Show failures that are caused by defects
- **Debugging** (done by developer)
  - Development activity that identifies the cause of a defect, repairs the code, and checks that the defect has been fixed correctly

Fundamentals of testing

Session coverage

- Why is testing necessary (K2)
- What is testing (K2)
- **General testing principles (K2)**
- Fundamental test process (K1)
- The psychology of testing (K2)

General testing principles

- Testing shows presence of defects
- Exhaustive testing is impossible
- Early testing
- Defect clustering
- Pesticide paradox
- Testing is context dependent
- Absence-of-errors fallacy

Principle 1

Testing shows presence of defects

- Testing can show that defects are present, but cannot prove that there are no defects
- Testing reduces the probability of undiscovered defects remaining in the software but, even if no defects are found, it is not a proof of correctness

Principle 2

Exhaustive testing is impossible

- Testing everything (all combinations of inputs and preconditions) is not feasible except for trivial cases.

  Instead of exhaustive testing, we use risk and priorities to focus testing efforts

Principle 3

Early testing

– Testing activities should start as early as possible in the software or system development life cycle, and should be focused on defined objectives

Principle 4

Defect clustering

– A small number of modules contain most of the defects discovered during pre-release testing, or show the most operational failures

Principle 5

Pesticide paradox

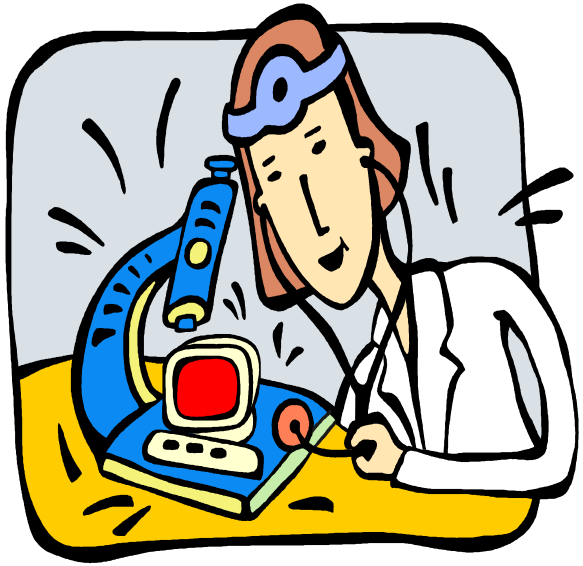
- If the same tests are repeated over and over again, eventually the same set of test cases will no longer find any new bugs

- To overcome this “pesticide paradox”, the test cases need to be regularly reviewed and revised, and new and different tests need to be written to exercise different parts of the software or system to potentially find more defects

Principle 6

Testing is context dependent

- Testing is done differently in different contexts. For example, safety-critical software is tested differently from an e-commerce site



Principle 7

Absence-of-errors fallacy

– Finding and fixing defects does not help if the system built is unusable and does not fulfill the users' needs and expectations

Fundamentals of testing

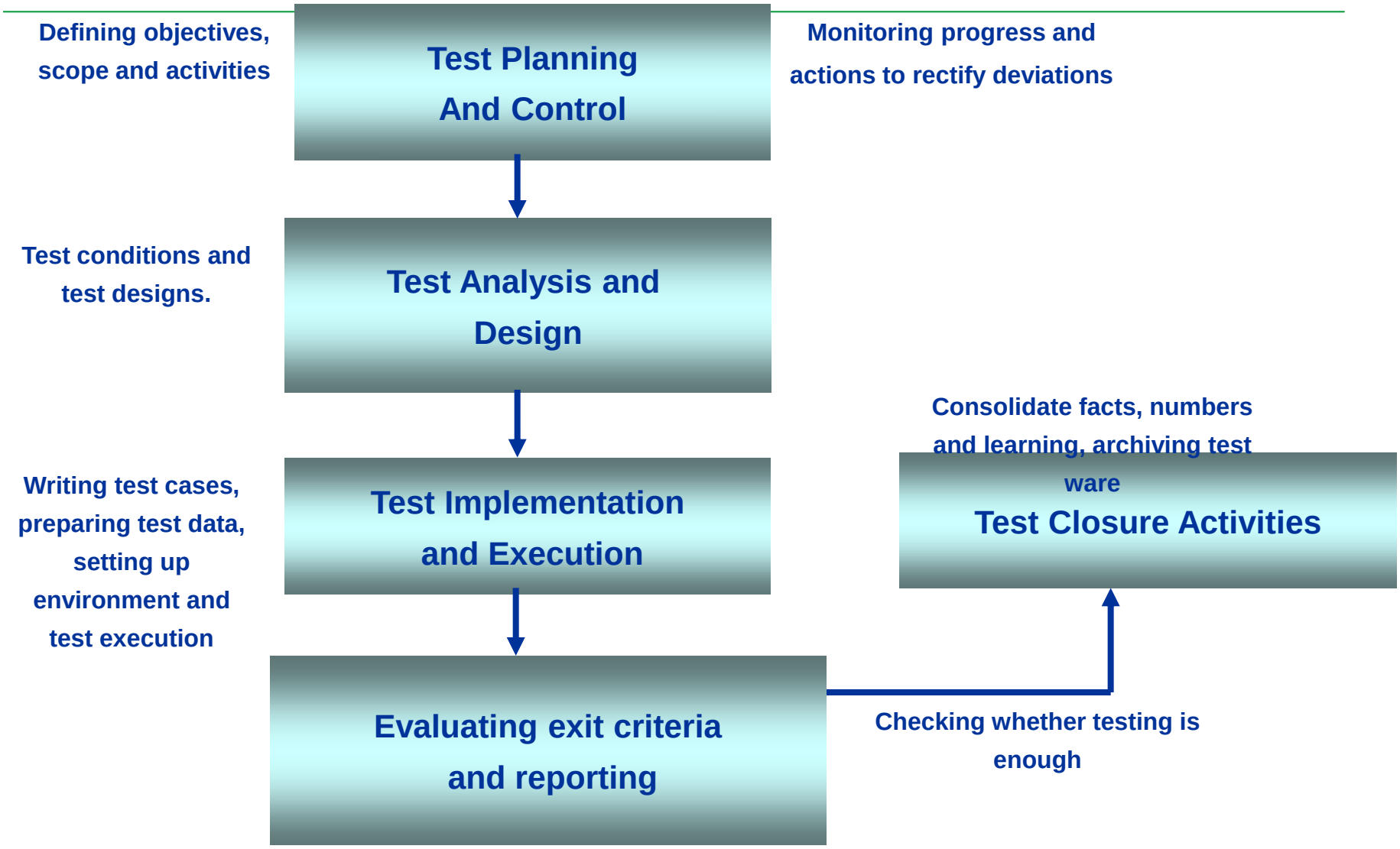
Session coverage

- Why is testing necessary (K2)
- What is testing (K2)
- General testing principles (K2)
- **Fundamental test process (K1)**
- The psychology of testing (K2)

Fundamental test process

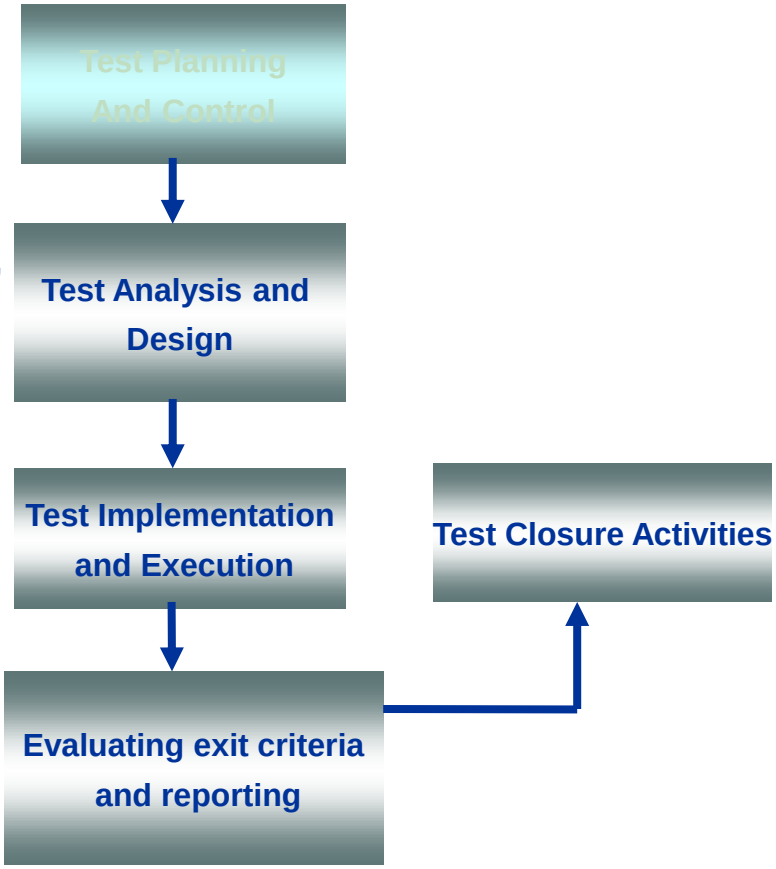
- Test planning and control
- Test analysis and design
- Test implementation and execution
- Evaluating exit criteria and reporting
- Test closure activities

Fundamental test process contd..



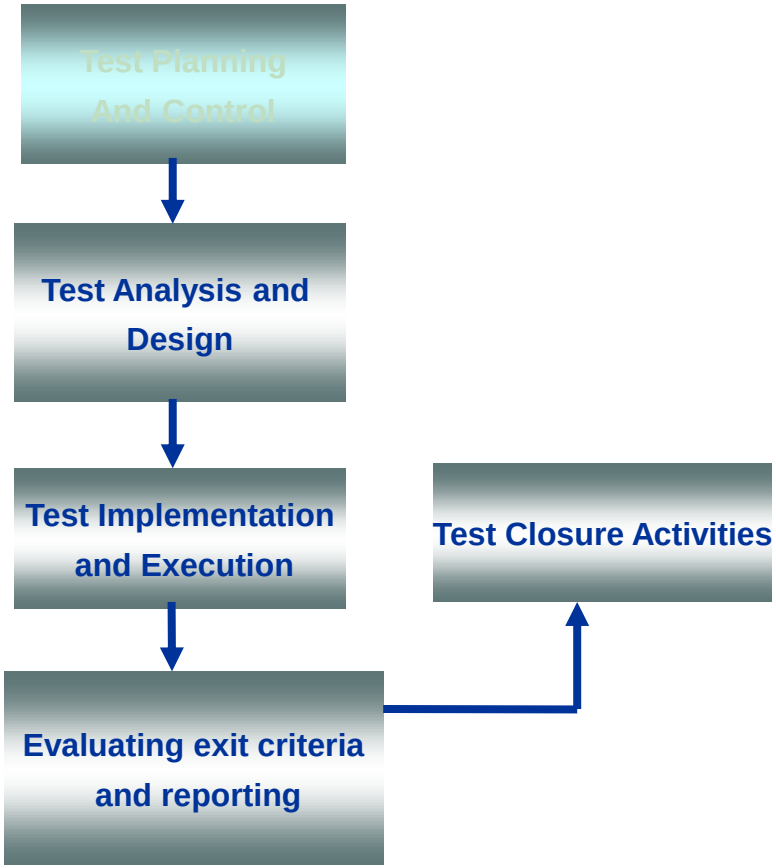
Test Planning

- Defining testing objective and scope
- Identifying risks and constraints
- Defining test policy and/or the test strategy
- Deciding test approach (techniques, test items, coverage, identifying and interfacing
- Defining entry and exit criteria for testing
- Identifying teams and skills involved in testing
- Identifying test resources (E.g. people, test environment, hardware, software, etc)
- Scheduling testing activities/tasks



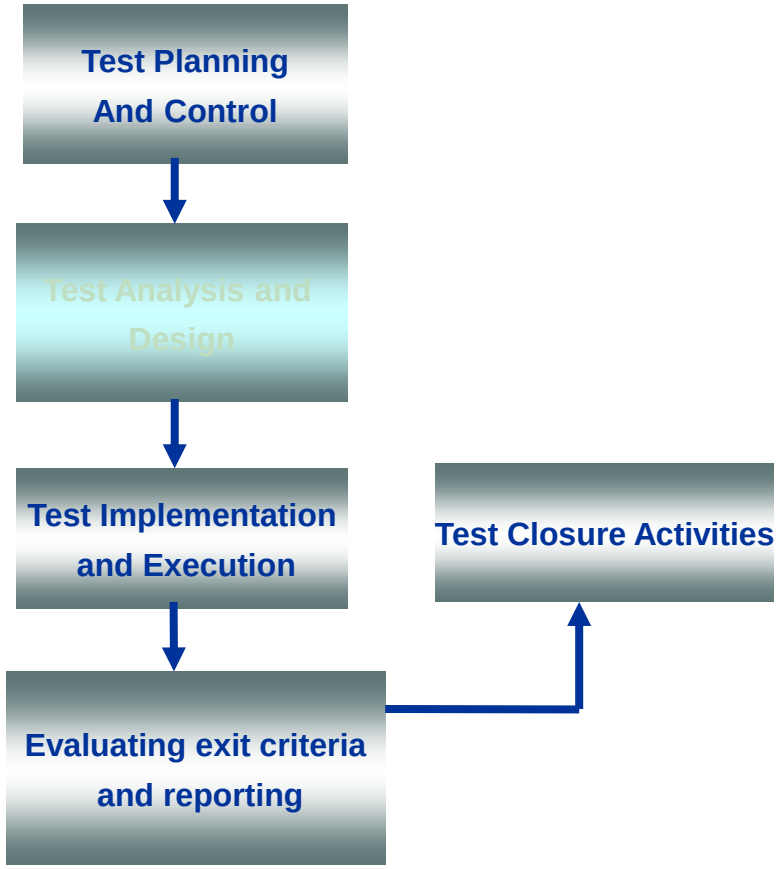
Test control

- Measuring and analyzing results
- Monitoring and documenting progress,
- test coverage and exit criteria
- Initiation of corrective actions
- Making decisions



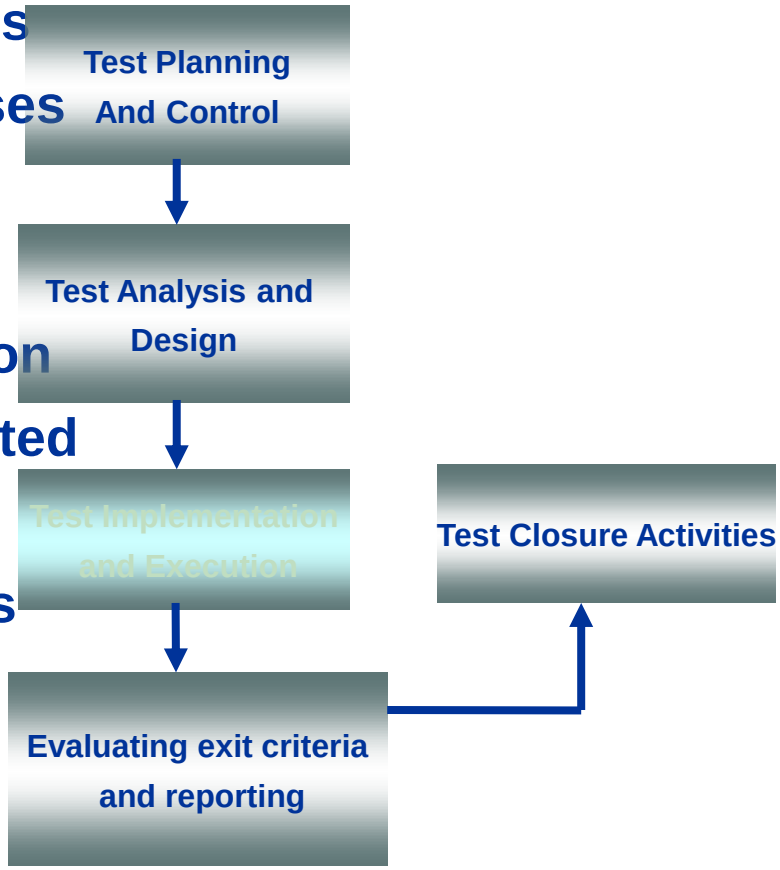
Test analysis and design

- Reviewing the test basis
- Identifying test conditions or test requirements
- Designing test conditions
- Evaluating testability
- Designing the test environment set-up



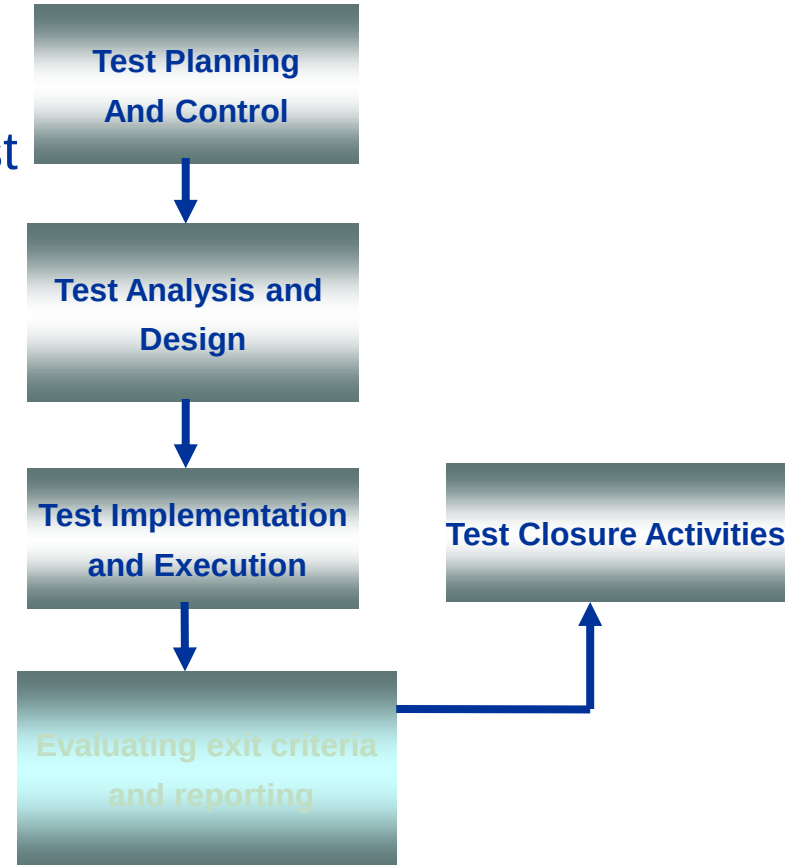
Test implementation and execution

- Developing and prioritizing test cases
- Creating test suites from the test cases
- Verifying the test environment
- Executing test cases
- Logging the outcome of test execution
- Comparing actual results with expected results
- Reporting discrepancies as incidents and analyzing them
- Repeating test activities as result of action taken (confirmation/retesting) and regression



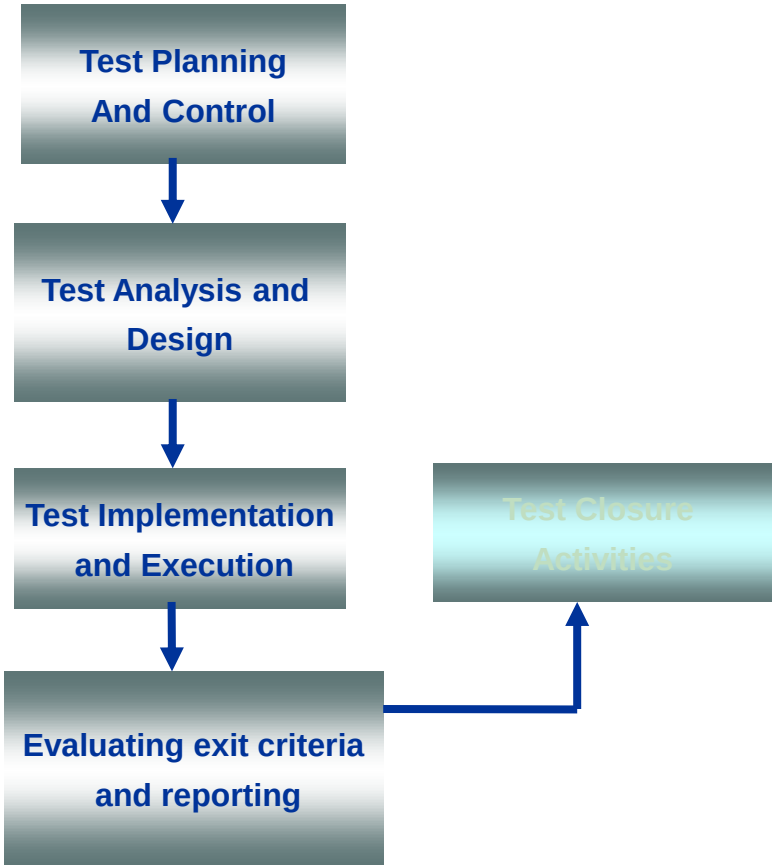
Evaluating exit / completion criteria and reporting

- Check the test logs against the exit /completion criteria defined in the test plan
- Assess if more tests are needed or if the exit criteria specified should be changed
- Writing a test summary report for stakeholders



Test closure activities

- Checking which planned deliverables have been delivered
- Finalizing and archiving testware
- Handover of testware
- Analyzing lessons learned



Fundamentals of testing

Session coverage

- Why is testing necessary (K2)
- What is testing (K2)
- General testing principles (K2)
- Fundamental test process (K1)
- **The psychology of testing (K2)**

Psychology of testing

- **Why test?**
  - Build confidence in software under test
  - To find defects
  - Prove that the software conforms to user requirements/ functional specifications
  - Reduce failure costs
  - Assess quality of software
- **Can testing prove software is correct?**
  - Not possible to prove system has no defects
  - Only possible to prove system has defects

Psychology of testing

- The purpose of testing is to build confidence that the system is working
- But purpose of testing is also to find defects
- Finding defects destroys confidence
- **So, purpose of testing is to destroy confidence?**

Paradox of testing

The best way to build confidence is to try and destroy it

Should programmers test their own code?

- Contribute to specification of test cases due to their technical understanding of software
- Execute tests on their own code since it is cheaper and defect fixing is faster with less effort

Levels of independence

"Independent testing is more effective"

Low

High

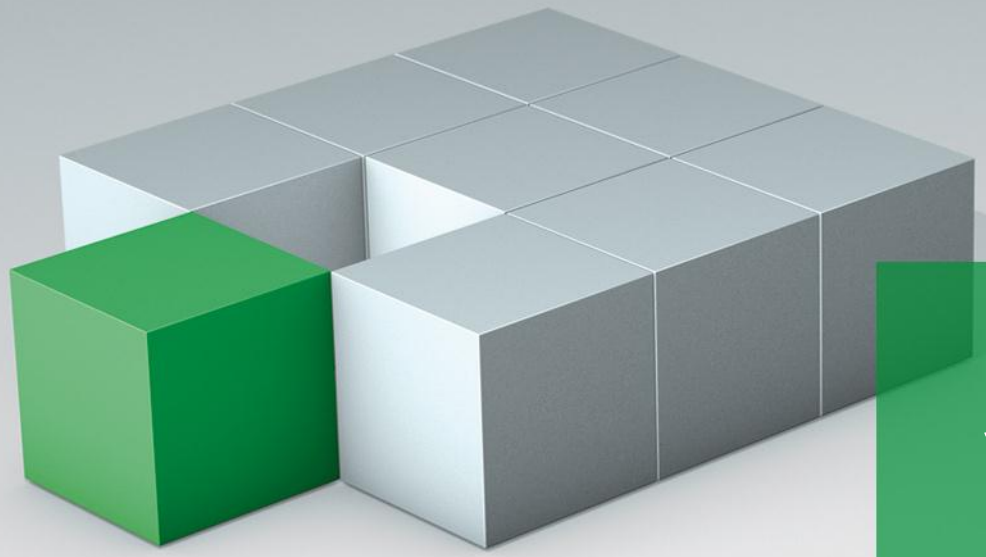
- **Tests designed by the person(s)** who wrote the software under test (low level of independence)
- **Tests designed by another person(s)** (E.g. From the development team)
- **Tests designed by a person(s)** from a different organizational group (E.g. An independent test team)
- **Tests designed by a person(s)** from a different organization or company (i.e. Outsourcing or certification by an external body)

Communication

- Testing perceived as a destructive activity
- Testers generally bring “bad /unwanted news” or “criticism”
- Need to communicate defect information to developers carefully
- Good interpersonal skills are needed to communicate factual information about defects, progress and risks in a constructive way
- Testers should maintain a good relationship with developers

“ A good tester is not the one who finds the most defects, but the one who manages to get most defects fixed”

Quality Governance
Transforming IT into Business Value



SQS India

Thank you for your attention