UECS3383 Software Quality Assurance

Lecture 01 – Introduction to Software Quality Assurance (SAQ)

Introduction to Software Quality?

- **Chapter Outline:**
  - What is software?
  - Categories of software?
  - Software errors, faults and failures
  - Classification of the causes of software errors
  - Nine Areas of Software Errors
  - Verification and Validation
  - What is quality?
  - What is software quality?
  - Software quality – definition
  - Software quality assurance – definition and objectives
  - Software quality assurance and software engineering

What is Software?

- IEEE Definition – Software is a set of computer programs, procedures, and possibly associated documentation and data pertaining to the operation of a computer system.
- Software consists of carefully-organized instructions and codes written by programmers in any of the various special computer languages and may come in several forms: single programs like script interpreters, packages containing a compiler, linker, and other tools; and large suites.

Categories of Software

- **System software** is responsible for controlling, integrating, and managing the individual hardware components of a computer system so that other software and the users of the system see it as a functional unit without having to be concerned with the low-level details.
- **Application software** handles multitudes of common and specialized tasks a user wants to perform, such as accounting, communicating, data processing, word processing etc.

Terms for Software Failures

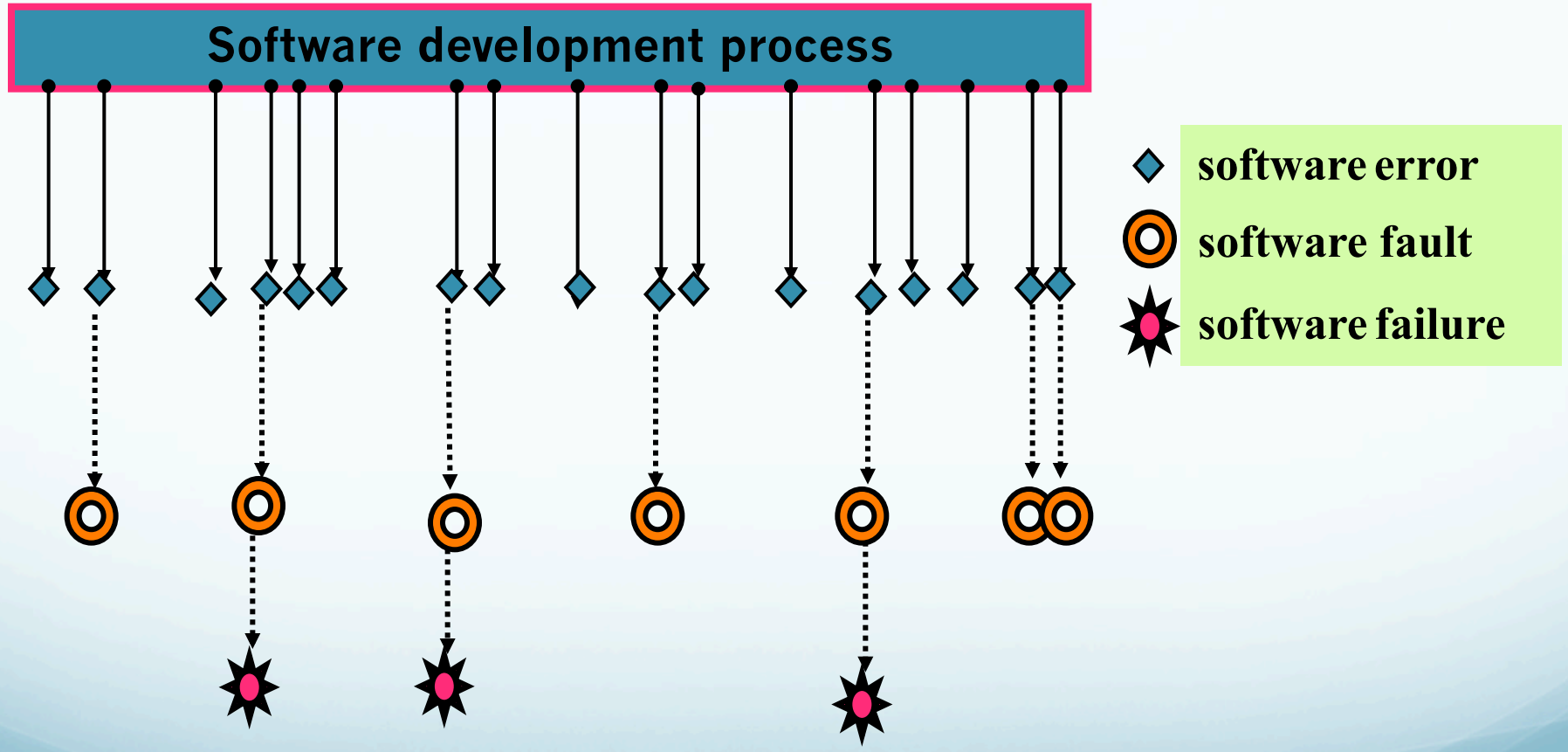
- Depending on where you're employed as a software tester, you will use different terms to describe what happens when software fails:

• Defect	• Variance
• Fault	• Failure
• Problem	• Inconsistency
• Error	• Feature
• Incident	• Bug
• Anomaly	

Software: Error, Fault & Failure

- **Error** – An error is a state of the system. In the absence of any corrective action by the system, an error state could lead to a failure which would not be attributed to any event subsequent to error
- **Fault / Defect** – A fault is the adjudged cause of an error
- **Failure** – a failure is said to occur whenever the external behavior of a system does not confirm to that prescribed in the system specification

Software: Errors, Faults & Failures



Nine Areas of Software Errors

1) Faulty requirements definition
2) Client-developer communication failures
3) Deliberate deviations from software requirements
4) Logical design errors
5) Coding errors
6) Non-compliance with documentation and coding instructions
7) Shortcomings of the testing process
8) User interface and procedure errors
9) Documentation errors

Causes of Software Defects

- **A human being can make an error (mistake)**
- **Errors produce defects**
  - Defects are faults / bugs in the program code, or in a document
- **If a defect in code is executed, that might cause a failure:**
  - Fail to do what it should do
  - Do something it shouldn't
- **Organizational factors**
  - Inefficient team communication
  - Incomplete data specifications
  - Unclearly defined requirements
  - Incorrect project documentation

Causes of Software Defects

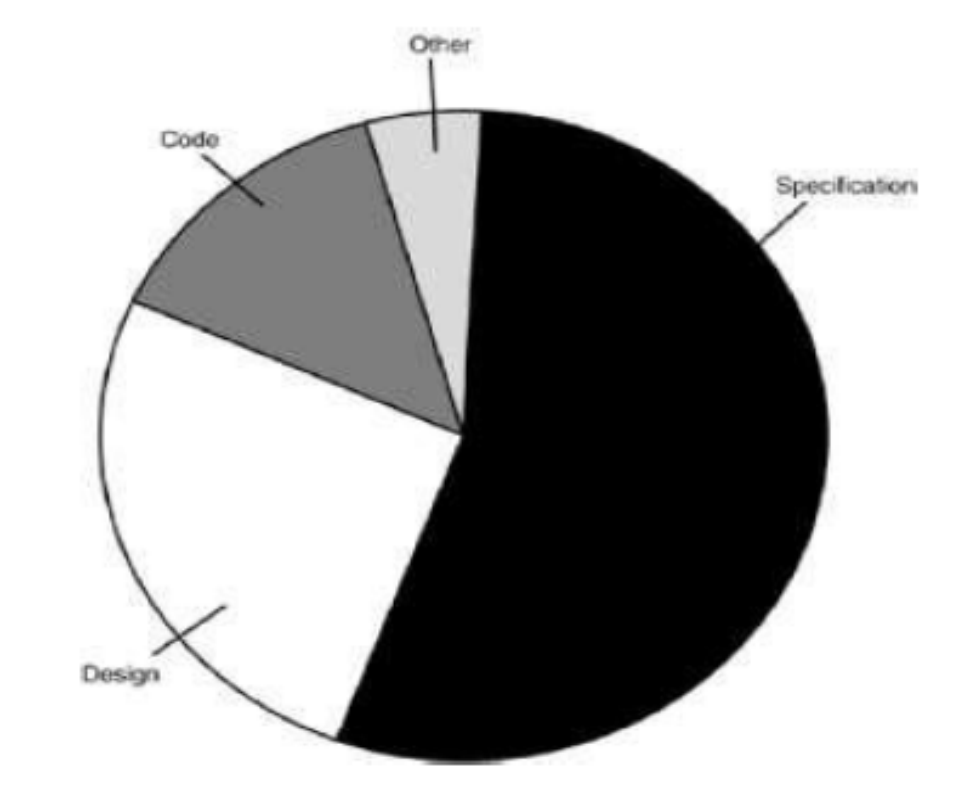
- **The human factor**
  - Humans make mistakes
  - Poor training
  - Time pressure
  - Code complexity
  - Complexity of infrastructure
  - Changing technologies
- **Environmental conditions** **(These can change the hardware conditions)**
  - Radiation
  - Magnetism
  - Electronic fields
  - Pollution, etc.

Software Bugs

- Software bug occurs when one or more of the following rules are true:
  - The software doesn't do something that the product specification says it should do.
  - The software does something that the product specification says it shouldn't do.
  - The software does something that the product specification doesn't mention.
  - The software doesn't do something that the product specification doesn't mention but should.
  - The software is difficult to understand, hard to use and slow.

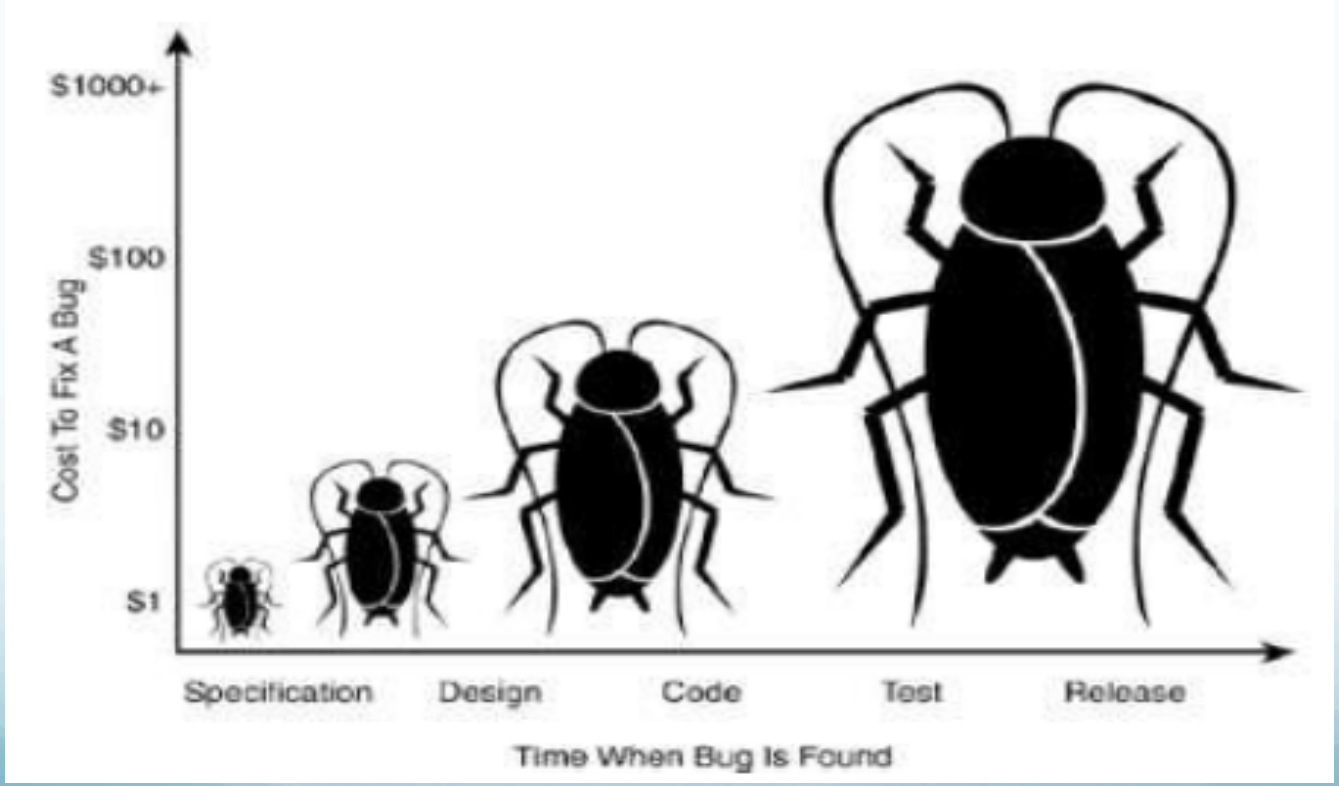
Why Do Bugs Occur?

- Bugs are caused for numerous reasons, but, in this sample project analysis, the main cause can be traced to the specification.



Cost of Bugs

- An example of how the cost of fixing these bugs can grow over time



What is Quality?

- **ISO 9000, 2nd edition, year 2000 Definition :**
  - The degree to which a set of inherent characteristics fufills requirements
  - Quality can be used with such adjectives as poor, good, or excellenct.
  - Inherent as opposed to assigned, means existing inside something, as a permanent characteristics
  - Theree key elements:
    - Requirements: ...customer in a made-to-order scenario or by product specifications in a commercial off-the-shelf scenario
    - Characteristics: ....capability of the deliverables or, the robustness 9fitness) of the product
    - Degree: ....quality is a continum, beginning with zero and moving towards, perhaps, infinity...

Quality

- **Quality** is a subjective term. Software quality will depend on who the customer is and the customer's involvement in the overall software project.
- Customer can be defined as the end users, customer acceptance testers, customer contract officers, customer management, the development organization's management, accountants, testers, salespeople, future software maintenance engineers, stockholders, and so on.
- Each type of customer who will be evaluating the value of the software application will have their own way of defining and measuring quality.

Definition of Good Design

- Good software design is the design that incorporates the latest but proven technology, has a scalable structure, facilitates easy integration with other systems, and is flexible to allow incorporating changes in requirements.
- Businesses always keep on expanding, thus any system deployed should also scale up nicely.
- the application should be easily integratable with these myriad systems.
- As Requirements keep changing very often ,the design should be flexible enough to take care of these changes in requirements

Definition of Good Code

- Good code is code that works as intended, is reasonably defect-free, and is readable and maintainable.
- For object-oriented programming, here are some typical ideas to consider in setting rules/standards:
  - ❖ Minimize or eliminate use of global variables.
  - ❖ Use descriptive function and method names—use both uppercase and lowercase, avoid abbreviations, and use as many characters as necessary to be adequately descriptive. Be consistent in naming conventions.
- Use descriptive variable names—use both uppercase and lowercase, avoid abbreviations, and use as many characters as necessary to be adequately descriptive. Be consistent in naming conventions.

Definition of Good Code ... (continued)

- Function and method sizes should be minimized. Less than 100 lines of code is good; less than 50 lines is preferable.
- Function descriptions should be clearly spelled out in comments preceding a function's code.
- Organize code for readability.
- Use white spaces generously—vertically and horizontally.
- Each line of code should contain 70 characters max.
- One code statement per line.
- Coding style should be consistent throughout a program (e.g., use of brackets, indentations, naming conventions).
- In adding comments, err on the side of too many rather than too few comments; a common rule of thumb is that there should be at least as many lines of comments (including header blocks) as lines of code.

Definition of Good Code ... (continued)

- Documentation of the overall program function and flow or if possible a separate flowchart and detailed program documentation.
- Make extensive use of error-handling procedures and status and error logging.
- To minimize complexity and increase maintainability, avoid too many levels of inheritance in class hierarchies (relative to the size and complexity of the application). Minimize use of multiple inheritances, and minimize use of operator overloading (note that the java programming language eliminates multiple inheritance and operator overloading).
- Make liberal use of exception handlers.

What is Software Quality?

- **IEEE Definition :**
  - The degree to which a system, component, or process meets specified requirements
  - The degree to which a system, component, or process meets customer or user needs or expectations
  - A planned and systematic pattern of all actions necessary to provide adequate confidence that an item or product conforms to established technical requirements
  - A set of activities designed to evaluate the process by which the products are developed or manufactured. Contrast with: quality control.

What is Software Quality?

- **Pressman's Definition:**
  - Software quality measures how well software is designed (quality of design), and how well the software conforms to that design (quality of conformance)
  - Software quality is conformance to explicitly stated functional and performance requirements, explicitly documented development standard, and implicitly characteristics that are expected of all professionally developed software.
- Whereas quality of conformance is concerned with implementation, quality of design measures how valid the design and requirements are in creating a worthwhile product

What is Software Quality?

- **Steve McConnell's "Code Complete" defines two types of quality (internal and external) characteristics:**
  - **External : Those parts of a product that face its users**
    - Correctness, Usability, Efficiency, Reliability, Integrity, Adaptability, Accuracy, Robustness
  - **Internal : Not apparent to the user**
    - Maintainability, Flexibility, Portability, Reusability, Testability, Understandability

Internal vs External Characteristics Dependencies

How focusing on the factor below affects the factor to the right	Correctness	Usability	Efficiency	Reliability	Integrity	Adaptability	Accuracy	Robustness
Correctness	↑		↑	↑			↑	↓
Usability		↑				↑	↑	
Efficiency	↓		↑	↓	↓	↓	↓	
Reliability	↑			↑	↑		↑	↓
Integrity			↓	↑	↑			
Adaptability					↓	↑		↑
Accuracy	↑		↓	↑		↓	↑	↓
Robustness	↓	↑	↓	↓	↓	↑	↓	↑

Helps it ↑

Hurts it ↓

What is Software Quality?

- Quality is a complex concept – it means different things to different people, and it is highly context dependent.
- Kitchenham and Pfleeger discuss five views of quality in a comprehensive manner of:
  - (1) Transcendental View
  - (2) User View
  - (3) Manufacturing View
  - (4) Product View
  - (5) Value-Based View

Different Views of Software Quality

- **Transcendental view**
  - Quality as something that can be recognized but is difficult to define
  - This view is not specific to software quality but has been applied in other complex areas of everyday life
- **User view**
  - It perceives quality as **fitness for purpose**
  - Question asked "Does the product satisfy user needs and expectations?"
- **Manufacturing view**
  - Quality is understood **as conformance to the specification**
  - Quality level of a product is determined by the extent to which the product meets its specifications
- **Product view**
  - Quality is viewed as tied to the **inherent characteristics of the product**
  - Inherent characteristics: internal qualities determines its external qualities
- **Value-Based view**
  - Quality is depends on the **amount a customer is willing to pay for it**

Software Quality vs Quality Software

Software Quality	Quality Software
It is a body of knowledge that describes "**what** must be done" and "**how** it must be done".	It exhibits ALL the functional capabilities and non-functional attributes that ensure it can be put to ALL its indended uses with the "least" effort, inconvenience and resource cost to user.
It embodies a "specification" and an "implementation of processes" for realizing quality software	E.g. Software should function correctly with respect to a specification that has been predefined by the client who has commissioned the development of the software
N/A	Software must possess attributes other than correctness with respect to a specification before it can be classified as a quality software

Verification and Validation

- **Verification**
  - Help in evaluating a software system by determining whether the product of a given development phase satisfies the requirements established before the start of that phase
  - Activities that check the correctness of a development phase are called verification activities
  - Aim at confirming a product is builds correctly
- **Validation**
  - Help in confirming a product meets its intended **use**.
  - Aims at confirming a product meets its customer's expectations
  - Focus on final product, which is extensively tested from the customer point of view
  - It establishes whether the product meets overall expectations of the users
  - Aim at confirming if one is building the correct product

What is Software Quality Assurance?

- **IEEE Definition:**
  - A planned and systematic pattern of all actions necessary to provide adequate confidence that an item or product conforms to established technical requirements
  - A set of activities designed to evaluate the process by which the products are developed or manufactured. Contrast with quality control

What is Software Quality Assurance?

- **SQA Expanded Definition:**
  - A systematic, planned set of actions necessary to provide adequate confidence that the software development process or the maintenance process of a software system product conforms to established functional technical requirements as well as with the managerial requirements of keeping the schedule and operating within the budgetary confines.
- Software quality assurance is a planned and systematic program of activities designed to ensure that a system has the desired characteristics

Software Quality Assurance (SQA)

- **Software Quality Assurance** (**SQA**) consists of a means of monitoring the software engineering processes and methods used to ensure quality.
- SQA encompasses the entire software development process, which includes processes such as:
  - Requirements Definition
  - Software Design
  - Coding
    - Source Code Control
    - Code Reviews
  - Change Management
  - Configuration Management
  - Testing
  - Release Management
  - Product Integration
- SQA is organized into goals, commitments, abilities, activities, measurements, and verifications.

SQA Activities and Processes

- SQA is a set of activities for ensuring quality in software engineering processes (that ultimately result in quality in software products). The activities establish and evaluate the processes that produce products.

- Processes could be:
  - Software Development Methodology
  - Project Management
  - Configuration Management
  - Requirements Development/Management
  - Estimation
  - Software Design
  - Testing
  - *etc*

SQA Activities and Processes

- Once the processes have been defined and implemented, Quality Assurance has the following responsibilities:
  - identify weaknesses in the processes
  - correct those weaknesses to continually improve the process
- SQA basically focuses on Process of the projects. It is a Prevention oriented. The *Breadth of SQA is* Organization wide. It relates to all products that will ever be created by a process or processes.
- The activities of SQA are as follows:
  - Process Definition and Implementation
  - Audits
  - Training

Quality Assurance vs Quality Control

Parameters	Quality Assurance	Quality Control
Definition	QA is a set of activities for ensuring quality in the processes by which products are developed.	QC is a set of activities for ensuring quality in products. The activities focus on identifying defects in the actual products produced. It is a set of activities designed to evaluate the quality of a developed and manufactured product
Objective	Its objective is to minimize the cost of guaranteeing quality by "preventing", "detect" and "correcting" errors to reduce the rate of products/services that do not qualify	The objective is to withholding of any product/services that does not qualify
Focus on	QA aims to prevent defects with a focus on the process used to make the product. It is a proactive quality process.	QC aims to identify (and correct) defects in the finished product. Quality control, therefore, is a reactive process.

Quality Assurance vs Quality Control

Parameters	Quality Assurance	Quality Control
Goal	The goal of QA is to improve development and test processes so that defects do not arise when the product is being developed.	The goal of QC is to identify defects after a product is developed and before it's released.
How	Establish a good quality management system and the assessment of its adequacy. Periodic conformance audits of the operations of the system.	Finding & eliminating sources of quality problems through tools & equipment so that customer's requirements are continually met.
What	Prevention of quality problems through planned and systematic activities including documentation.	The activities or techniques used to achieve and maintain the product quality, process and service.
Responsibility	Everyone on the team involved in developing the product is responsible for quality assurance.	Quality control is usually the responsibility of a specific team that tests the product for defects.

Quality Assurance vs Quality Control

Parameters	Quality Assurance	Quality Control
Example	Verification is an example of QA	Validation/Software Testing is an example of QC
Statistical Techniques	Statistical Tools & Techniques can be applied in both QA & QC. When they are applied to processes (process inputs & operational parameters), they are called Statistical Process Control (SPC); & it becomes the part of QA.	When statistical tools & techniques are applied to finished products (process outputs), they are called as Statistical Quality Control (SQC) & comes under QC.

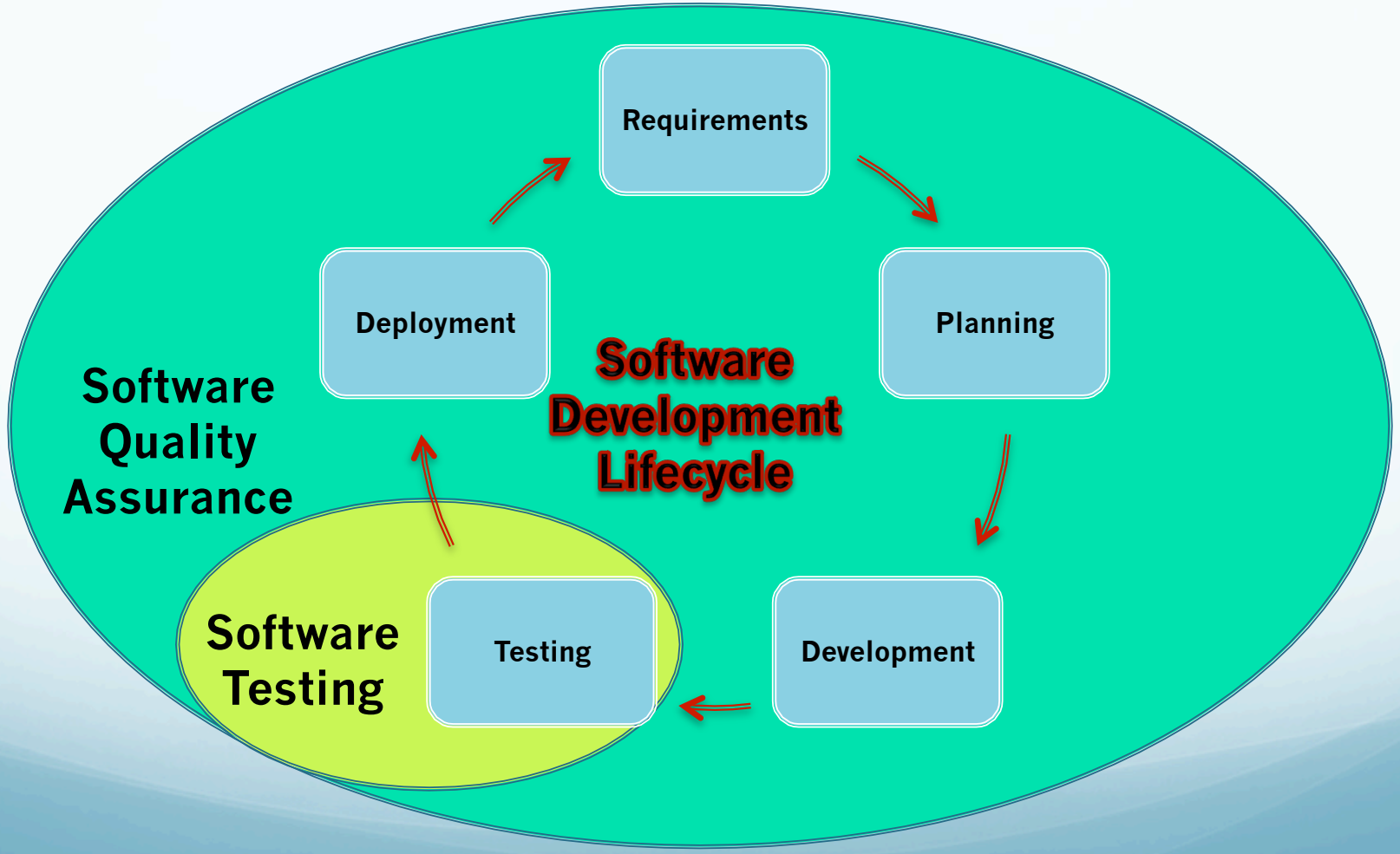
Quality Assurance vs Quality Control

Parameters	Quality Assurance	Quality Control
Definition	QA is a set of activities for ensuring quality in the processes by which products are developed.	QC is a set of activities for ensuring quality in products. The activities focus on identifying defects in the actual products produced. It is a set of activities designed to evaluate the quality of a developed and manufactured product
Objective	Its objective is to minimize the cost of guaranteeing quality by "preventing", "detect" and "correcting" errors to reduce the rate of products/services that do not qualify	The objective is to withholding of any product/services that does not qualify
	Generally, perform **major verification** and **minor validation**	Generally, perform **minor verification** and **major validation**

Quality Assurance vs Quality Control

Quality Assurance	Quality Control
Its objective is to minimize the cost of guaranteeing quality by "preventing", "detect" and "correcting" errors to reduce the rate of products/services that do not qualify	Is a set of activities designed to evaluate the quality of a developed and manufactured product
	The objective is to withholding of any product/services that does not qualify
Generally, perform **major verification** and **minor validation**	Generally, perform **minor verification** and **major validation**

Quality Assurance vs Testing



SQA – Software Development

- The objectives of SQA activities in software development are:
  1) Assuring an acceptable level of confidence that the software will conform to functional technical requirements.
  2) Assuring an acceptable level of confidence that the software will conform to managerial scheduling and budgetary requirements.
  3) Initiation and management of activities for the improvement and greater efficiency of software development and SQA activities.

SQA – Software Maintenance

- The objectives of SQA activities in software maintenance are:
  1) Assuring an acceptable level of confidence that the software maintenance activities will conform to the functional technical requirements.
  2) Assuring an acceptable level of confidence that the software maintenance activities will conform to managerial scheduling and budgetary requirements.
  3) Initiate and manage activities to improve and increase the efficiency of software maintenance and SQA activities.

Techniques for Improving Software Quality

- Software-quality objectives
- Explicit quality-assurance activity
- Testing strategy
- Software-engineering guidelines
- Informal technical reviews
- Formal technical reviews
- External audits

Techniques for Improving Software Quality ..... (continued)

- Improve Development Process
  - Change-control procedures
  - Measurement of results
  - Prototyping
- Setting Objectives
  - Programmers have high achievement motivation. They will work to the objective specified, but they must be told what the objectives are
  - Objectives conflict and it's generally not possible to do well on all of them

The “Ideal” Way....

- Recommended Combination of:
  - Formal inspections of all requirements, all architecture, and designs for critical parts of a system
  - Modeling or prototyping
  - Code reading or inspections
  - Execution testing

The General Principle…

- The General Principle of Software Quality is that improving quality reduces development costs
  - The industry-average productivity for a software product is about 10 to 50 of lines of delivered code per person per day
  - Debugging and associated refactoring and other rework consume about 50 percent of the time on a traditional, naive software-development cycle

SQA and Software Engineering

- Software engineering is the application of a systematic, disciplined, quantifiable approach to development, operation and maintenance of software.
- The characteristics of software engineering, especially its systematic, disciplined and quantitative approach, make software engineering a good environment for achieving SQA objectives.