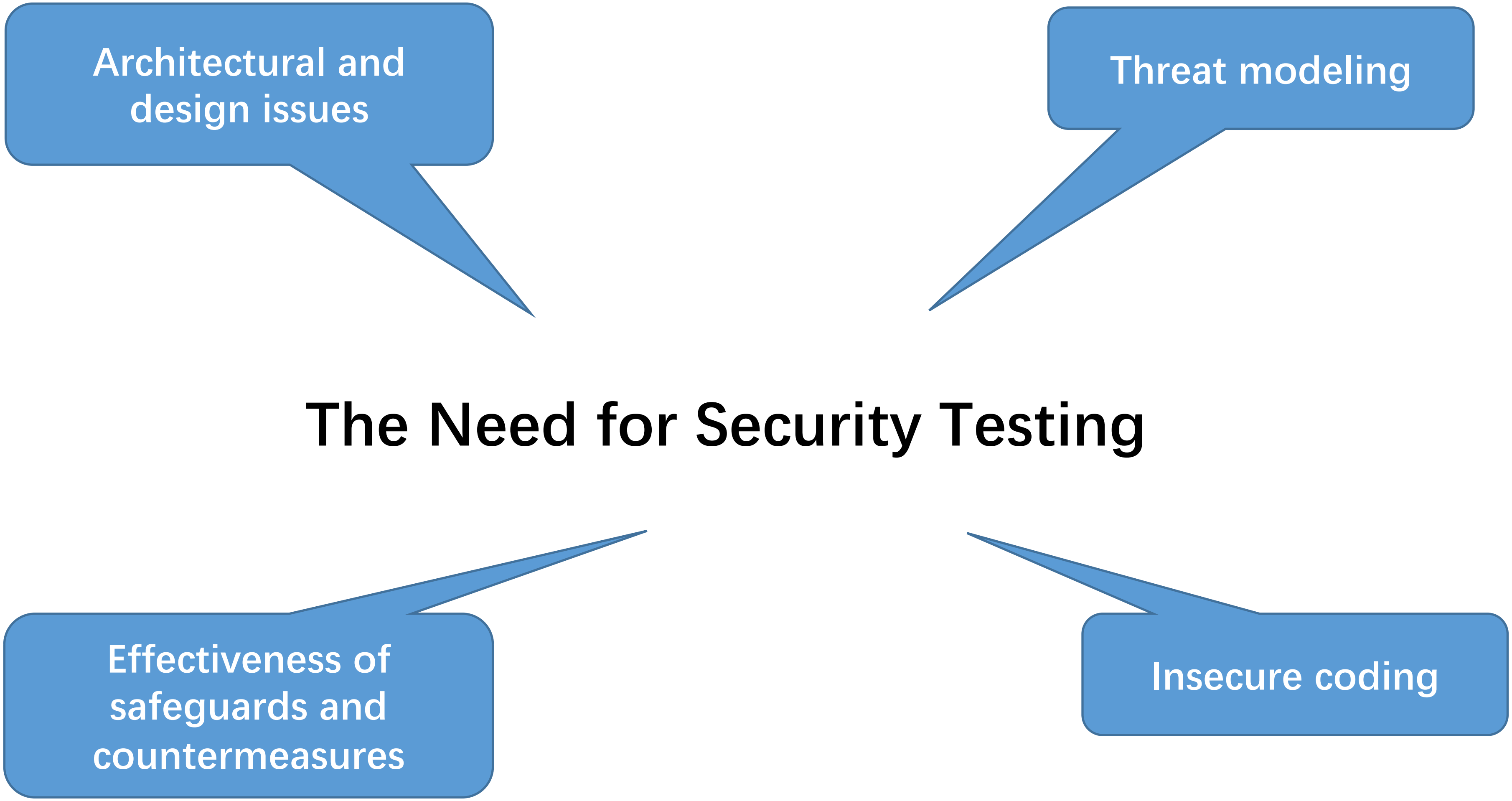
Secure Software Design and Engineering
(CY-321)

Software Security Testing

Dr. Zubair Ahmad

The Need for Security Testing



Architectural and design issues

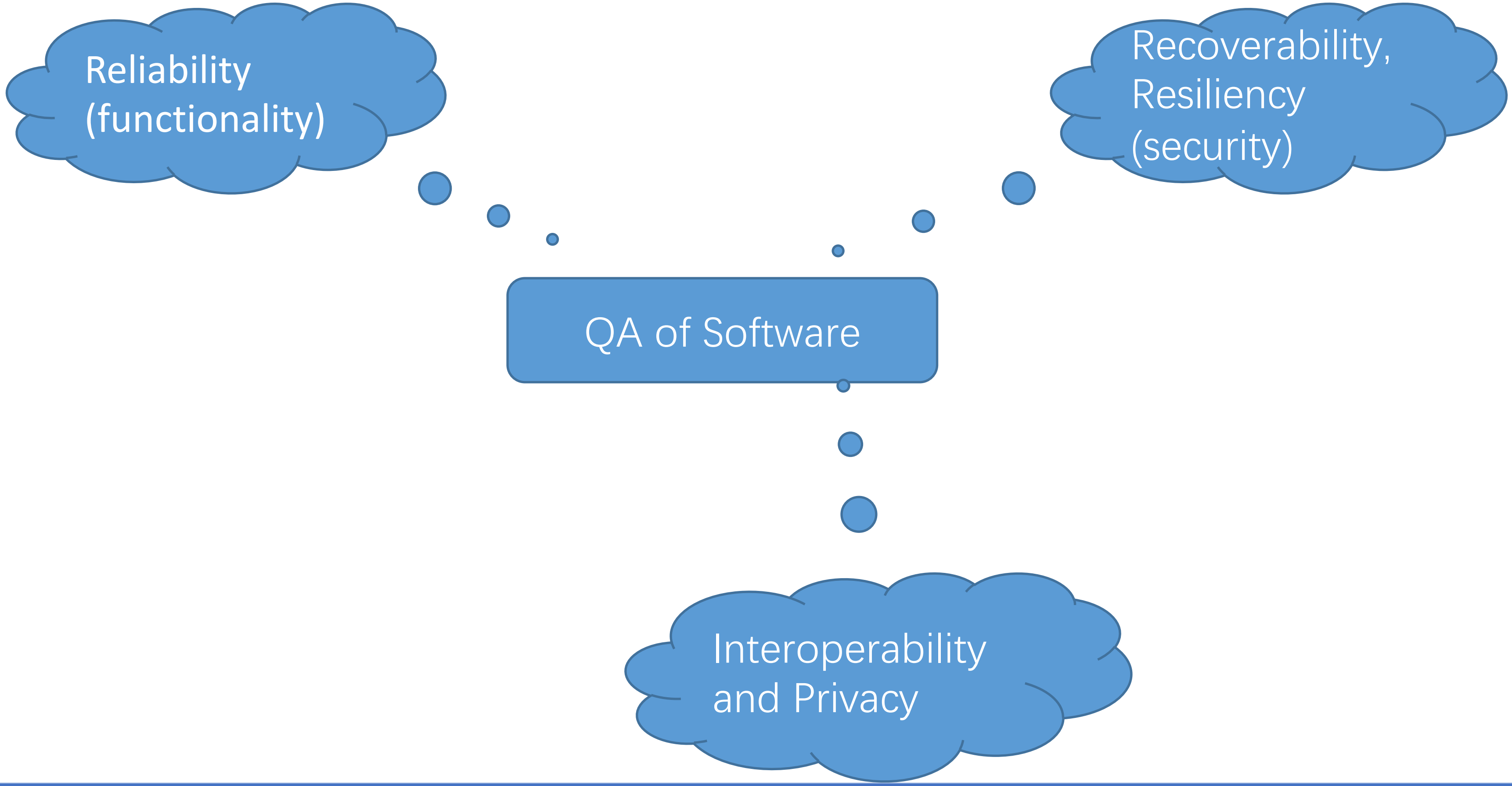
Threat modeling

Effectiveness of safeguards and countermeasures

Insecure coding

What to Test in Software Testing?

The software testing teams are rightfully referred to as quality assurance (QA) teams



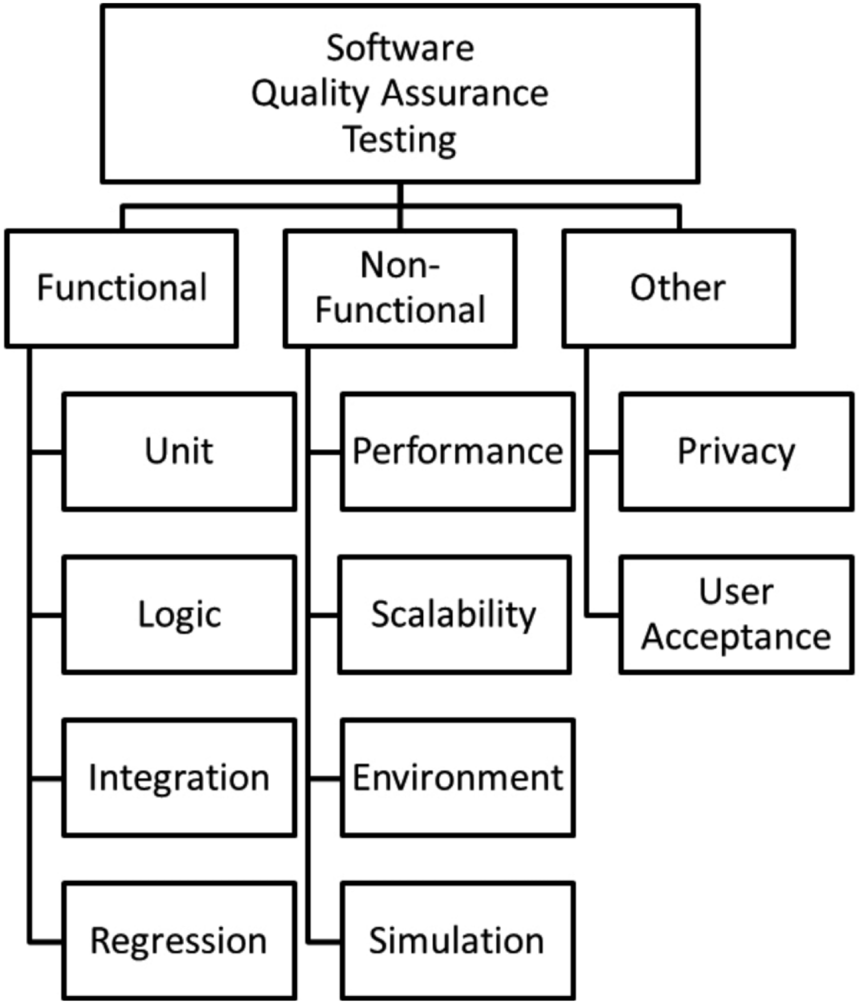
Reliability (functionality)

Recoverability, Resiliency (security)

QA of Software

Interoperability and Privacy

Types of Software QA Testing

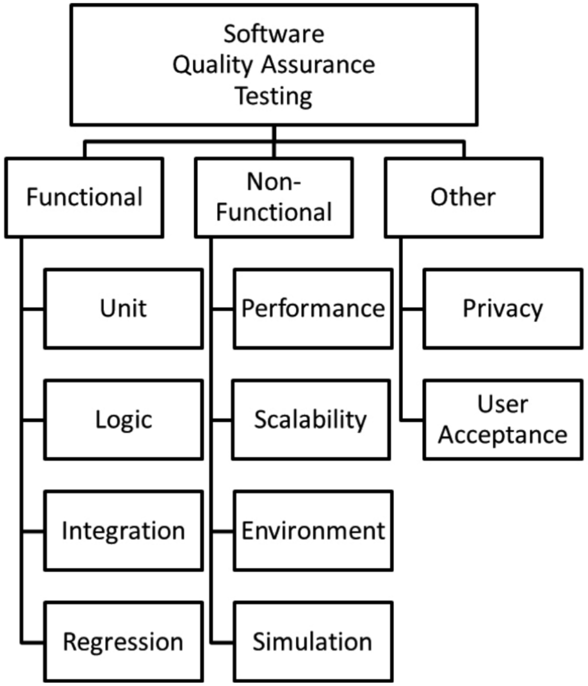


Types of Software QA Testing

Functional Testing

Functional testing is also referred to as *reliability* testing

To check if the software is reliable, a.k.a. is functioning as it is supposed to, according to the requirements specified by the business owner.



Functional Testing

Unit Testing

First process to ensure that the software is functioning properly, according to specifications

Performed during the implementation phase (coding) of the SDLC

Performed by breaking the functionality of the software into smaller parts and each part is tested in isolation from the other parts

Modular programming in mind

Functional Testing

Unit Testing

Dangling code

Unit testing can be used to find Quality of Code (QoC) issues

Uncover inefficiencies, **Cyclomatic complexities** and vulnerabilities in code

Infinite loop constructs → DoS Attacks

Functional Testing

Logic Testing

Validates the accuracy of the software processing logic

Logic testing also includes the testing of predicates

Logic testing is usually performed by negating or mutating (varying) the intended functionality

Boolean predicates return a true or false depending on whether the software logic is met or not

Functional Testing

Logic Testing

Code was not Unit Tested

```
public int Add(int p_iA, int p_iB)
{
    return p_iA + p_iB;
}

public int Multiply(int p_iA, int p_iB)
{
   return p_iA + p_iB;
}
```

Code was Unit Tested

```
public int Add(int p_iA, int p_iB)
{
    return p_iA + p_iB;
}

public int Multiply(int p_iA, int p_iB)
{
return p_iA * p_iB;
}
```

Functional Testing

Integration Testing

Type of Testing when units of code are combined

The security of the *sum of all parts* should also be tested

If individual code units have successfully passed unit testing, but fail when they are integrated, then it is a clear cut indication of software problems upon integration

Functional Testing

Regression Testing

Software is not static ??

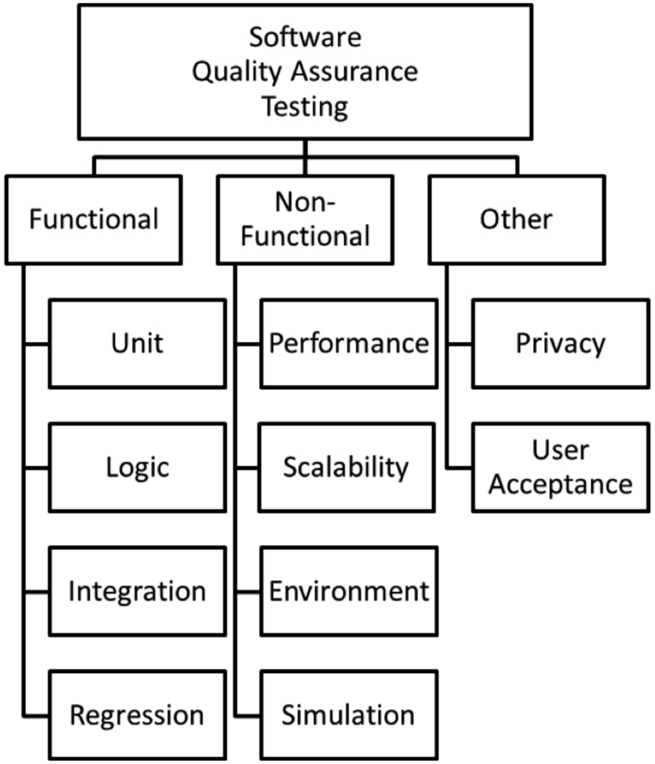
Whenever code or data is modified, there is a likelihood for those changes to break something that was previously functional

To validate that the software did not break previous functionality or security and regress to a non- functional or insecure state

Non-Functional Testing

Covers testing for the recoverability and environmental aspects of the software

To check if the software will be available when required and that it has appropriate replication, load balancing, interoperability and disaster recovery mechanisms



Non-Functional Testing

Performance Testing

To ensure that the software is performing to the SLA and expectations of the business

Secure features can have a significant impact on performance

Performance testing is not performed with the intent of finding vulnerabilities (bugs or flaws) but with the goal of determining bottlenecks that are present in the software

Bottlenecks can be reduced by tuning the software

Tuning is performed to optimize resource allocation

Non-Functional Testing

Performance Testing

Load Testing

Stress Testing

Non-Functional Testing

Performance Testing

Load Testing

The goal of identifying the maximum operating capacity for the software

Also referred to as longevity or endurance or volume testing

Non-Functional Testing

Performance Testing

Stress Testing

If load testing is to determine the point at which the software can operate with maximum capacity, stress testing is taking that test one step further

It is mainly aimed to determine the breaking point of the software, i.e., the point at which the software can no longer function

the software is subjected to extreme conditions such as maximum concurrency, limited computing resources, or heavy loads.

Non-Functional Testing

Performance Testing

Stress Testing

Primarily performed with **two objectives**

First, if the software can recover gracefully upon failure, when the software breaks

Second is to assure that the software operates according to the design principle of failing securely

Non-Functional Testing

Scalability Testing

Main objectives are to identify the loads (which can be obtained from load testing)

And to mitigate any bottlenecks that will hinder the ability of the software to scale to handle more load or changes in business processes or technology.

Non-Functional Testing

Environment Testing

The testing of the security of the environment itself in which the software will operate

Needs to verify the integrity of not just the configuration of the environment but also that of the data

Trust boundaries and Sandboxes

Non-Functional Testing

Interoperability Testing

When software operates in disparate environments, it is imperative to verify the resiliency of the interfaces that exist between the environments.

To check the software’s upstream and downstream dependency interfaces

- **security standards**
- **complete mediation**
- tokens used for transfer of credentials cannot be stolen, spoofed and replayed, and

Non-Functional Testing

Disaster Recovery (DR) Testing

The ability of the software to restore its operation after a disaster happens

DR testing verifies the recoverability of the software

Uncovers data accuracy, integrity and system availability issues

Used to gauge the effectiveness of error handling and auditing in software as well

Does the software fail securely and how does it report errors upon downtime?

Is there proper logging of the failure in place?

Non-Functional Testing

Simulation Testing

A common issue faced by software teams is that the software functions as desired in the development and test environments but fails in the production environment

The most probable root cause for such varied behavior is that the configuration settings in these environments differ.

The effectiveness of least privilege implementation and configuration mismatches can be uncovered using simulation testing

Other Testing

Privacy Testing

Software should be tested to assure privacy

For software that handles personal data, privacy testing must be part of the test plan

encompass the monitoring of network traffic and the communication between end-points to assure that personal information is not disclosed

Tests for the appropriateness of notices and disclaimers when personal information is collected must also be conducted

Other Testing

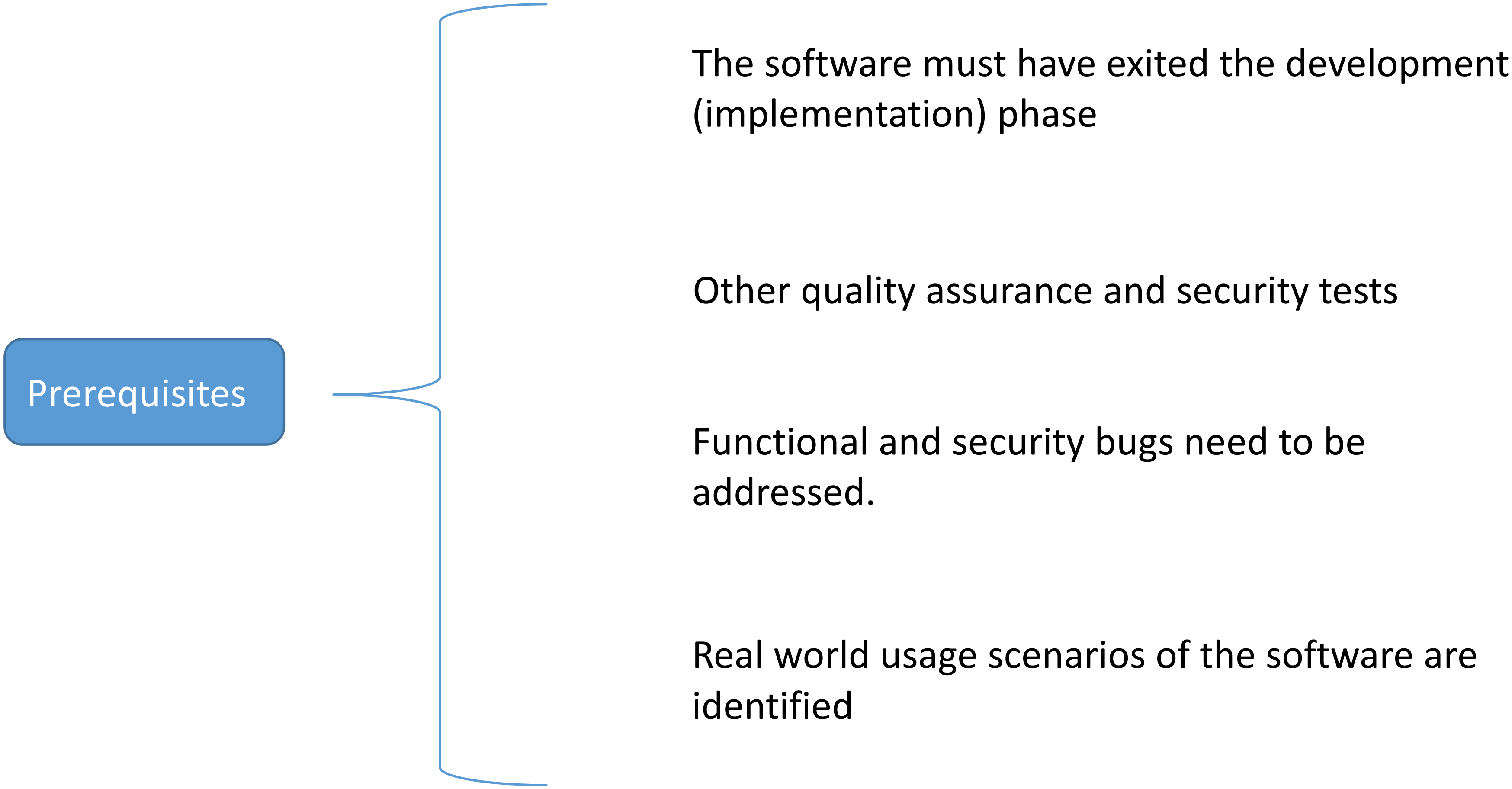
User Acceptance Testing (UAT)

During the software acceptance phase, the end user needs to be assured that the software meets their specified requirements

Also known as end user testing or smoke testing

Other Testing

User Acceptance Testing (UAT)



Prerequisites

The software must have exited the development (implementation) phase

Other quality assurance and security tests

Functional and security bugs need to be addressed.

Real world usage scenarios of the software are identified

Security Testing Methods

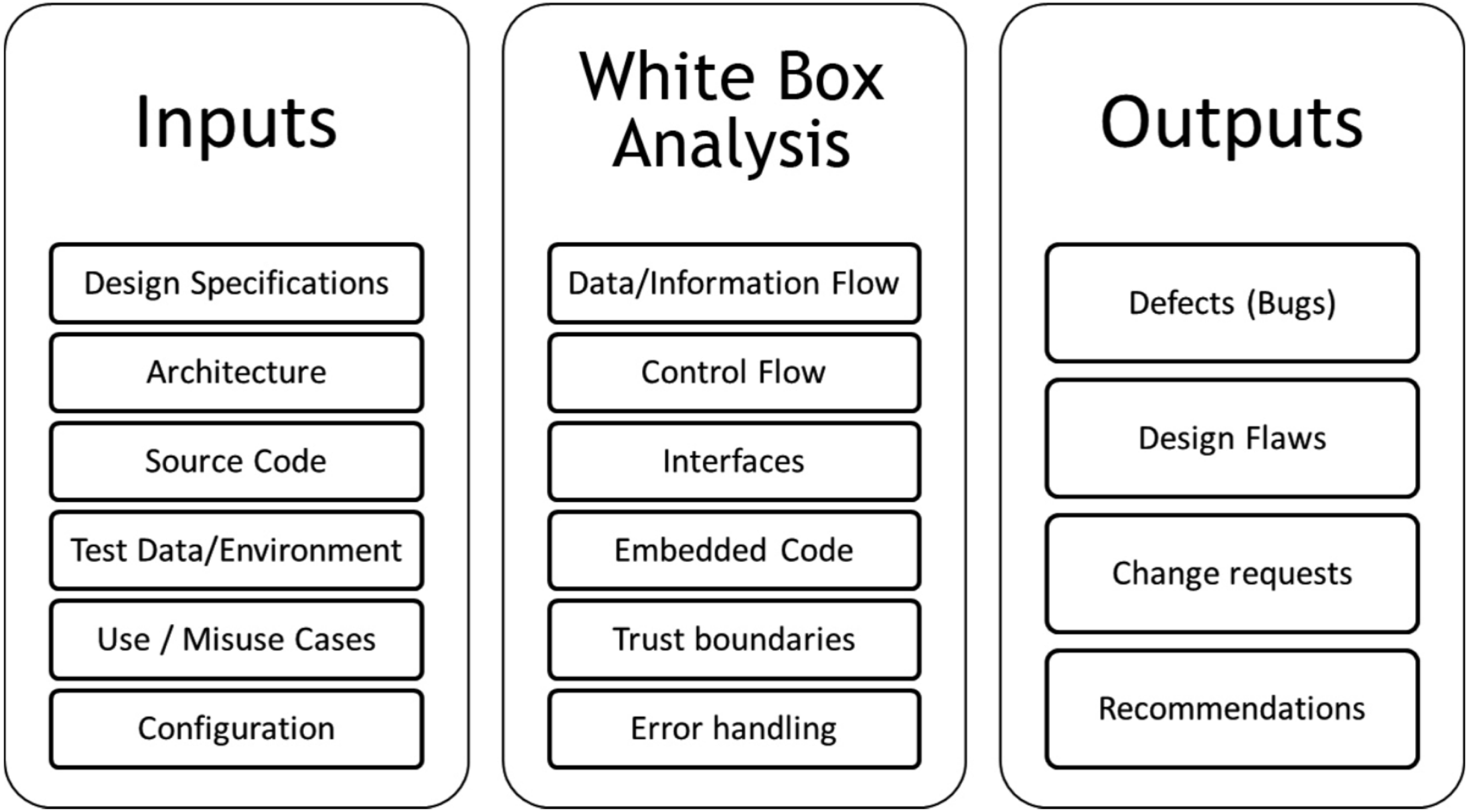
White Box Testing

Glass box or Clear box testing

Performed based on the knowledge of how the software is designed and implemented

To perform white box security testing, it is imperative to first understand the scope, context and intended functionality of the software so that the inverse of that can be tested with an attacker’s perspective.

Security Testing Methods



Inputs

- Design Specifications
- Architecture
- Source Code
- Test Data/Environment
- Use / Misuse Cases
- Configuration

White Box Analysis

- Data/Information Flow
- Control Flow
- Interfaces
- Embedded Code
- Trust boundaries
- Error handling

Outputs

- Defects (Bugs)
- Design Flaws
- Change requests
- Recommendations

Security Testing Methods

Black Box Testing

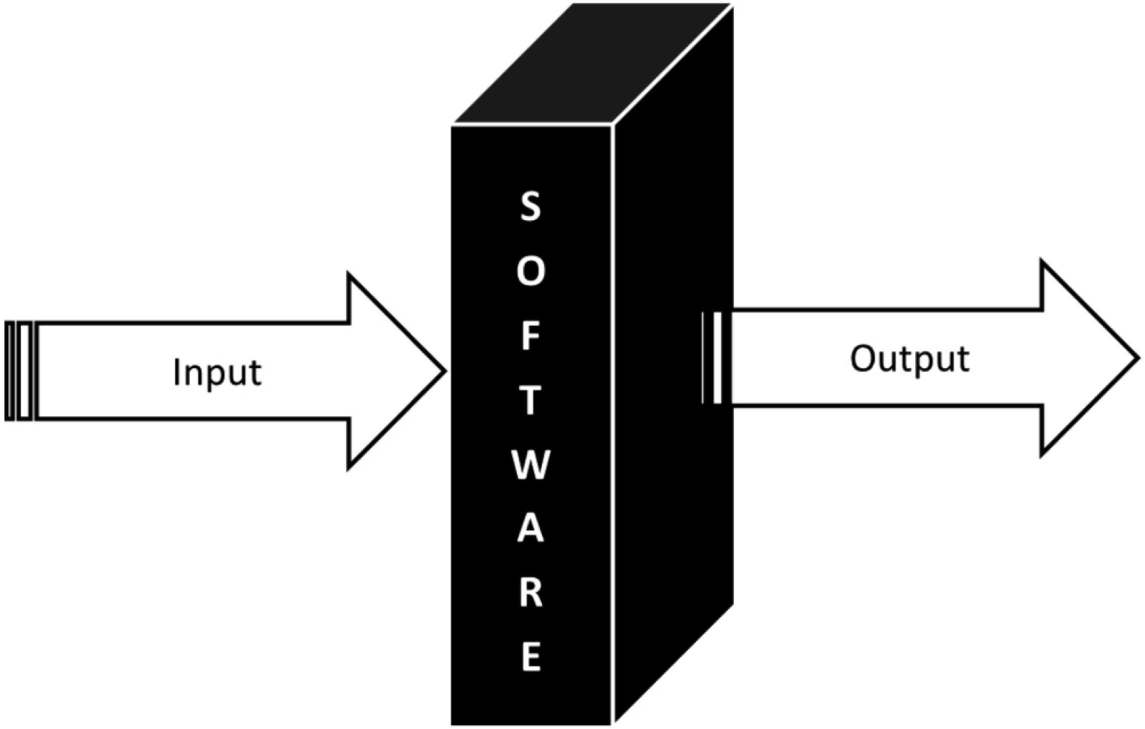
Broadly known as ***zero knowledge* assessment**

The tester has very limited to no knowledge of the internal working of the software being tested

While **white box** testing is **structural analysis** of the software's security, **black box testing** is **behavioral analysis** of the software's security.

Security Testing Methods

Black Box Testing



Questions??

zubair.ahmad@giki.edu.pk

Office: G14 FCSE lobby