# Test Levels

## Component / Unit Testing

* Testing separate components of the software
* Respective tests are called:
  + Module, unit, program or class tests
* **Individual testing** 
  + Components are tested individually
  + Isolated from all other software components
* **Isolation**
  + Prevents **external influences** on the components
* Component test checks aspects **internal to the component**
  + Interaction with neighbors is not performed

## Integration Testing

* Composing units to form **larger structural units** and subsystems
* Done by developers, testers, or special integration teams
* Supposes that components are **already tested individually**
* **Levels of Integration Testing**
  + Component integration testing
    - Expose defects in the interfaces and interaction between integrated components
    - Also called “Integration test in the small”
  + System integration testing
    - Testing the integration of systems and packages
    - Testing interfaces to external organizations
    - Also called “Integration test in the large”
* **Off-the-shelf Products**
  + Standard, **existing components** used with some modification
  + Usually not subject of component testing
  + Must be tested for **integration**
* After assembling the components **new fault may occur**
* Testing must confirm that all components **collaborate** correctly
* The main goal - **exposing faults**
  + In the interfaces
  + In the **interaction** between integrated components
* Some Typical Problems
  + Wrong **interface formats**
    - Incompatible interface formats
    - Wrong files format
  + Typical faults in **data exchange**
    - Syntactically wrong or no data
    - Different interpretation of received data
    - Timing problems

## System Testing

## Acceptance Testing

# Test Types

## Risk-Based Testing

* The possibility of a negative or undesirable outcome or event
* Any problem that may occur
  + Would decrease perceptions of product quality or project success
* Factors for classifying the level of risk:
  + **Likelihood** of the problem occurring
    - Arises from technical considerations
    - E.g. programming languages used, bandwidth of connections, etc.
  + **Impact** of the problem in case it occurs
    - Arises from business considerations
    - E.g. financial loss, number of users affected, etc.
* The more important risks are tested first

## Functional Testing

* Verifies input-output behavior
* Black box testing
* Test basis – functional requirements
* What the system must be able to do
* **System Testing – mainly used**
* **Acceptance Testing – mainly used**

## Non-functional Testing

* Is it easy to use, comfortable and effective
* Reliability, Usability, Efficiency
* **Performance Tests**
  + Processing speed and response time
* **Load Tests**
  + Behavior in increasing system loads
    - Number of simultaneous users
    - Number of transactions
* **Stress Tests**
  + Behavior when overloaded
* **Volume test**
  + Behavior dependent on the amount of data
* **Testing of security** 
  + Against unauthorized access
  + Service attacks
* **Stability**
  + Mean time between failures
  + Failure rate with a given user profile
* **Robustness Tests**
  + Response
  + Examination of **exception handling** and recovery to errors
* **Compatibility and data conversion**
  + Compatibility to given systems
  + Import/export of data
* **Different configurations of the system**
  + Back-to-back testing
* **Usability test**
  + Ease of learning the system
  + Ease and efficiency of operation
  + Understandability of the system

## Structural Testing

* Often referred to as ‘white-box’ or ‘glass-box’ testing
* Uses information about the **internal code structure or architecture**
* Tools can be used to measure the code coverage of elements, such as statements or decisions
* **Component testing – mostly used**
* **Integration testing – mostly used**
* **System testing**
* **Acceptance testing**

## Re-testing

* After a defect is detected and fixed, the software should be **re-tested to confirm that the bug is fixed**

## Regression Testing

* Testing of a previously tested program following modification to **ensure that defects have not been introduced** or uncovered in unchanged areas of the software, as a result of the changes made. It is performed when the software or its environment is changed.
* Test cases, used in regression testing, run **many times**
  + They have to be well documented and reusable
  + Strong candidates for **test automation**

## Maintenance Testing

* Software does not wear out and tear
* Some design faults already exist
* Bugs are about to be revealed