BINUS UNIVERSITY

People
Innovation
Excellence

Course : COMP6100/Software Engineering
Effective Period : Desember 2017

Testing Applications and Security Engineering

Session 17 - 18

Acknowledgement

These slides have been adapted from Pressman, R.S. (2015). *Software Engineering : A Practioner's Approach. 8th ed*. McGraw-Hill Companies.Inc, Americas, New York. ISBN 978 1 259 253157. Chapter 23, 24, 25, 26, and 27

People
Innovation
Excellence

Learning Objectives

LO 3 : Demonstrate the quality assurances and the potential showcase business project

Contents

- Testing Conventional Applications
- Testing Object Oriented Applications
- Testing Web Applications
- Testing Mobile Applications
- Security Engineering

Testing Conventional Applications

Software Testing Fundamentals : Testability

- Operability —it operates cleanly
- Observability —the results of each test case are readily observed
- Controllability —the degree to which testing can be automated and optimized
- Decomposability —testing can be targeted
- Simplicity —reduce complex architecture and logic to simplify tests
- Stability —few changes are requested during testing
- Understandability —of the design

People
Innovation
Excellence

Testing Conventional Applications

What is a “Good” Test?

- A good test has a high probability of finding an error
- A good test is not redundant.
- A good test should be “best of breed”
- A good test should be neither too simple nor too complex

Testing Conventional Applications

BINUS UNIVERSITY

Internal and External Views of Testing

- Any engineered product (and most other things) can be tested in one of two ways:
  - Knowing the specified function that a product has been designed to perform, tests can be conducted that demonstrate each function is fully operational while at the same time searching for errors in each function;
  - Knowing the internal workings of a product, tests can be conducted to ensure that "all gears mesh," that is, internal operations are performed according to specifications and all internal components have been adequately exercised.

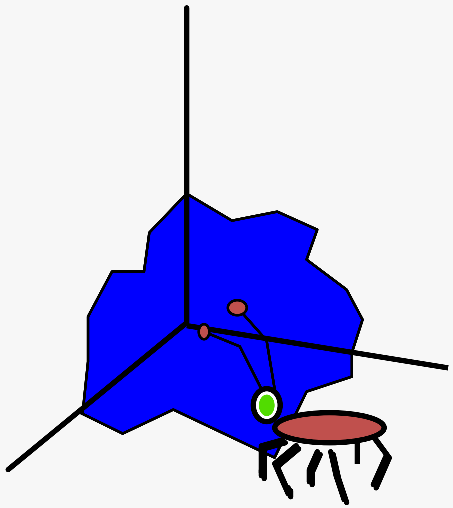
People
Innovation
Excellence

Testing Conventional Applications

Test Case Design

"Bugs lurk in corners and congregate at boundaries ..."

Boris Beizer

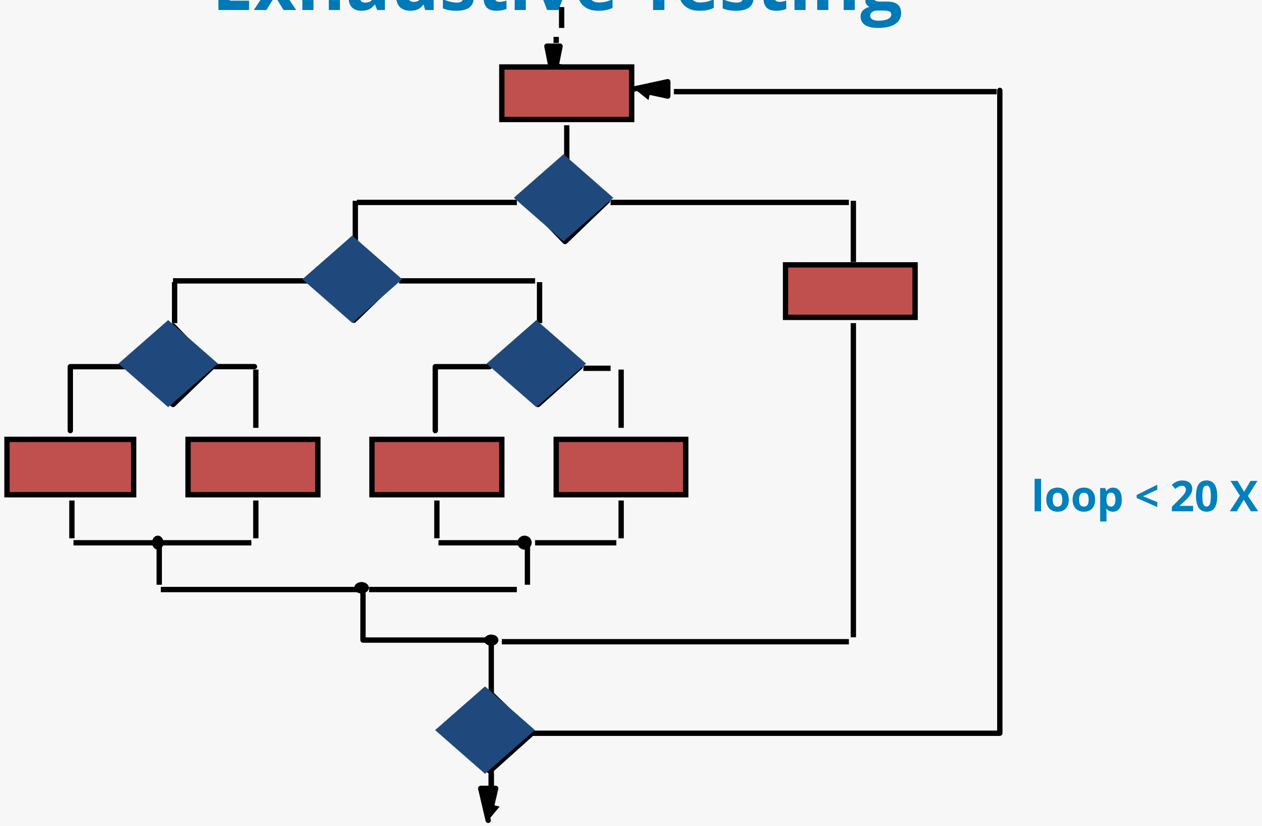


People
Innovation
Excellence

OBJECTIVE	to uncover errors
CRITERIA	in a complete manner
CONSTRAINT	with a minimum of effort and time

Testing Conventional Applications

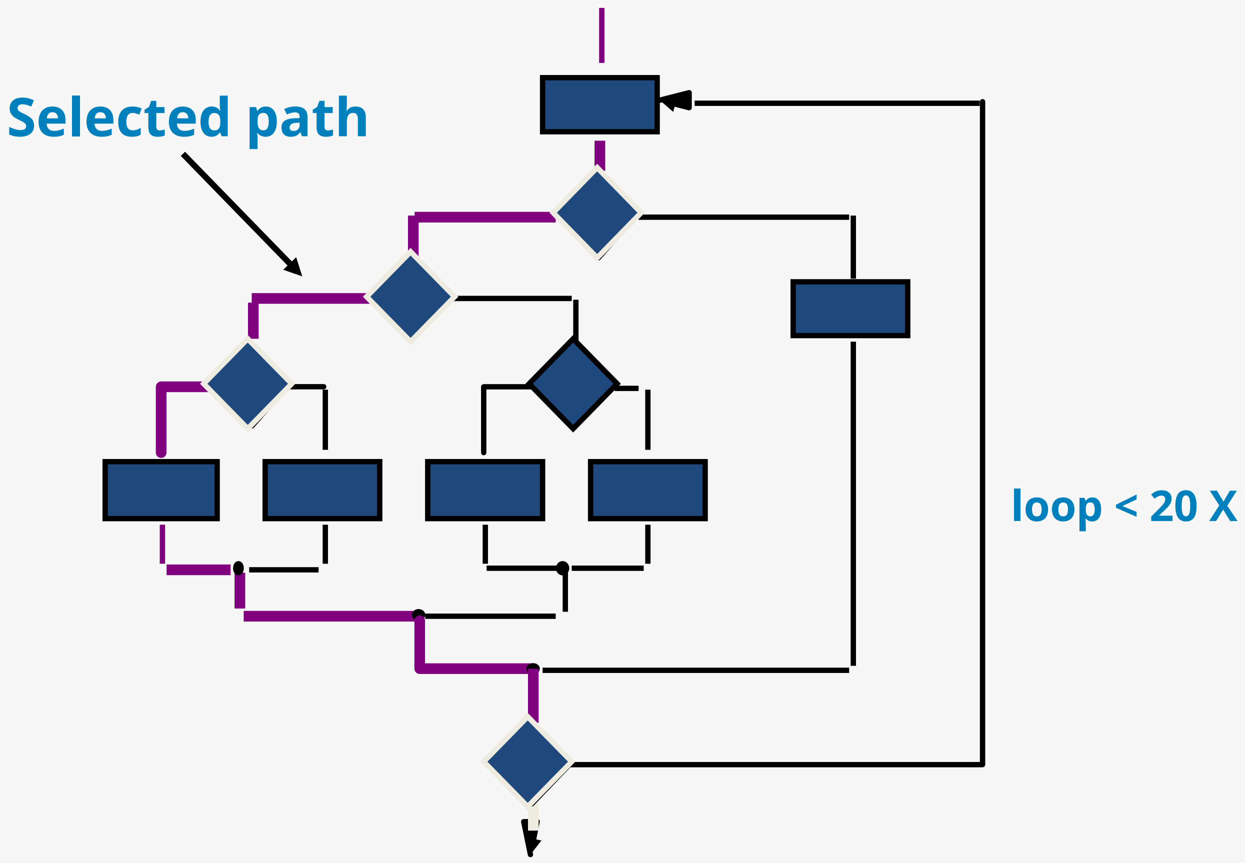
Exhaustive Testing



There are 10^{14} possible paths! If we execute one test per millisecond, it would take 3,170 years to test this program!!

Testing Conventional Applications

Selective Testing

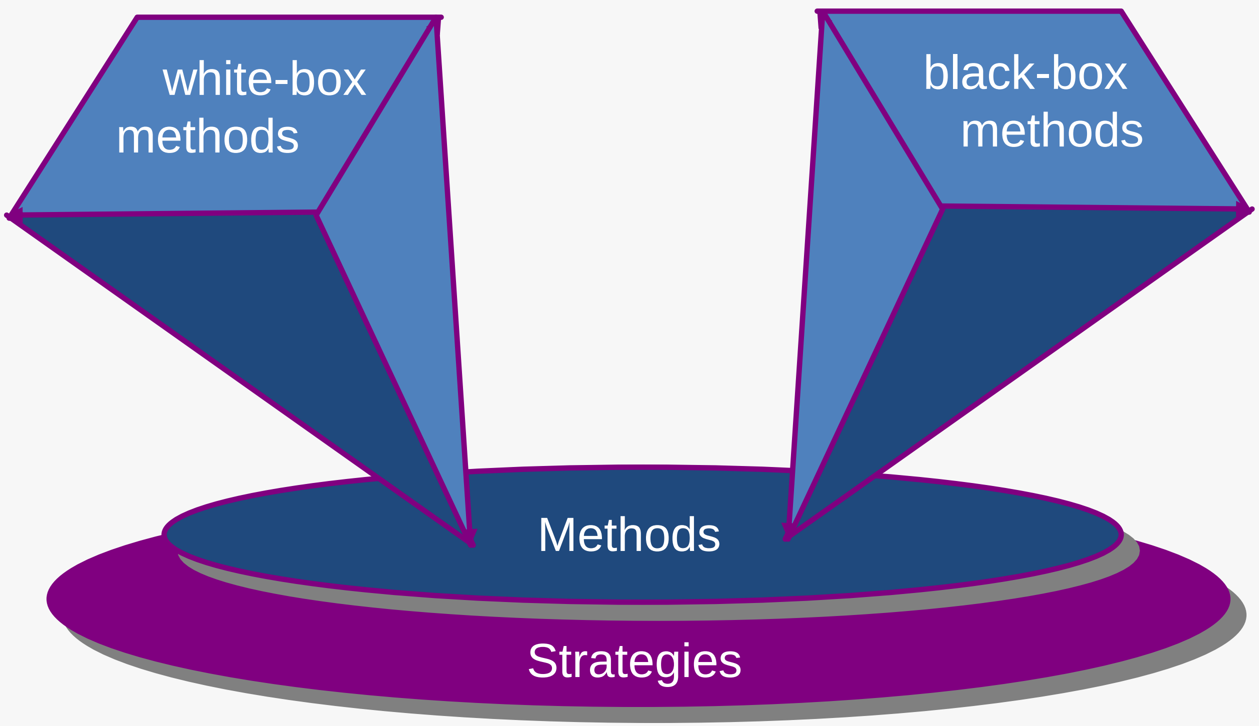


People
Innovation
Excellence

Testing Conventional Applications

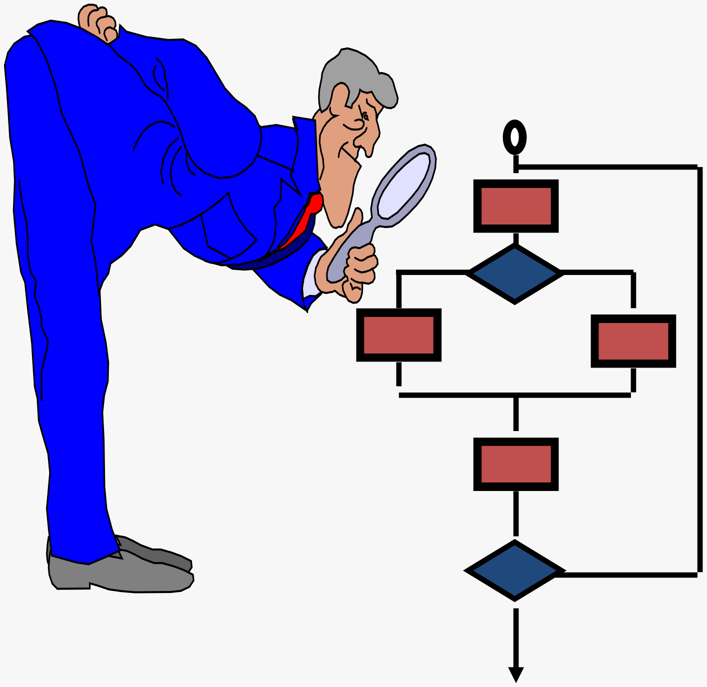
People
Innovation
Excellence

Software Testing



Testing Conventional Applications

White-Box Testing



... our goal is to ensure that all

statements and conditions have

been executed at least once ...

People
Innovation
Excellence

Testing Conventional Applications

White-Box Testing

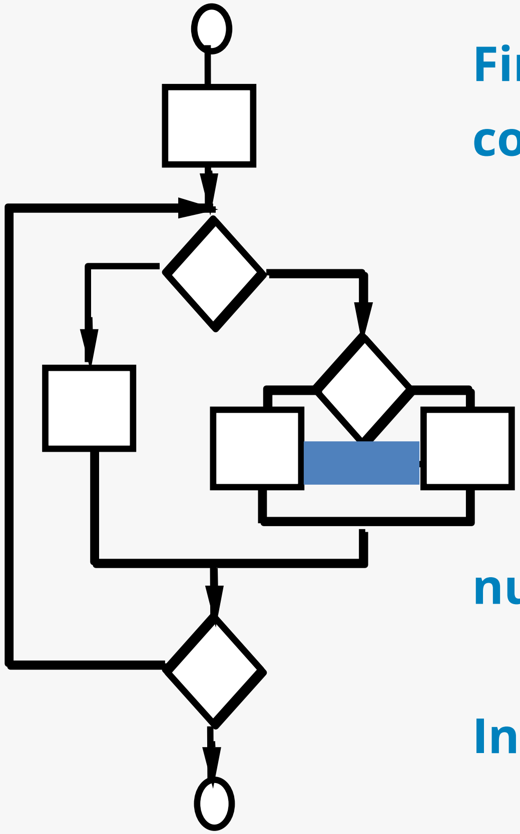
- logic errors and incorrect assumptions are inversely proportional to a path's execution probability

- we often believe that a path is not likely to be executed; in fact, reality is often counter intuitive

- typographical errors are random; it's likely that untested paths will contain some

Testing Conventional Applications

Basis Path Testing



First, we compute the cyclomatic complexity:

or

number of enclosed areas + 1

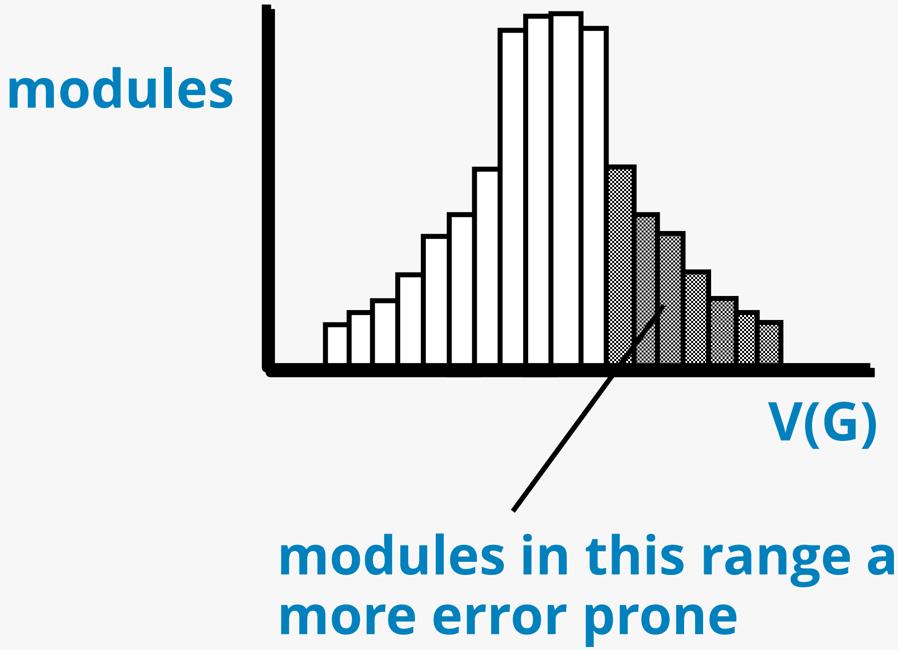
In this case, V(G) = 4

People
Innovation
Excellence

Testing Conventional Applications

Cyclomatic Complexity

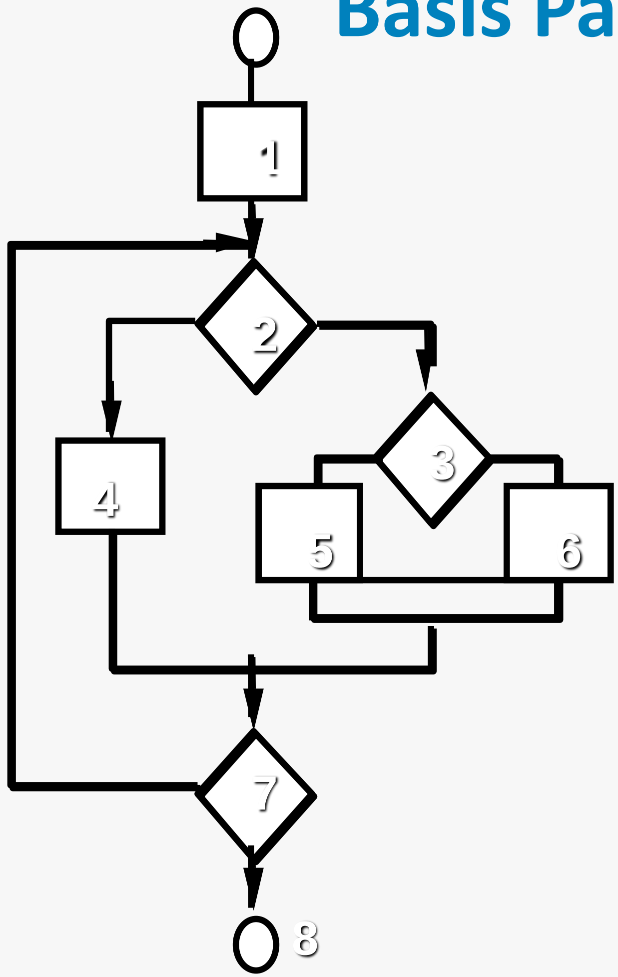
A number of industry studies have indicated that the higher V(G), the higher the probability or errors.



People
Innovation
Excellence

Testing Conventional Applications

Basis Path Testing



Next, we derive the independent paths:

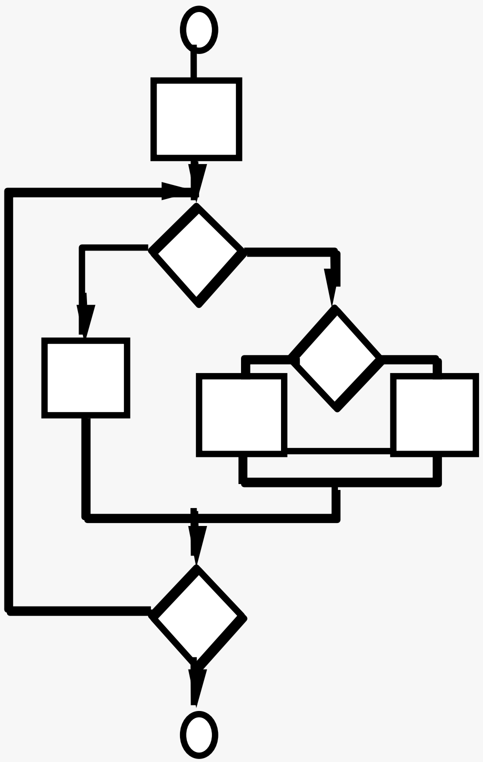
Since V(G) = 4, there are four paths

- Path 1: 1,2,3,6,7,8
- Path 2: 1,2,3,5,7,8
- Path 3: 1,2,4,7,8
- Path 4: 1,2,4,7,2,4,...7,8

Finally, we derive test cases to exercise these paths.

Testing Conventional Applications

Basis Path Testing Notes



- you don't need a flow chart, but the picture will help when you trace program paths
- count each simple logical test, compound tests count as 2 or more
- basis path testing should be applied to critical modules

People
Innovation
Excellence

Testing Conventional Applications

Basis Path Testing - Graph Matrices

- A graph matrix is a square matrix whose size (i.e., number of rows and columns) is equal to the number of nodes on a flow graph
- Each row and column corresponds to an identified node, and matrix entries correspond to connections (an edge) between nodes.
- By adding a *link weight* to each matrix entry, the graph matrix can become a powerful tool for evaluating program control structure during testing

Testing Conventional Applications

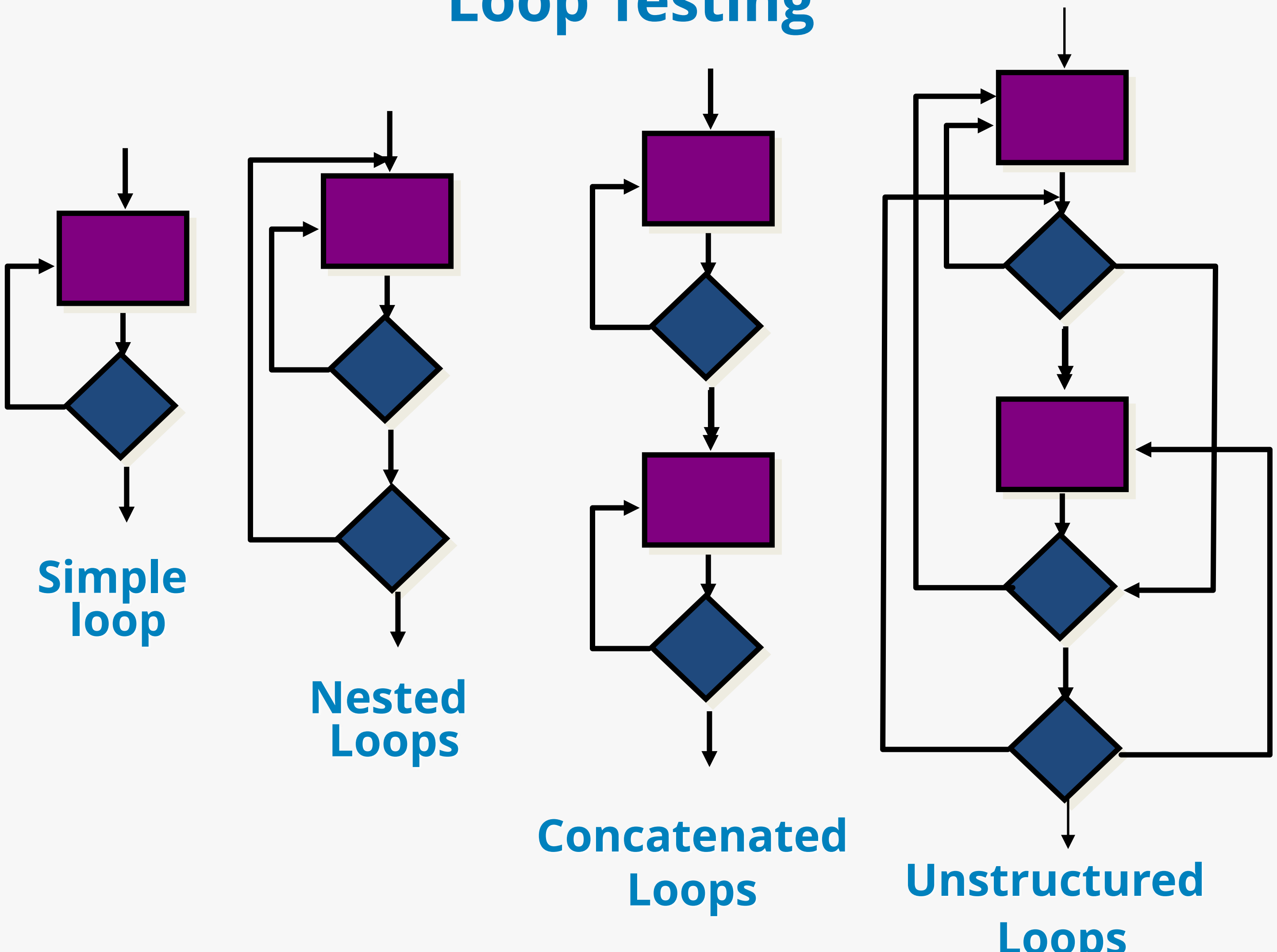
Control Structure Testing

- **Condition testing — a test case design method that exercises the logical conditions contained in a program module**
- **Data flow testing — selects test paths of a program according to the locations of definitions and uses of variables in the program**

People
Innovation
Excellence

Testing Conventional Applications

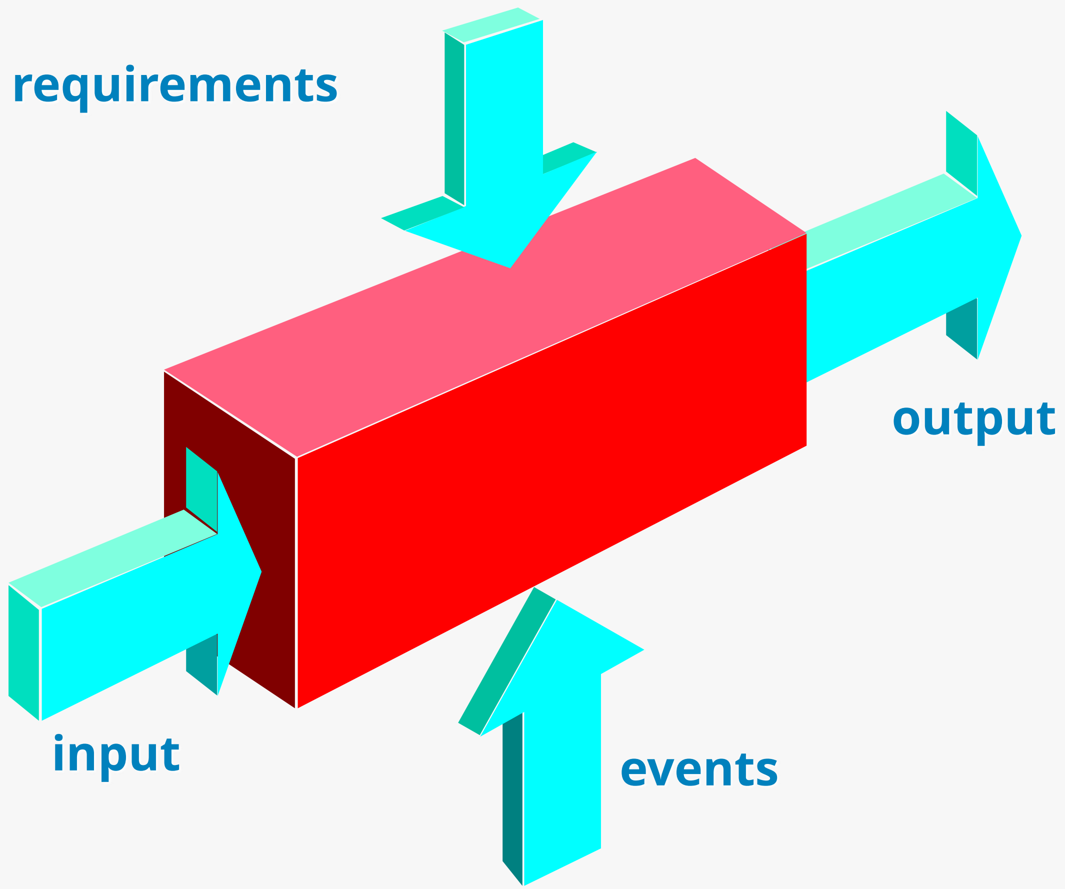
Loop Testing



People
Innovation
Excellence

Testing Conventional Applications

Black-Box Testing



People
Innovation
Excellence

Testing Conventional Applications

Black-Box Testing

- How is functional validity tested?
- How is system behavior and performance tested?
- What classes of input will make good test cases?
- Is the system particularly sensitive to certain input values?
- How are the boundaries of a data class isolated?
- What data rates and data volume can the system tolerate?
- What effect will specific combinations of data have on system operation?

People
Innovation
Excellence

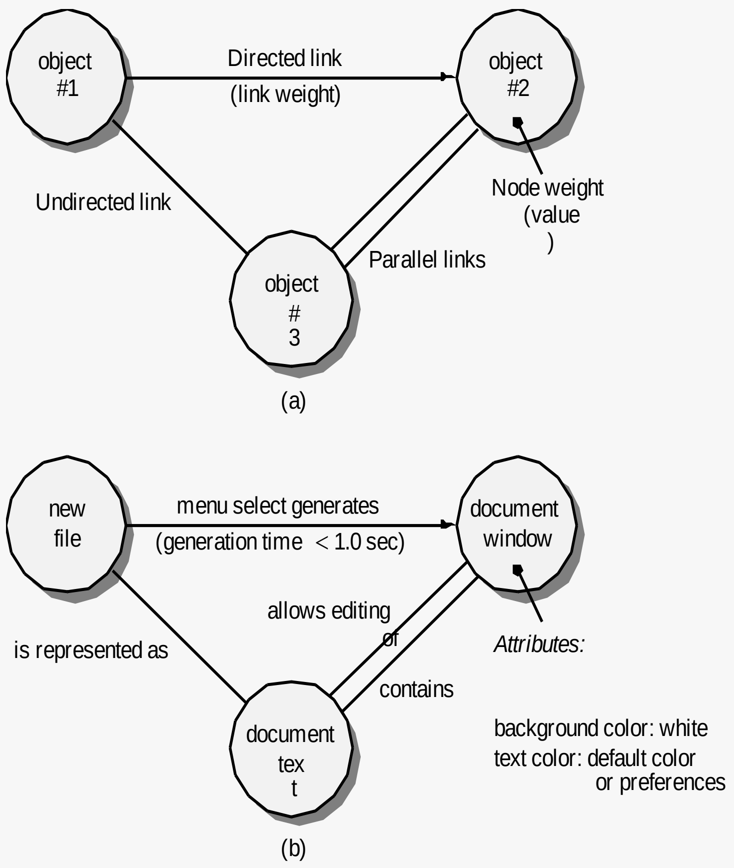
BINUS UNIVERSITY

Testing Conventional Applications

Graph-Based Methods

To understand the objects that are modeled in software and the relationships that connect these objects

In this context, we consider the term “objects” in the broadest possible context. It encompasses data objects, traditional components (modules), and object-oriented elements of computer software.

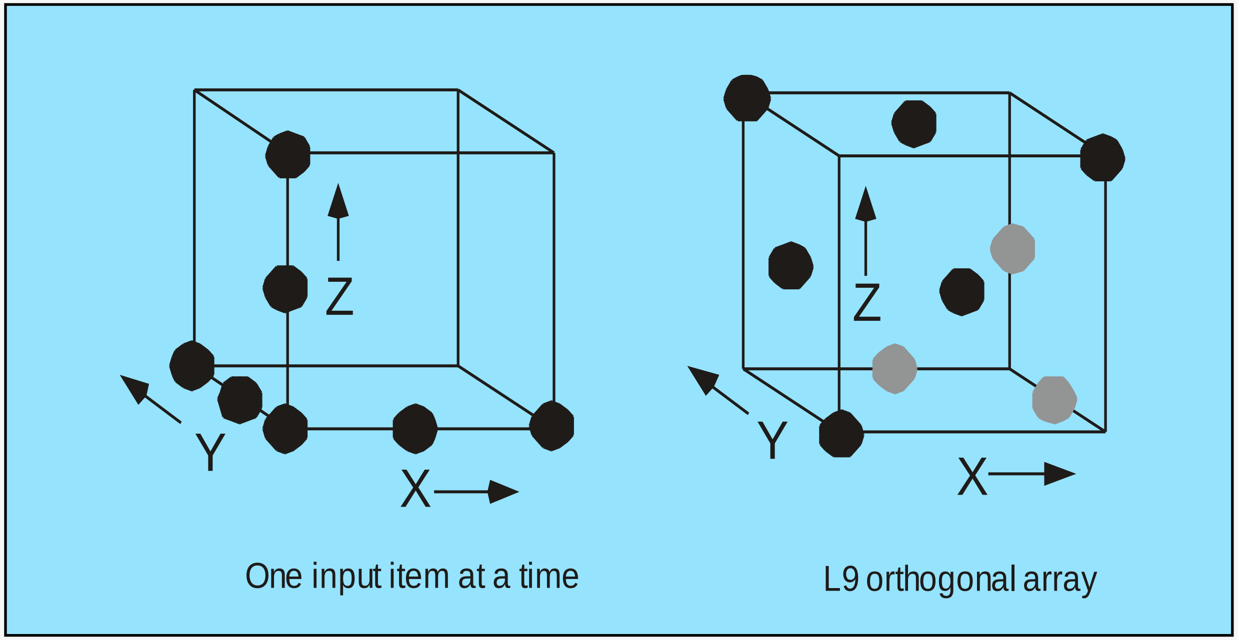


People
Innovation
Excellence

Testing Conventional Applications

Orthogonal Array Testing

- Used when the number of input parameters is small and the values that each of the parameters may take are clearly bounded



People
Innovation
Excellence

Testing Object Oriented Application

- To adequately test OO systems, three things must be done:
  - the definition of testing must be broadened to include error discovery techniques applied to object-oriented analysis and design models
  - the strategy for unit and integration testing must change significantly, and
  - the design of test cases must account for the unique characteristics of OO software.

Testing Object Oriented Application

‘Testing’ OO Models

- **The review of OO analysis and design models is especially useful because the same semantic constructs (e.g., classes, attributes, operations, messages) appear at the analysis, design, and code level**
- **Therefore, a problem in the definition of class attributes that is uncovered during analysis will circumvent side affects that might occur if the problem were not discovered until design or code (or even the next iteration of analysis).**

Correctness of OO Models

- During analysis and design, semantic correctness can be asesssed based on the model's conformance to the real world problem domain.
- If the model accurately reflects the real world (to a level of detail that is appropriate to the stage of development at which the model is reviewed) then it is semantically correct.
- To determine whether the model does, in fact, reflect real world requirements, it should be presented to problem domain experts who will examine the class definitions and hierarchy for omissions and ambiguity.
- Class relationships (instance connections) are evaluated to determine whether they accurately reflect real-world object connections.

Testing Object Oriented Application

People
Innovation
Excellence

Class Model Consistency

- Revisit the CRC model and the object-relationship model.
- Inspect the description of each CRC index card to determine if a delegated responsibility is part of the collaborator's definition.
- Invert the connection to ensure that each collaborator that is asked for service is receiving requests from a reasonable source.
- Using the inverted connections examined in the preceding step, determine whether other classes might be required or whether responsibilities are properly grouped among the classes.
- Determine whether widely requested responsibilities might be combined into a single responsibility.

OO Testing Strategies

- Unit testing
  - the concept of the unit changes
  - the smallest testable unit is the encapsulated class
  - a single operation can no longer be tested in isolation (the conventional view of unit testing) but rather, as part of a class
- Integration Testing
  - *Thread-based testing* integrates the set of classes required to respond to one input or event for the system
  - *Use-based testing* begins the construction of the system by testing those classes (called *independent classes*) that use very few (if any) of server classes. After the independent classes are tested, the next layer of classes, called *dependent classes*
  - *Cluster testing* defines a cluster of collaborating classes (determined by examining the CRC and object-relationship model) is exercised by designing test cases that attempt to

Testing Object Oriented Application

OO Testing Strategies

- Validation Testing
  - details of class connections disappear
  - draw upon use cases that are part of the requirements model
  - Conventional black-box testing methods can be used to drive validation tests

OOT Methods

Berard [Ber93] proposes the following approach:

1. Each test case should be uniquely identified and should be explicitly associated with the class to be tested
2. The purpose of the test should be stated.
3. A list of testing steps should be developed for each test and should contain:
   a. a list of specified states for the object that is to be tested
   b. a list of messages and operations that will be exercised as a on sequence of the test
   c. a list of exceptions that may occur as the object is tested
   d. a list of external conditions (i.e., changes in the environment external to the software that must exist in order to properly conduct the test)
   e. supplementary information that will aid in understanding or implementing the test.

Testing Object Oriented Application

People
Innovation
Excellence

Testing Methods

- Fault-based testing
  - The tester looks for plausible faults (i.e., aspects of the implementation of the system that may result in defects). To determine whether these faults exist, test cases are designed to exercise the design or code.
- Class Testing and the Class Hierarchy
  - Inheritance does not obviate the need for thorough testing of all derived classes. In fact, it can actually complicate the testing process.
- Scenario-Based Test Design
  - Scenario-based testing concentrates on what the user does, not what the product does. This means capturing the tasks (via use-cases) that the user has to perform, then applying them and their variants as tests.

Testing Object Oriented Application

OOT Methods: Random Testing

- Random testing
  - identify operations applicable to a class
  - define constraints on their use
  - identify a minimum test sequence
    - an operation sequence that defines the minimum life history of the class (object)
  - generate a variety of random (but valid) test sequences
    - exercise other (more complex) class instance life histories

OOT Methods: Partition Testing

- Partition Testing
  - reduces the number of test cases required to test a class in much the same way as equivalence partitioning for conventional software
  - state-based partitioning
    - categorize and test operations based on their ability to change the state of a class
  - attribute-based partitioning
    - categorize and test operations based on the attributes that they use
  - category-based partitioning
    - categorize and test operations based on the generic function each performs

Testing Object Oriented Application

OOT Methods: Inter-Class Testing

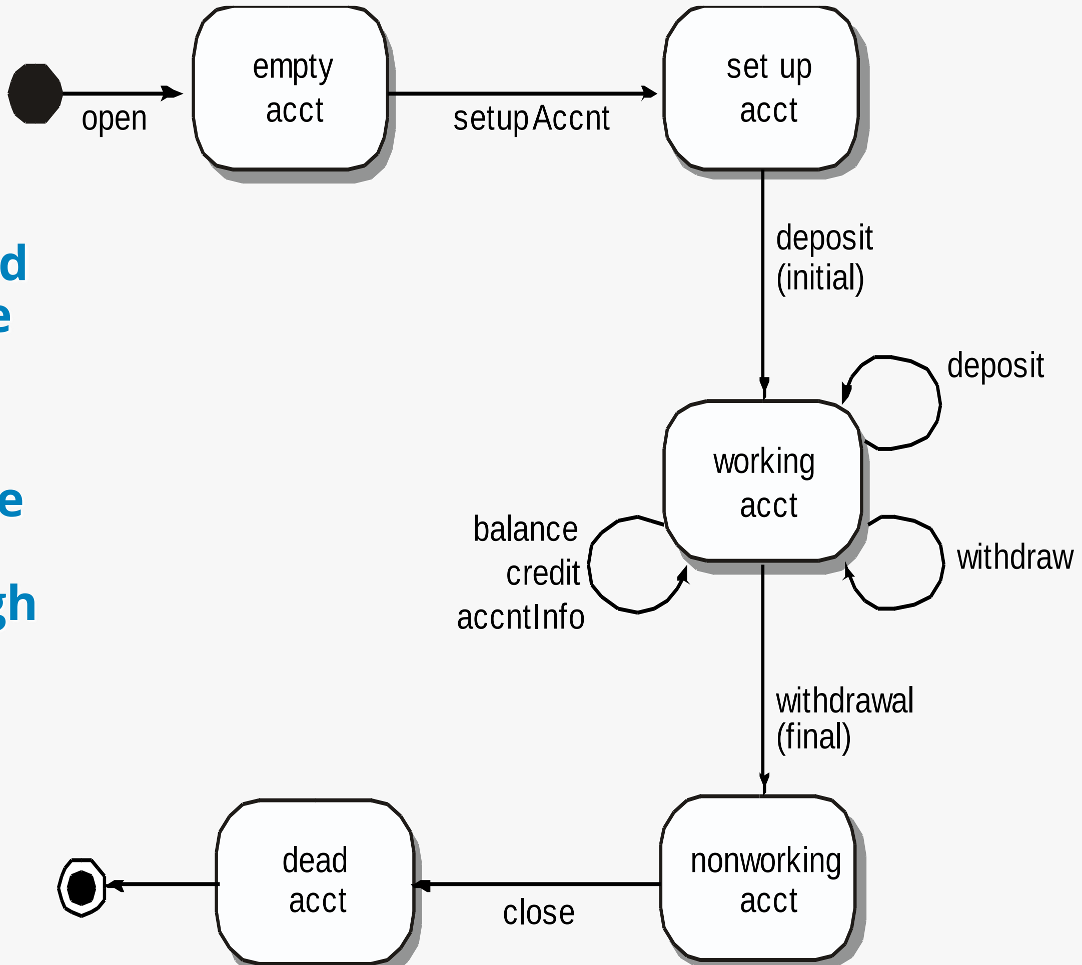
- Inter-class testing
  - For each client class, use the list of class operators to generate a series of random test sequences. The operators will send messages to other server classes.
  - For each message that is generated, determine the collaborator class and the corresponding operator in the server object.
  - For each operator in the server object (that has been invoked by messages sent from the client object), determine the messages that it transmits.
  - For each of the messages, determine the next level of operators that are invoked and incorporate these into the test sequence

Testing Object Oriented Application

OOT Methods: Behavior Testing

The tests to be designed should achieve all state coverage [KIR94].

That is, the operation sequences should cause the account class to make transition through all allowable states



Testing Web Applications

Testing Quality Dimensions-I

- ***Content*** is evaluated at both a syntactic and semantic level.
  - syntactic level
  - semantic level
- ***Function*** is tested for correctness, instability, and general conformance to appropriate implementation standards (e.g.,Java or XML language standards).
- ***Structure*** is assessed to ensure that it
  - properly delivers WebApp content and function
  - is extensible
  - can be supported as new content or functionality is added.
- ***Usability*** is tested to ensure that each category of user
  - is supported by the interface
  - can learn and apply all required navigation syntax and semantics
- ***Navigability*** is tested to ensure that
  - all navigation syntax and semantics are exercised to uncover any navigation errors (e.g., dead links, improper links,

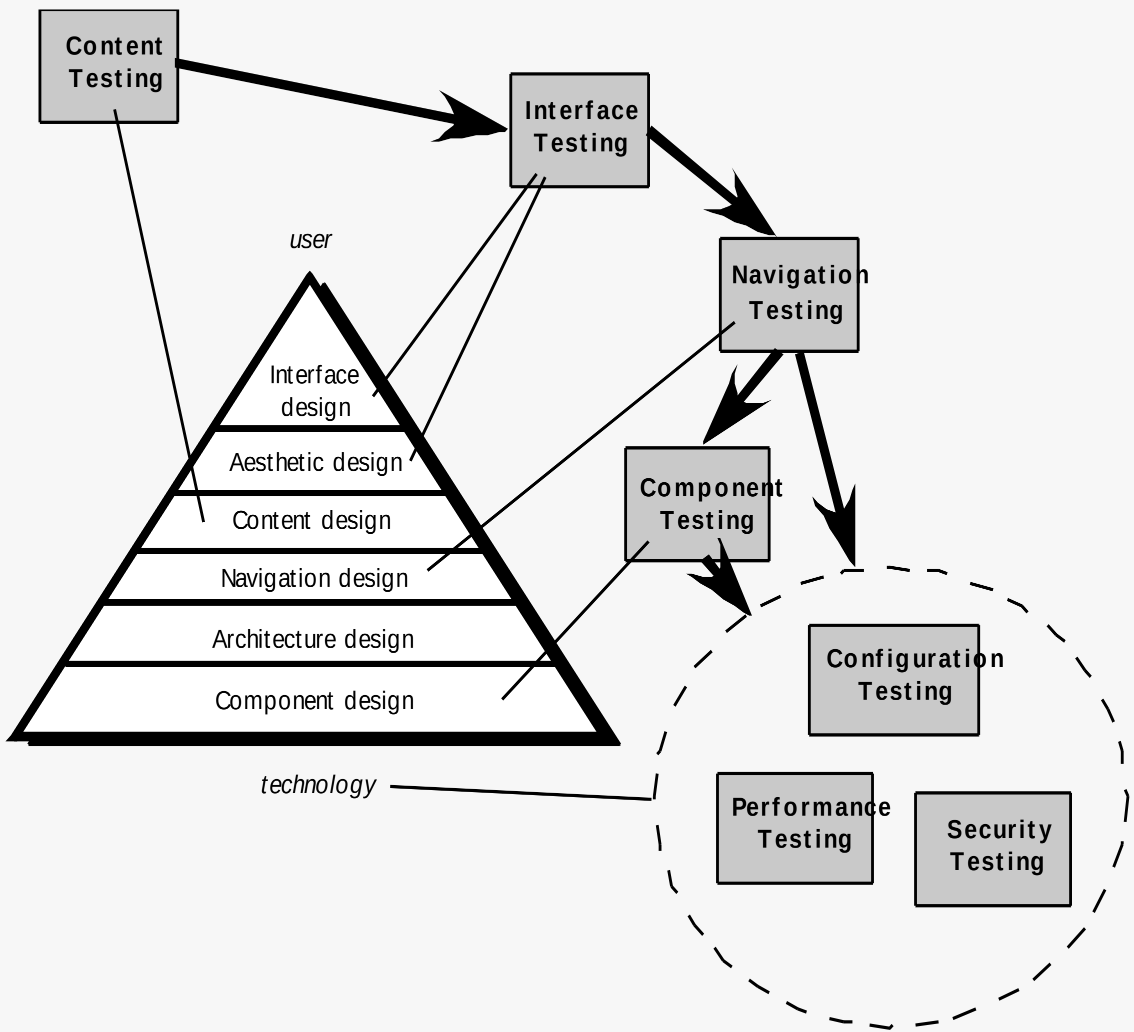
Testing Web Applications

Testing Quality Dimensions-II

- ***Performance*** is tested under a variety of operating conditions, configurations, and loading to ensure that
  – the system is responsive to user interaction
  – the system handles extreme loading without unacceptable operational degradation
- ***Compatibility*** is tested by executing the WebApp in a variety of different host configurations on both the client and server sides.
  – The intent is to find errors that are specific to a unique host configuration.
- ***Interoperability*** is tested to ensure that the WebApp properly interfaces with other applications and/or databases.
- ***Security*** is tested by assessing potential vulnerabilities and attempting to exploit each.
  – Any successful penetration attempt is deemed a security failure.

Testing Web Applications

The Testing Process



People
Innovation
Excellence

Testing Web Applications

Content Testing

- Content testing has three important objectives:
  - to uncover syntactic errors (e.g., typos, grammar mistakes) in text-based documents, graphical representations, and other media
  - to uncover semantic errors (i.e., errors in the accuracy or completeness of information) in any content object presented as navigation occurs, and
  - to find errors in the organization or structure of content that is presented to the end user

Testing Web Applications

Assessing Content Semantics

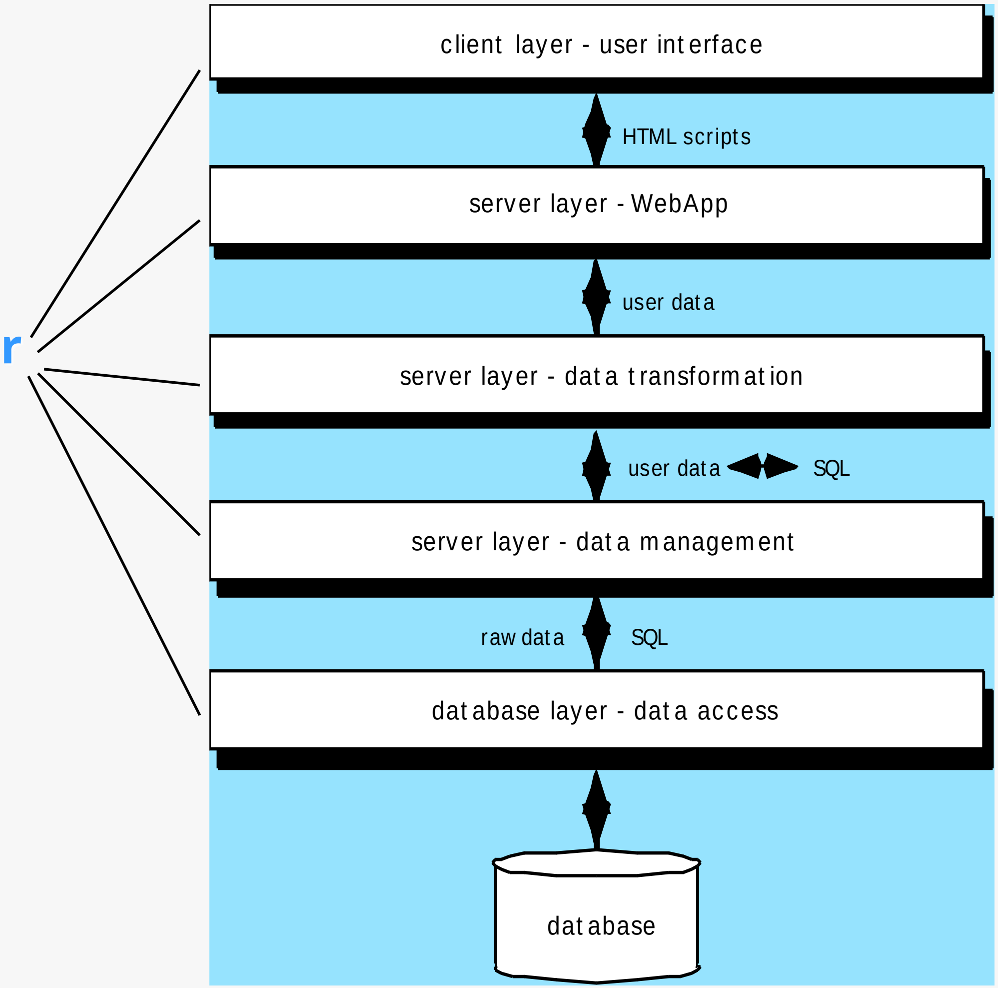
- Is the information factually accurate?
- Is the information concise and to the point?
- Is the layout of the content object easy for the user to understand?
- Can information embedded within a content object be found easily?
- Have proper references been provided for all information derived from other sources?
- Is the information presented consistent internally and consistent with information presented in other content objects?
- Is the content offensive, misleading, or does it open the door to litigation?
- Does the content infringe on existing copyrights or trademarks?
- Does the content contain internal links that supplement existing content? Are the links correct?
- Does the aesthetic style of the content conflict with the aesthetic style of the interface?

People
Innovation
Excellence

Testing Web Applications

Database Testing

Tests are defined for each layer



People
Innovation
Excellence

Testing Web Applications

People
Innovation
Excellence

User Interface Testing

- Interface features are tested to ensure that design rules, aesthetics, and related visual content is available for the user without error.
- Individual interface mechanisms are tested in a manner that is analogous to unit testing.
- Each interface mechanism is tested within the context of a use-case or NSU for a specific user category.
- The complete interface is tested against selected use-cases and NSUs to uncover errors in the semantics of the interface.
- The interface is tested within a variety of environments (e.g., browsers) to ensure that it will be compatible

Testing Mobile Applications

Testing Guidelines

- *Understand the network and device landscape before testing to identify bottlenecks*
- *Conduct tests in uncontrolled real-world test condition (filed-based testing)*
- *Select the right automation test tool*
- *Use the Weighted Device Platform Matrix method to identify the most critical hardware/platform combination to test*
- *Check the end-to-end functional flow in all possible platforms at least once*
- *Conduct performance testing, GUI testing, and compatibility testing using actual devices*
- *Measure performance only in realistic conditions of wireless and user load*

People
Innovation
Excellence

Testing Mobile Applications

Testing Strategies

- Developing a MobileApp testing strategy requires an understanding of both software testing and the challenges that make mobile devices and their network infrastucture unique.
- In addition to a thorough knowledge of conventional software testing approach, a MobileApp tester should have a good understanding of telecommunications principles and an awarness of the differences and capabilities of mobile operating systems platforms.
- This basic knowledge must be complemented with a thorough understanding of the different types of mobile testing (e.g. MobileApp testing, mobile handset testing, mobile website testing), the use of simulators, test automations tools, and remote data access services (RDA).

Testing Mobile Applications

BINUS UNIVERSITY

People
Innovation
Excellence

Criteria Testing Tools and Environments

- Object identification
- Security
- Devices
- Functionality
- Emulators and plug-ins
- Connectivity

Security Engineering

Analyzing Security Requirements

- An important part of building secure systems is anticipating conditions or threats that may be used to damage system resources or render them inaccessible to authorized users.
- This process is called *threat analysis.*
- Once the system assets, vulnerabilities, and threats have been identified, controls can be created to either avoid attacks or mitigate their damage.

Security Engineering

Analyzing Security Requirements

- Software security is an essential prerequisite for software integrity, availability, reliability, and safety
- It may not possible to create a system that can be defend its assets againt all possible threats, and for that reason, it may be necessary to encourage users to maintain backup copies of critical data, redundant system component, and ensure privacy controls are in place

People
Innovation
Excellence

Security Engineering

Security and Privacy in an Online World

- Social Media
- Mobile Applications
- Cloud Computing
- The Internet of Things

Security Engineering

Security Risk Analysis

- Identify assets
- Create an architecture overview
- Decompose the application
- Identify threats
- Documented the threats
- Rate the threats

References

- Pressman, R.S. (2015). ***Software Engineering : A Practioner's Approach. 8th ed***. McGraw-Hill Companies.Inc, Americas, New York. ISBN : 978 1 259 253157.
- **Software Testing, http://www.youtube.com/watch?v=caElFKbceP0&list=PLED41984073D5B532**
- **Conventional sw testing** http://www.testingexcellence.com/conventional-software-testing-on-an-extreme-programming-team/
- **Testing OO SW** http://diwww.epfl.ch/researchlgl/research/ongoing/testing.html
- **Web testing** http://www.manageengine.com/products/qengine/web-testing.html

People
Innovation
Excellence

Q & A

People
Innovation
Excellence