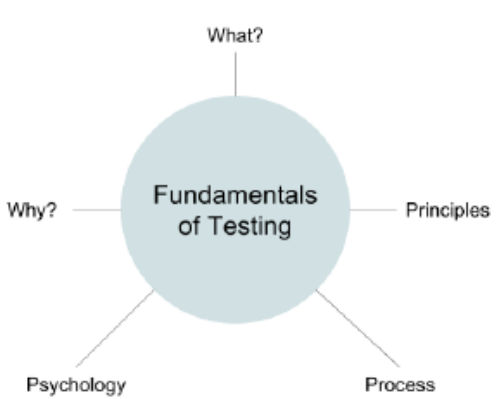
IN3240/IN4240 Software Testing

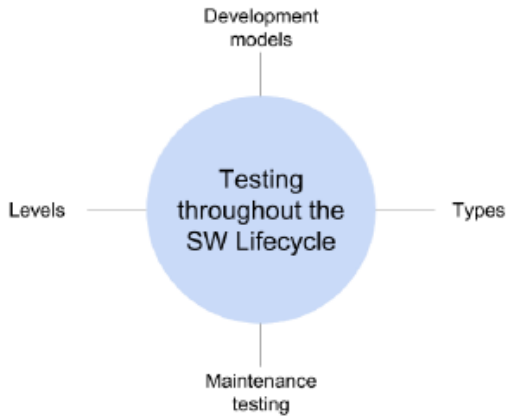
Test Design Techniques

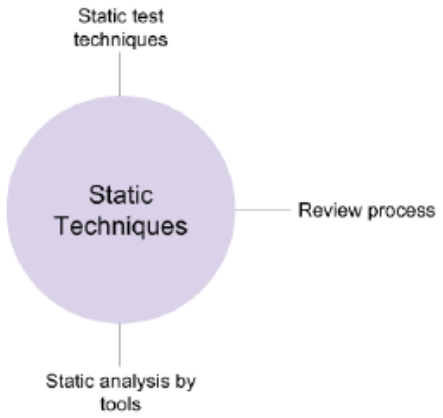
Chapter 4 – Part 1

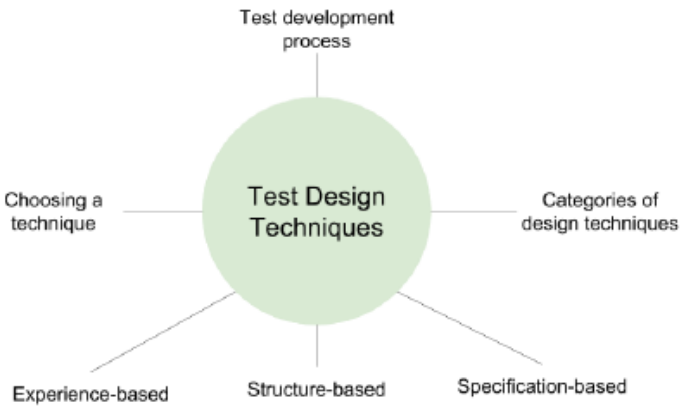
1. The test development process
2. Categories of test design techniques
3. Specification-based testing (Black-box)

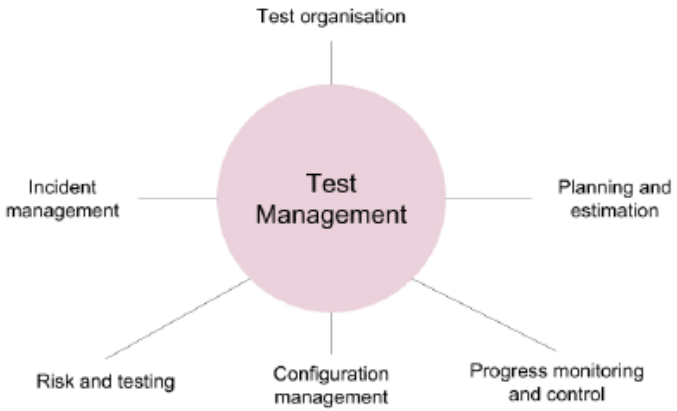
An Overview

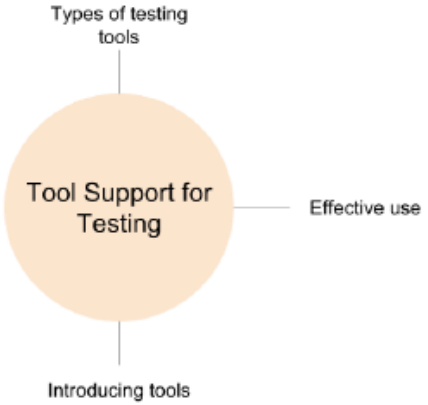

What?
Why?
Fundamentals of Testing
Principles
Psychology
Process

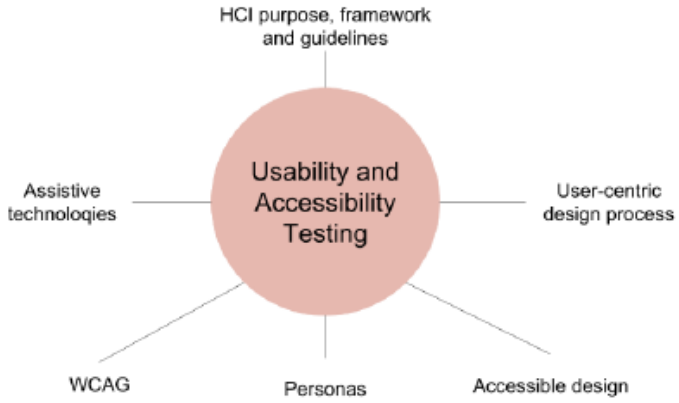

Development models
Levels
Testing throughout the SW Lifecycle
Types
Maintenance testing

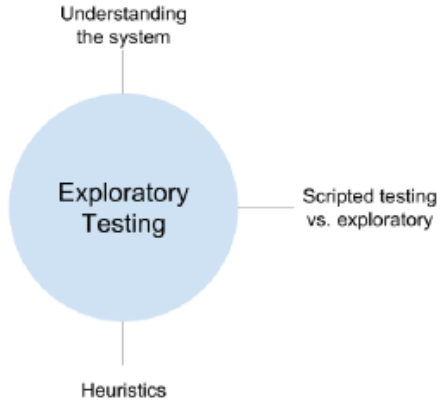

Static test techniques
Static Techniques
Review process
Static analysis by tools


Test development process
Choosing a technique
Test Design Techniques
Categories of design techniques
Experience-based
Structure-based
Specification-based


Test organisation
Incident management
Test Management
Planning and estimation
Risk and testing
Configuration management
Progress monitoring and control


Types of testing tools
Tool Support for Testing
Effective use
Introducing tools


HCI purpose, framework and guidelines
Assistive technologies
Usability and Accessibility Testing
User-centric design process
WCAG
Personas
Accessible design


Understanding the system
Exploratory Testing
Scripted testing vs. exploratory
Heuristics

Testing techniques

Recall the difference between static and dynamic test techniques:

Static testing



- **Manual examination and automated** analysis of code without executing it
- Can be applied to other work products

Dynamic testing



- Requires source code to be executed

1. The test development process
- *1.1 Background*
- *1.2 Test analysis*
- *1.3 Test design*
- *1.4 Test implementation*

2. Categories of test design techniques

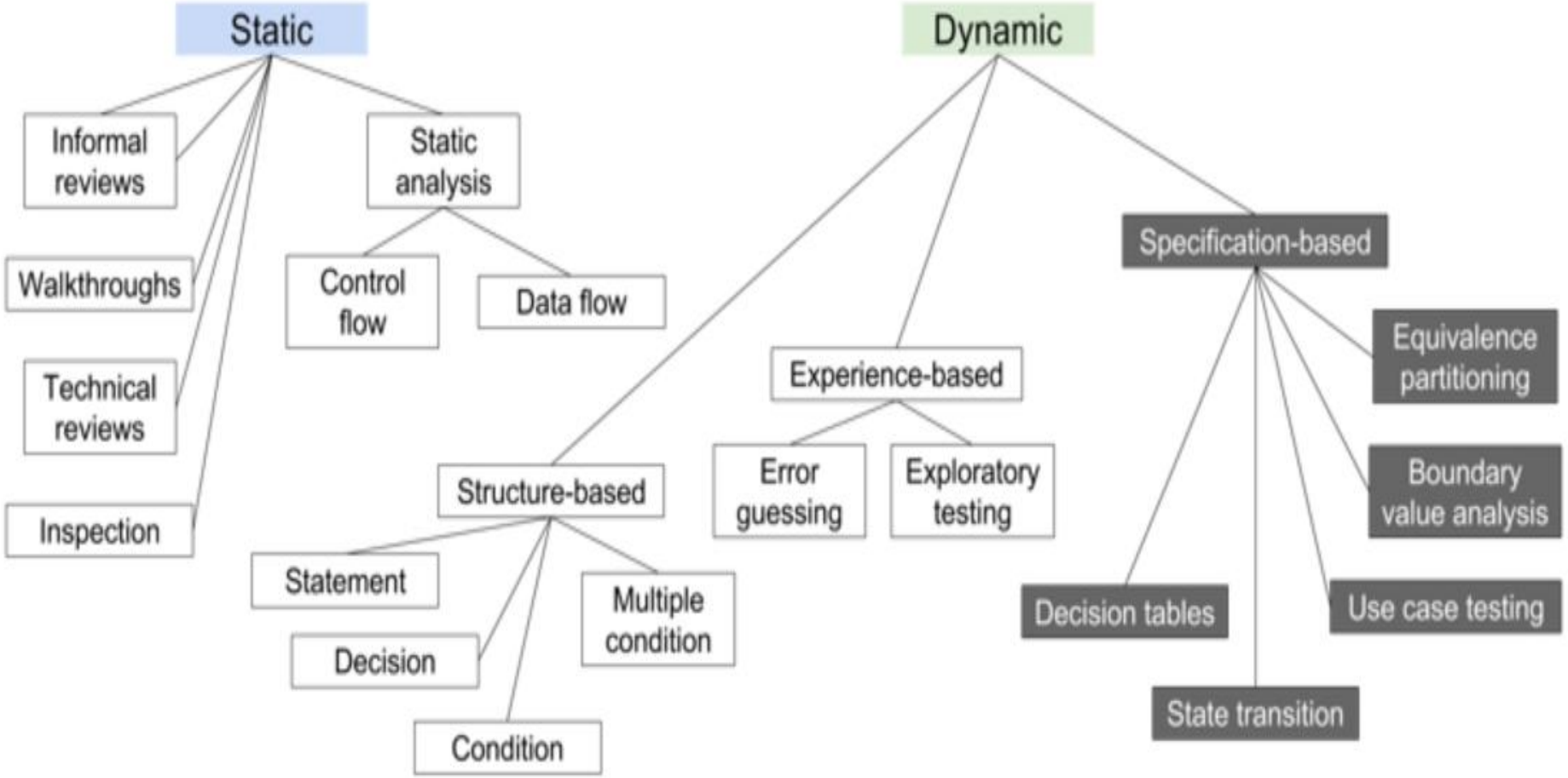
3. Specification-based techniques (black-box)
- *3.1 Equivalence partitioning*
- *3.2 Boundary value analysis*
- *3.3 Decision table testing*
- *3.4 State transition testing*
- *3.5 Use case testing*

4. Structure-based techniques (white-box)
- *4.1 Statement testing and coverage*
- *4.2 Decision testing and coverage*
- *4.3 Other structure-based techniques*

5. Experience-based techniques

6. Choosing test techniques

Categories of test design techniques



1. The test development process
- *1.1 Background*
- *1.2 Test analysis*
- *1.3 Test design*
- *1.4 Test implementation*

2. Categories of test design techniques

3. Specification-based techniques (black-box)
- *3.1 Equivalence partitioning*
- *3.2 Boundary value analysis*
- *3.3 Decision table testing*
- *3.4 State transition testing*
- *3.5 Use case testing*

4. Structure-based techniques (white-box)
- *4.1 Statement testing and coverage*
- *4.2 Decision testing and coverage*
- *4.3 Other structure-based techniques*

5. Experience-based techniques

6. Choosing test techniques

Background

Before we start testing, we need to know

- What are we trying to test?
- What are the inputs?
- What are the results that should be produced by those inputs?
- How do we prepare the tests?
- How do we run the tests?

To answer these questions, we will look at

- Test conditions
- Test cases
- Test procedures

1. The test development process
- 1.1 Background
- 1.2 Test analysis
- 1.3 Test design
- 1.4 Test implementation

2. Categories of test design techniques

3. Specification-based techniques (black-box)
- 3.1 Equivalence partitioning
- 3.2 Boundary value analysis
- 3.3 Decision table testing
- 3.4 State transition testing
- 3.5 Use case testing

4. Structure-based techniques (white-box)
- 4.1 Statement testing and coverage
- 4.2 Decision testing and coverage
- 4.3 Other structure-based techniques

5. Experience-based techniques

6. Choosing test techniques

Background

The test design process can be done in different ways, from very informal (little or no documentation), to very formal.
The level of formality depends on the context of the testing, including:

- The maturity of testing process
- The maturity of development process
- The organization
- Time constraints
- People involved

1. The test development process
- *1.1 Background*
- *1.2 Test analysis*
- *1.3 Test design*
- *1.4 Test implementation*

2. Categories of test design techniques

3. Specification-based techniques (black-box)
- *3.1 Equivalence partitioning*
- *3.2 Boundary value analysis*
- *3.3 Decision table testing*
- *3.4 State transition testing*
- *3.5 Use case testing*

4. Structure-based techniques (white-box)
- *4.1 Statement testing and coverage*
- *4.2 Decision testing and coverage*
- *4.3 Other structure-based techniques*

5. Experience-based techniques

6. Choosing test techniques

Test development

1. Test analysis

2. Test design

3. Test implementation

1. The test development process
- *1.1 Background*
- *1.2 Test analysis*
- *1.3 Test design*
- *1.4 Test implementation*

2. Categories of test design techniques

3. Specification-based techniques (black-box)
- *3.1 Equivalence partitioning*
- *3.2 Boundary value analysis*
- *3.3 Decision table testing*
- *3.4 State transition testing*
- *3.5 Use case testing*

4. Structure-based techniques (white-box)
- *4.1 Statement testing and coverage*
- *4.2 Decision testing and coverage*
- *4.3 Other structure-based techniques*

5. Experience-based techniques

6. Choosing test techniques

1. Test analysis

The test basis documentation is analyzed in order to determine *what* to test, i.e. to *identify the test conditions*.

Test condition *(Def.)* = an item or event that could be verified by one or more test cases

1. The test development process
- *1.1 Background*
- *1.2 Test analysis*
- *1.3 Test design*
- *1.4 Test implementation*

2. Categories of test design techniques

3. Specification-based techniques (black-box)
- *3.1 Equivalence partitioning*
- *3.2 Boundary value analysis*
- *3.3 Decision table testing*
- *3.4 State transition testing*
- *3.5 Use case testing*

4. Structure-based techniques (white-box)
- *4.1 Statement testing and coverage*
- *4.2 Decision testing and coverage*
- *4.3 Other structure-based techniques*

5. Experience-based techniques

6. Choosing test techniques

1. Test analysis

Examples of test conditions

- A function
- A transaction
- Business rules
- Other structural elements (menus in web pages, etc.)

1. The test development process
- *1.1 Background*
- *1.2 Test analysis*
- *1.3 Test design*
- *1.4 Test implementation*

2. Categories of test design techniques

3. Specification-based techniques (black-box)
- *3.1 Equivalence partitioning*
- *3.2 Boundary value analysis*
- *3.3 Decision table testing*
- *3.4 State transition testing*
- *3.5 Use case testing*

4. Structure-based techniques (white-box)
- *4.1 Statement testing and coverage*
- *4.2 Decision testing and coverage*
- *4.3 Other structure-based techniques*

5. Experience-based techniques

6. Choosing test techniques

Test possibilities

“Throw a wide net!”

First; identify as many test conditions as possible

Second; select which one to develop in more detail

We can't test everything (P2). We must select a subset of all possible tests, but this subset must have a high probability of finding most of the defects in the system.

We need a suitable test design technique to guide our selection and to prioritize the test conditions.

1. The test development process
- *1.1 Background*
- *1.2 Test analysis*
- *1.3 Test design*
- *1.4 Test implementation*

2. Categories of test design techniques

3. Specification-based techniques (black-box)
- *3.1 Equivalence partitioning*
- *3.2 Boundary value analysis*
- *3.3 Decision table testing*
- *3.4 State transition testing*
- *3.5 Use case testing*

4. Structure-based techniques (white-box)
- *4.1 Statement testing and coverage*
- *4.2 Decision testing and coverage*
- *4.3 Other structure-based techniques*

5. Experience-based techniques

6. Choosing test techniques

Test possibilities

“Throw a wide net!”

First; identify as many test conditions as possible

Second; select which one to develop in more detail

We can't test everything (P2). We must select a subset of all possible tests, but this subset must have a high probability of finding most of the defects in the system.

We need a suitable test design technique to guide our selection and to prioritize the test conditions.

1. The test development process
- *1.1 Background*
- *1.2 Test analysis*
- *1.3 Test design*
- *1.4 Test implementation*

2. Categories of test design techniques

3. Specification-based techniques (black-box)
- *3.1 Equivalence partitioning*
- *3.2 Boundary value analysis*
- *3.3 Decision table testing*
- *3.4 State transition testing*
- *3.5 Use case testing*

4. Structure-based techniques (white-box)
- *4.1 Statement testing and coverage*
- *4.2 Decision testing and coverage*
- *4.3 Other structure-based techniques*

5. Experience-based techniques

6. Choosing test techniques

Test possibilities

“Throw a wide net!”

First; identify as many test conditions as possible

Second; select which one to develop in more detail

We can't test everything (P2). We must select a subset of all possible tests, but this subset must have a high probability of finding most of the defects in the system.

We need a suitable test design technique to guide our selection and to prioritize the test conditions.

1. The test development process
- *1.1 Background*
- *1.2 Test analysis*
- *1.3 Test design*
- *1.4 Test implementation*

2. Categories of test design techniques

3. Specification-based techniques (black-box)
- *3.1 Equivalence partitioning*
- *3.2 Boundary value analysis*
- *3.3 Decision table testing*
- *3.4 State transition testing*
- *3.5 Use case testing*

4. Structure-based techniques (white-box)
- *4.1 Statement testing and coverage*
- *4.2 Decision testing and coverage*
- *4.3 Other structure-based techniques*

5. Experience-based techniques

6. Choosing test techniques

Test possibilities

“Throw a wide net!”

First; identify as many test conditions as possible

Second; select which one to develop in more detail

We can't test everything (P2). We must select a subset of all possible tests, but this subset must have a high probability of finding most of the defects in the system.

We need a suitable test design technique to guide our selection and to prioritize the test conditions.

1. The test development process
- *1.1 Background*
- *1.2 Test analysis*
- *1.3 Test design*
- *1.4 Test implementation*

2. Categories of test design techniques

3. Specification-based techniques (black-box)
- *3.1 Equivalence partitioning*
- *3.2 Boundary value analysis*
- *3.3 Decision table testing*
- *3.4 State transition testing*
- *3.5 Use case testing*

4. Structure-based techniques (white-box)
- *4.1 Statement testing and coverage*
- *4.2 Decision testing and coverage*
- *4.3 Other structure-based techniques*

5. Experience-based techniques

6. Choosing test techniques

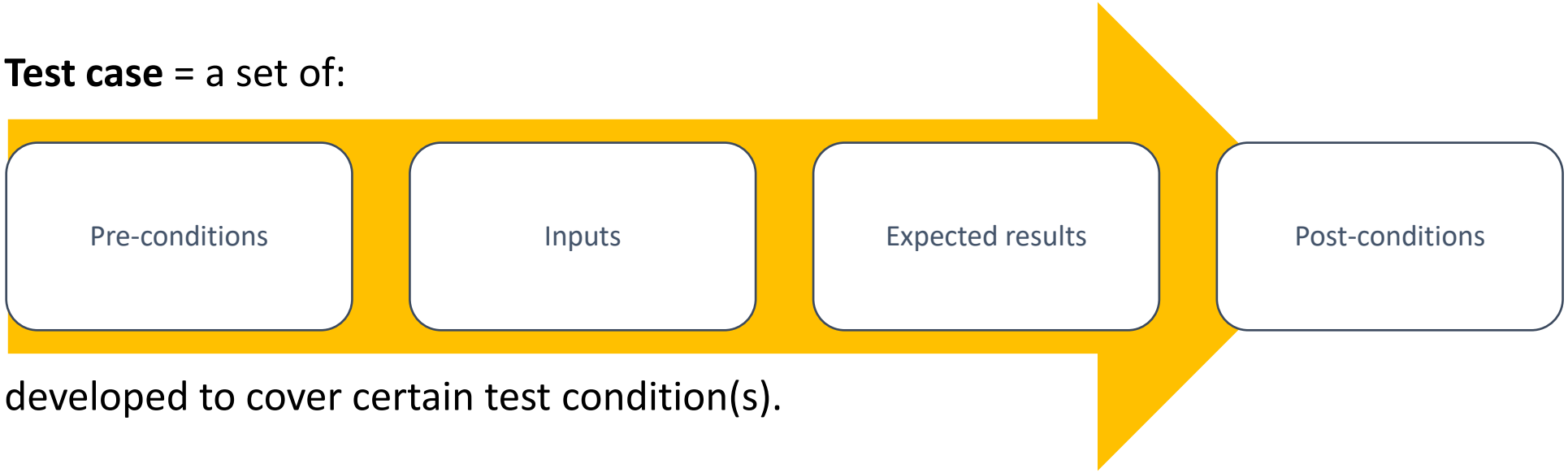
2. Test design

During test design:



are created and specified.

Test case = a set of:



developed to cover certain test condition(s).

1. The test development process
- 1.1 Background
- 1.2 Test analysis
- 1.3 Test design
- 1.4 Test implementation

2. Categories of test design techniques

3. Specification-based techniques (black-box)
- 3.1 Equivalence partitioning
- 3.2 Boundary value analysis
- 3.3 Decision table testing
- 3.4 State transition testing
- 3.5 Use case testing

4. Structure-based techniques (white-box)
- 4.1 Statement testing and coverage
- 4.2 Decision testing and coverage
- 4.3 Other structure-based techniques

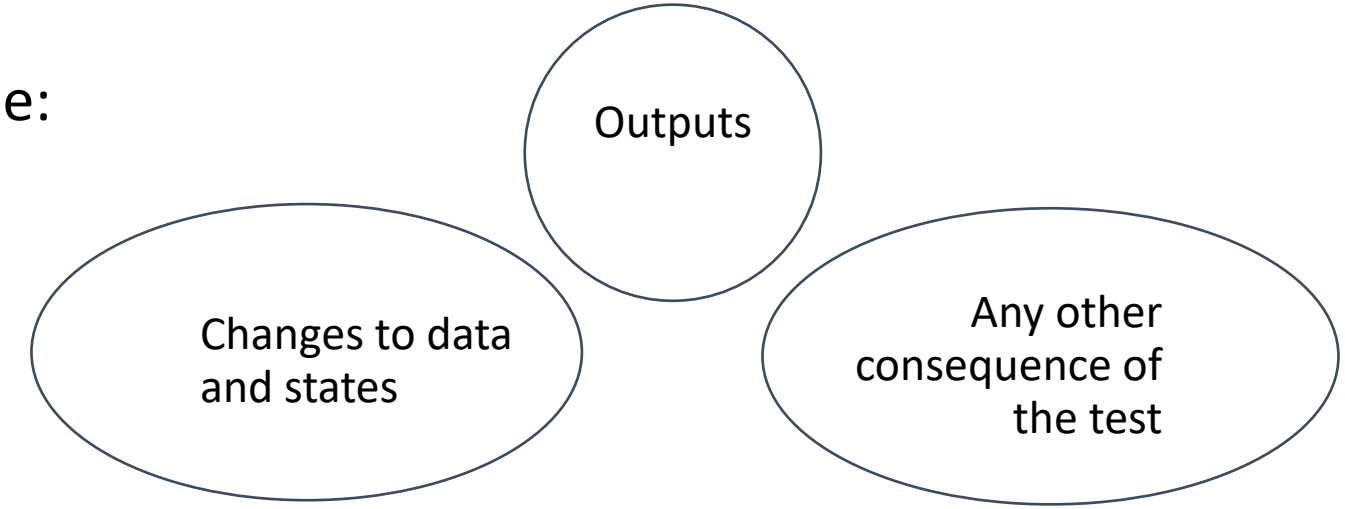
5. Experience-based techniques

6. Choosing test techniques

Test oracle

To know what the system *should* do we need to have a source of information about the correct behavior of the system – an oracle.

Expected results include:



If expected results have not been defined, then a plausible but erroneous result may be interpreted as the correct one.

Expected results should ideally be defined prior to test execution.

1. The test development process
- *1.1 Background*
- *1.2 Test analysis*
- *1.3 Test design*
- *1.4 Test implementation*

2. Categories of test design techniques

3. Specification-based techniques (black-box)
- *3.1 Equivalence partitioning*
- *3.2 Boundary value analysis*
- *3.3 Decision table testing*
- *3.4 State transition testing*
- *3.5 Use case testing*

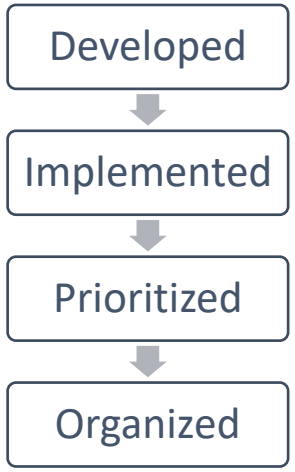
4. Structure-based techniques (white-box)
- *4.1 Statement testing and coverage*
- *4.2 Decision testing and coverage*
- *4.3 Other structure-based techniques*

5. Experience-based techniques

6. Choosing test techniques

3. Test implementation

During test implementation the test cases are organized in the test procedures:



A manual test procedure
specifies the sequence of action to be taken for executing of a test.

An automated test procedure (test script)
If tests are run using a test execution tool, the sequence of action is specified in a test script.

1. The test development process
- *1.1 Background*
- *1.2 Test analysis*
- *1.3 Test design*
- *1.4 Test implementation*

2. Categories of test design techniques

3. Specification-based techniques (black-box)
- *3.1 Equivalence partitioning*
- *3.2 Boundary value analysis*
- *3.3 Decision table testing*
- *3.4 State transition testing*
- *3.5 Use case testing*

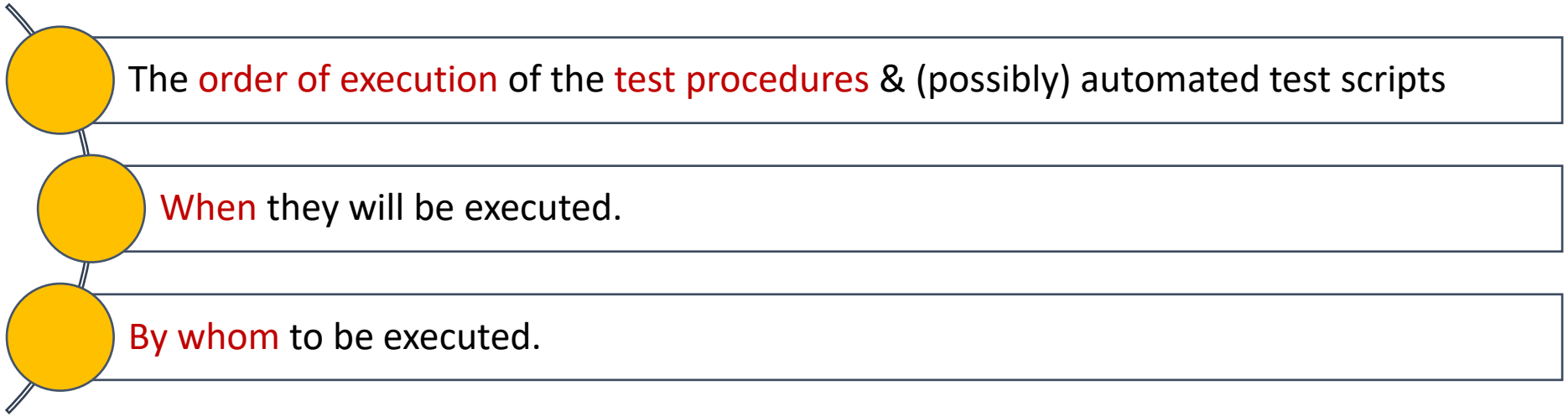
4. Structure-based techniques (white-box)
- *4.1 Statement testing and coverage*
- *4.2 Decision testing and coverage*
- *4.3 Other structure-based techniques*

5. Experience-based techniques

6. Choosing test techniques

3. Test implementation

The test execution schedule defines:



The test execution schedule will take into account such factors as:

- risks
- regression tests
- prioritization
- technical and logical dependencies

1. The test development process
- *1.1 Background*
- *1.2 Test analysis*
- *1.3 Test design*
- *1.4 Test implementation*

2. Categories of test design techniques

3. Specification-based techniques (black-box)
- *3.1 Equivalence partitioning*
- *3.2 Boundary value analysis*
- *3.3 Decision table testing*
- *3.4 State transition testing*
- *3.5 Use case testing*

4. Structure-based techniques (white-box)
- *4.1 Statement testing and coverage*
- *4.2 Decision testing and coverage*
- *4.3 Other structure-based techniques*

5. Experience-based techniques

6. Choosing test techniques

3. Test implementation

Writing the test procedure is another opportunity to prioritize the tests, to ensure that the best testing is done in the time available.

A good **rule of thumb** is 'Find the scary stuff first'. However, the definition of what is 'scary' depends on the business, system or project and on the risks of the project.

1. The test development process
- *1.1 Background*
- *1.2 Test analysis*
- *1.3 Test design*
- *1.4 Test implementation*

2. Categories of test design techniques

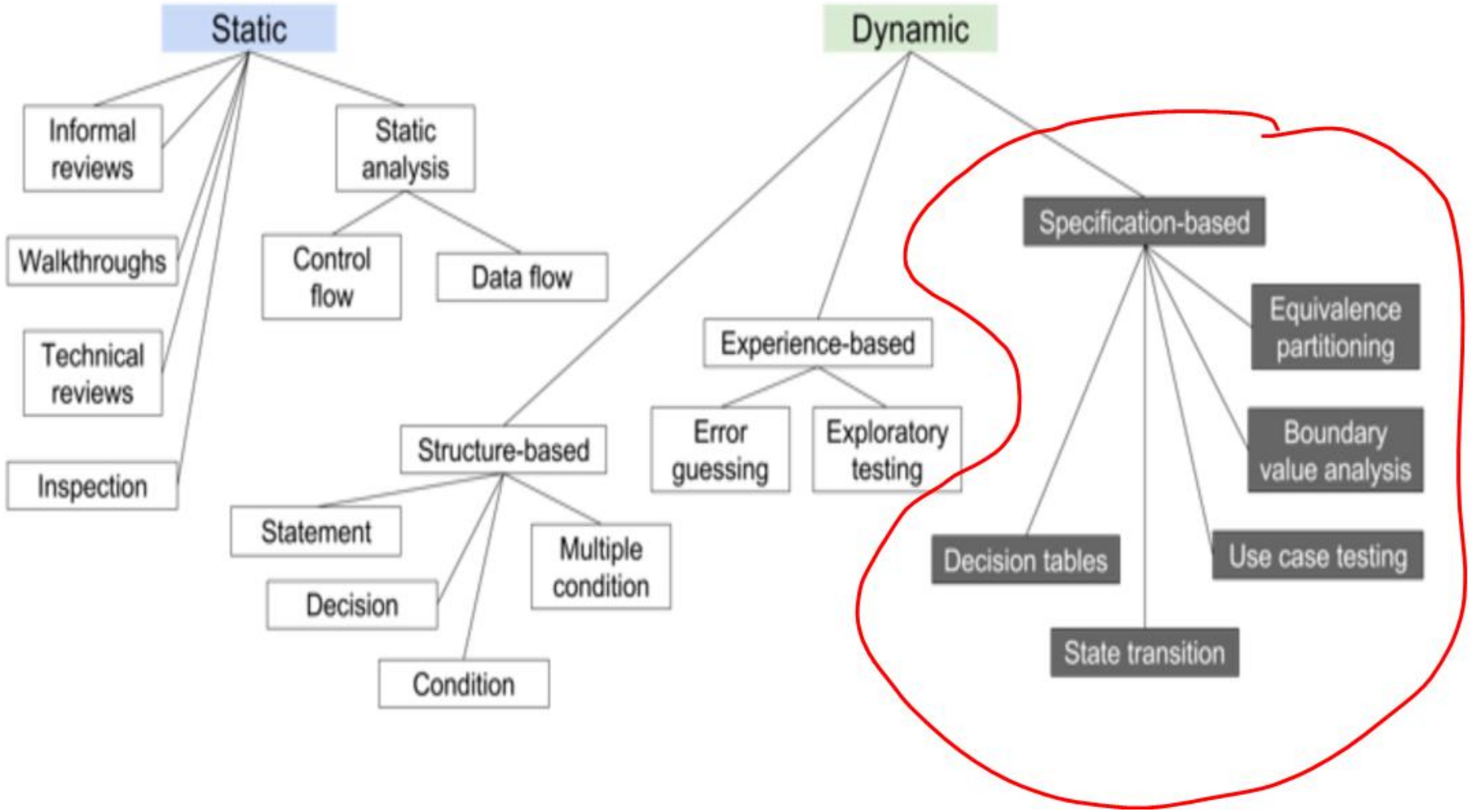
3. Specification-based techniques (black-box)
- *3.1 Equivalence partitioning*
- *3.2 Boundary value analysis*
- *3.3 Decision table testing*
- *3.4 State transition testing*
- *3.5 Use case testing*

4. Structure-based techniques (white-box)
- *4.1 Statement testing and coverage*
- *4.2 Decision testing and coverage*
- *4.3 Other structure-based techniques*

5. Experience-based techniques

6. Choosing test techniques

Categories of test design techniques



1. The test development process
- *1.1 Background*
- *1.2 Test analysis*
- *1.3 Test design*
- *1.4 Test implementation*

2. Categories of test design techniques

3. Specification-based techniques (black-box)
- *3.1 Equivalence partitioning*
- *3.2 Boundary value analysis*
- *3.3 Decision table testing*
- *3.4 State transition testing*
- *3.5 Use case testing*

4. Structure-based techniques (white-box)
- *4.1 Statement testing and coverage*
- *4.2 Decision testing and coverage*
- *4.3 Other structure-based techniques*

5. Experience-based techniques

6. Choosing test techniques

Common features of black-box techniques

- We use models (formal or informal) to specify the problem to be solved, as well as the software or its components.
- We derive systematically the test cases from these models.

1. The test development process
- *1.1 Background*
- *1.2 Test analysis*
- *1.3 Test design*
- *1.4 Test implementation*

2. Categories of test design techniques

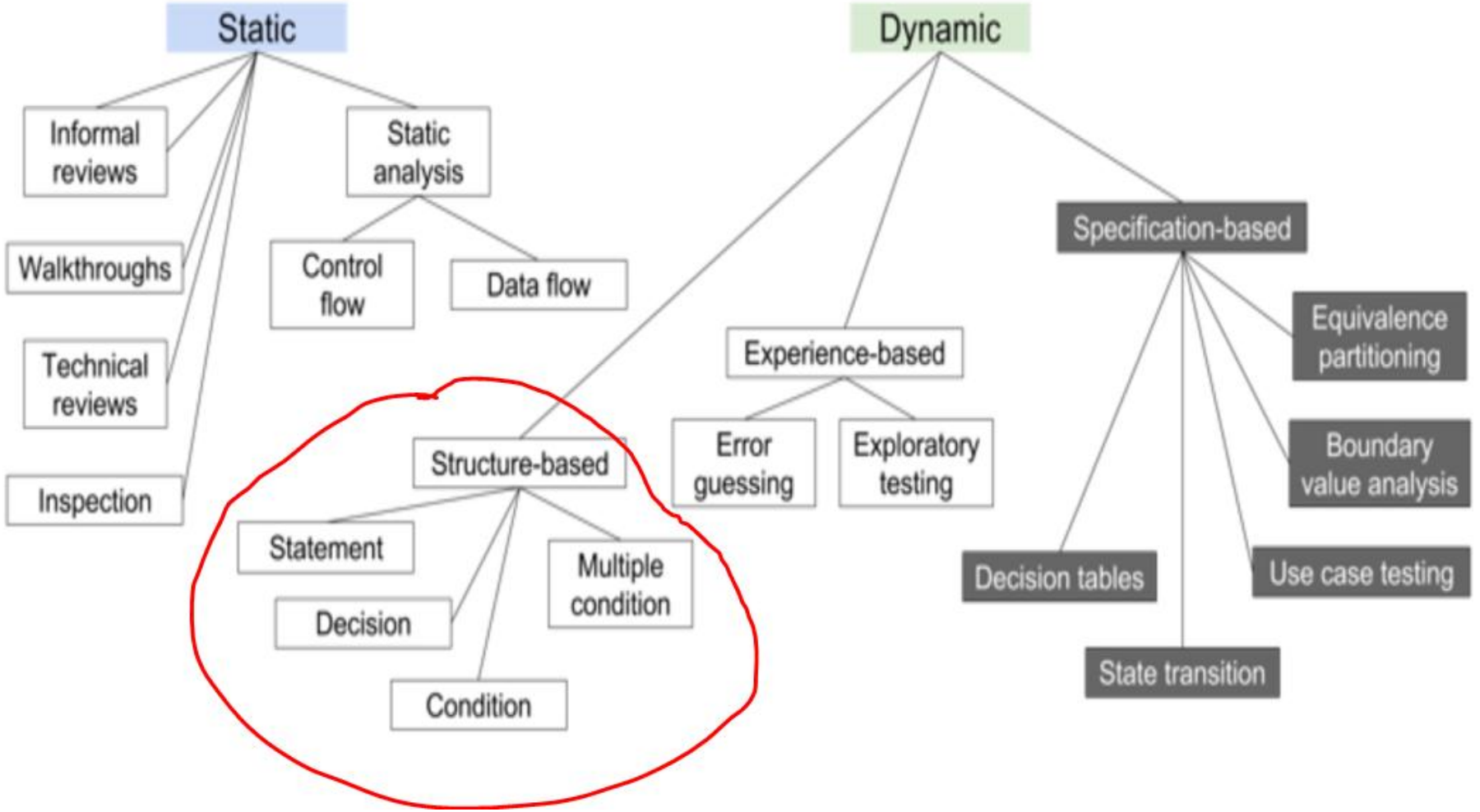
3. Specification-based techniques (black-box)
- *3.1 Equivalence partitioning*
- *3.2 Boundary value analysis*
- *3.3 Decision table testing*
- *3.4 State transition testing*
- *3.5 Use case testing*

4. Structure-based techniques (white-box)
- *4.1 Statement testing and coverage*
- *4.2 Decision testing and coverage*
- *4.3 Other structure-based techniques*

5. Experience-based techniques

6. Choosing test techniques

Categories of test design techniques



1. The test development process
- *1.1 Background*
- *1.2 Test analysis*
- *1.3 Test design*
- *1.4 Test implementation*

2. Categories of test design techniques

3. Specification-based techniques (black-box)
- *3.1 Equivalence partitioning*
- *3.2 Boundary value analysis*
- *3.3 Decision table testing*
- *3.4 State transition testing*
- *3.5 Use case testing*

4. Structure-based techniques (white-box)
- *4.1 Statement testing and coverage*
- *4.2 Decision testing and coverage*
- *4.3 Other structure-based techniques*

5. Experience-based techniques

6. Choosing test techniques

Common features of white-box techniques

- Serve two purposes: coverage measurements and structural test design.
- For the existing black box test cases, we can measure the test coverage of the software using tools. Examples: Statement, branch, path coverage.
- Further test cases can be derived from the internal structure of the software (e.g., control flow and logic), requiring knowledge of the implementation.
- Structural analysis helps identify defects such as unreachable code, missing decision outcomes, and unintended logic paths.
- Further test cases (including black-box/specification based) can in this way be derived systematically to increase the test coverage.

1. The test development process
- *1.1 Background*
- *1.2 Test analysis*
- *1.3 Test design*
- *1.4 Test implementation*

2. Categories of test design techniques

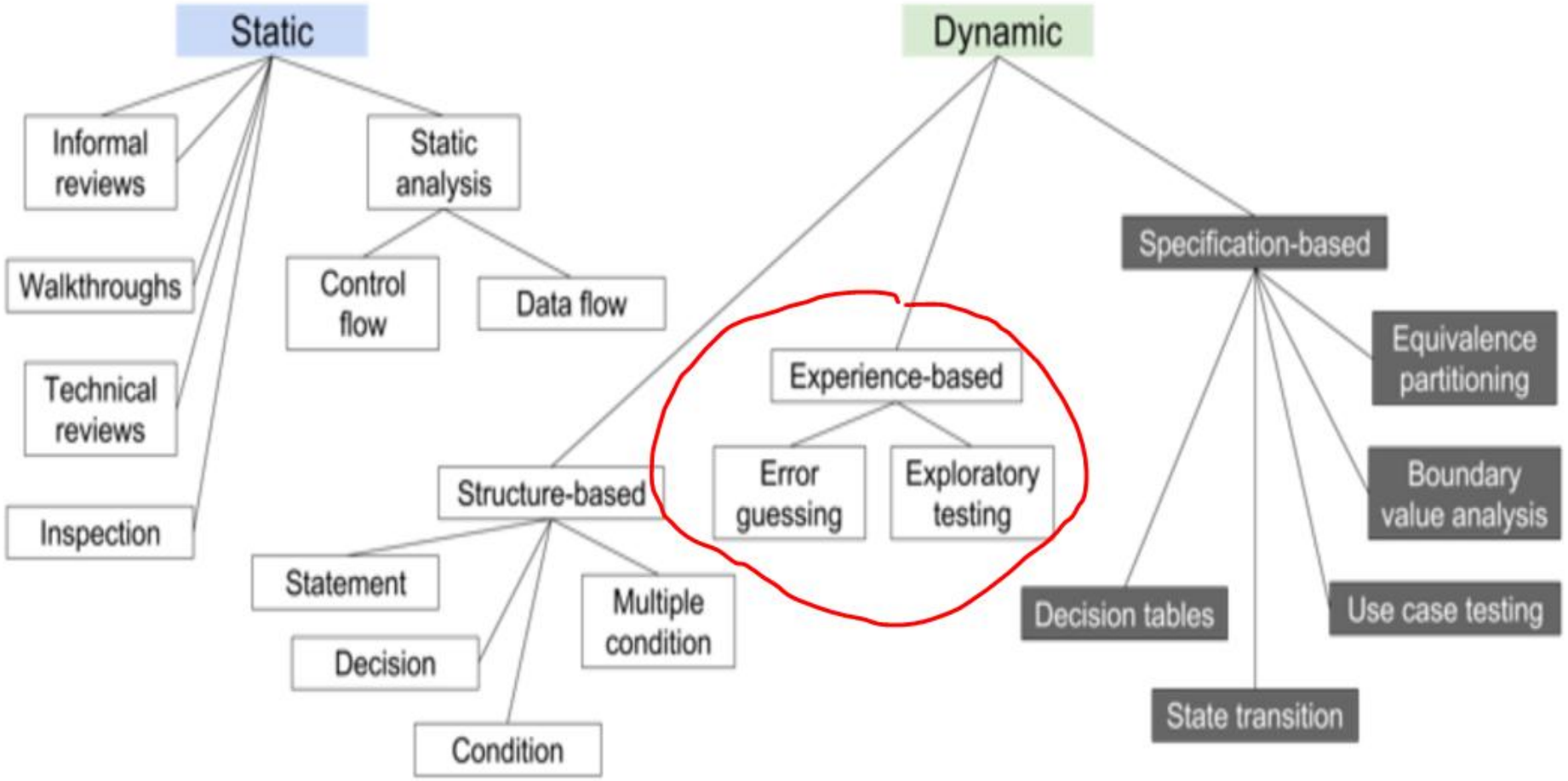
3. Specification-based techniques (black-box)
- *3.1 Equivalence partitioning*
- *3.2 Boundary value analysis*
- *3.3 Decision table testing*
- *3.4 State transition testing*
- *3.5 Use case testing*

4. Structure-based techniques (white-box)
- *4.1 Statement testing and coverage*
- *4.2 Decision testing and coverage*
- *4.3 Other structure-based techniques*

5. Experience-based techniques

6. Choosing test techniques

Categories of test design techniques



1. The test development process
- *1.1 Background*
- *1.2 Test analysis*
- *1.3 Test design*
- *1.4 Test implementation*

2. Categories of test design techniques

3. Specification-based techniques (black-box)
- *3.1 Equivalence partitioning*
- *3.2 Boundary value analysis*
- *3.3 Decision table testing*
- *3.4 State transition testing*
- *3.5 Use case testing*

4. Structure-based techniques (white-box)
- *4.1 Statement testing and coverage*
- *4.2 Decision testing and coverage*
- *4.3 Other structure-based techniques*

5. Experience-based techniques

6. Choosing test techniques

Common features of experience-based techniques

The test cases are derived from the knowledge and experience of people:

- Knowledge of testers, developers, users and other stakeholders about the software, its usage and its environment.

- Knowledge about likely defects and their distribution.

1. The test development process

- *1.1 Background*
- *1.2 Test analysis*
- *1.3 Test design*
- *1.4 Test implementation*

2. Categories of test design techniques

3. Specification-based techniques (black-box)

- *3.1 Equivalence partitioning*
- *3.2 Boundary value analysis*
- *3.3 Decision table testing*
- *3.4 State transition testing*
- *3.5 Use case testing*

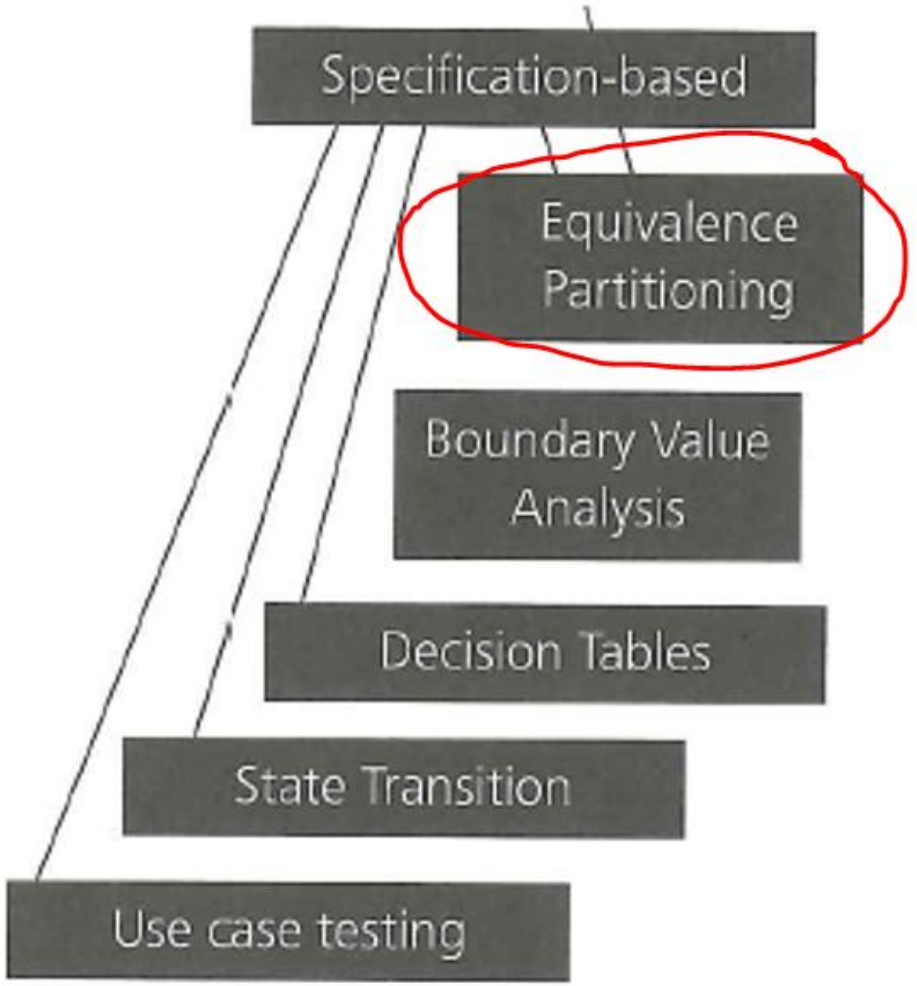
4. Structure-based techniques (white-box)

- *4.1 Statement testing and coverage*
- *4.2 Decision testing and coverage*
- *4.3 Other structure-based techniques*

5. Experience-based techniques

6. Choosing test techniques

Black-box techniques



1. The test development process
- *1.1 Background*
- *1.2 Test analysis*
- *1.3 Test design*
- *1.4 Test implementation*

2. Categories of test design techniques

3. Specification-based techniques (black-box)
- *3.1 Equivalence partitioning*
- *3.2 Boundary value analysis*
- *3.3 Decision table testing*
- *3.4 State transition testing*
- *3.5 Use case testing*

4. Structure-based techniques (white-box)
- *4.1 Statement testing and coverage*
- *4.2 Decision testing and coverage*
- *4.3 Other structure-based techniques*

5. Experience-based techniques

6. Choosing test techniques

Equivalence partitioning

- The **basic idea** is to divide a set of test conditions into partitions (subsets) where the elements in each partition (subset) can be considered the same.
- It is important that the different partitions do not have common elements.
- We need only to test one condition from each partition, because all the conditions in the same partition will be treated in the same way by the software.

1. The test development process
- *1.1 Background*
- *1.2 Test analysis*
- *1.3 Test design*
- *1.4 Test implementation*

2. Categories of test design techniques

3. Specification-based techniques (black-box)
- *3.1 Equivalence partitioning*
- *3.2 Boundary value analysis*
- *3.3 Decision table testing*
- *3.4 State transition testing*
- *3.5 Use case testing*

4. Structure-based techniques (white-box)
- *4.1 Statement testing and coverage*
- *4.2 Decision testing and coverage*
- *4.3 Other structure-based techniques*

5. Experience-based techniques

6. Choosing test techniques

Equivalence partitioning

Given a set A, and the subsets A1 and A2 where

$A1 \cup A2 = A$ (union)

$A1 \cap A2 = \emptyset$ (intersection)

i.e. that A1 and A2 together constitute the whole of A, and that A1 and A2 are disjoint sets without common elements.

We then say an A1 and A2 constitute one partition of A.

1. The test development process
- *1.1 Background*
- *1.2 Test analysis*
- *1.3 Test design*
- *1.4 Test implementation*

2. Categories of test design techniques

3. Specification-based techniques (black-box)
- *3.1 Equivalence partitioning*
- *3.2 Boundary value analysis*
- *3.3 Decision table testing*
- *3.4 State transition testing*
- *3.5 Use case testing*

4. Structure-based techniques (white-box)
- *4.1 Statement testing and coverage*
- *4.2 Decision testing and coverage*
- *4.3 Other structure-based techniques*

5. Experience-based techniques

6. Choosing test techniques

Equivalence partitioning

Example

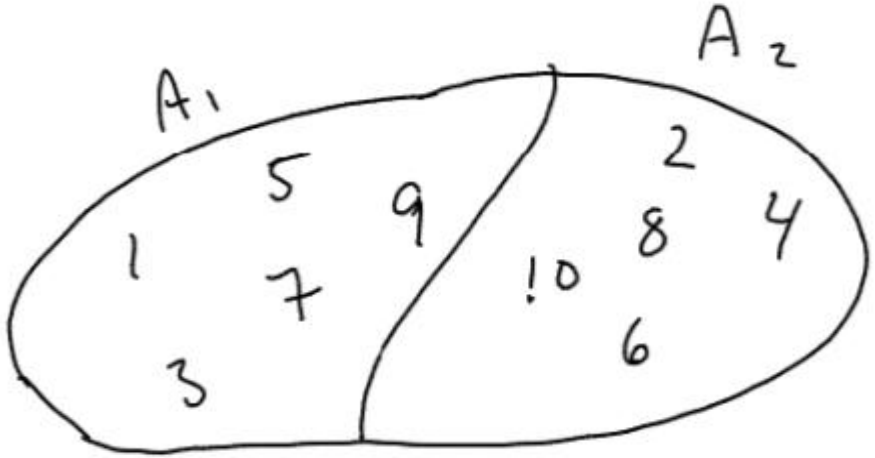
A = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10},

A1 = {1, 3, 5, 7, 9}

A2 = {2, 4, 6, 8, 10}

We see that

A1 ∪ A2 = A and A1 ∩ A2 = Ø.



The subsets A1 and A2 constitute one partition of A

1. The test development process
- *1.1 Background*
- *1.2 Test analysis*
- *1.3 Test design*
- *1.4 Test implementation*

2. Categories of test design techniques

3. Specification-based techniques (black-box)
- *3.1 Equivalence partitioning*
- *3.2 Boundary value analysis*
- *3.3 Decision table testing*
- *3.4 State transition testing*
- *3.5 Use case testing*

4. Structure-based techniques (white-box)
- *4.1 Statement testing and coverage*
- *4.2 Decision testing and coverage*
- *4.3 Other structure-based techniques*

5. Experience-based techniques

6. Choosing test techniques

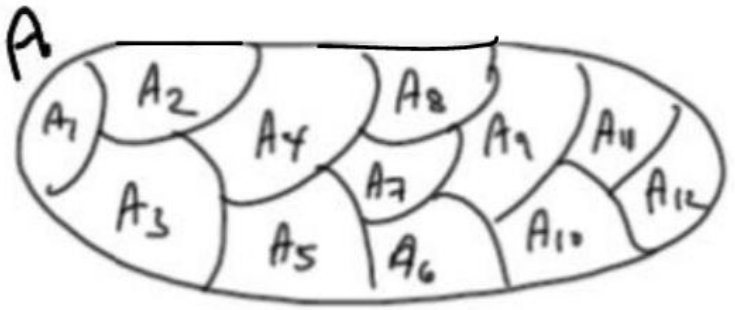
Equivalence partitioning

Given the subsets

$A_1, A_2, A_3, \ldots, A_n$

If $A_1 \cup A_2 \cup A_3 \cup \ldots \cup A_n = A$ and $A_i \cap A_j = \emptyset$ for all $i \neq j$

$A_1, A_2, A_3, \ldots, A_n$ constitute one partition of A



1. The test development process
- *1.1 Background*
- *1.2 Test analysis*
- *1.3 Test design*
- *1.4 Test implementation*

2. Categories of test design techniques

3. Specification-based techniques (black-box)
- *3.1 Equivalence partitioning*
- *3.2 Boundary value analysis*
- *3.3 Decision table testing*
- *3.4 State transition testing*
- *3.5 Use case testing*

4. Structure-based techniques (white-box)
- *4.1 Statement testing and coverage*
- *4.2 Decision testing and coverage*
- *4.3 Other structure-based techniques*

5. Experience-based techniques

6. Choosing test techniques

Equivalence partitioning

Examples

- Students

  bachelor / master

- Tickets

  children / youth / adults / older

- Vehicles

  gasoline / diesel / electric

1. The test development process
- 1.1 Background
- 1.2 Test analysis
- 1.3 Test design
- 1.4 Test implementation

2. Categories of test design techniques

3. Specification-based techniques (black-box)
- 3.1 Equivalence partitioning
- 3.2 Boundary value analysis
- 3.3 Decision table testing
- 3.4 State transition testing
- 3.5 Use case testing

4. Structure-based techniques (white-box)
- 4.1 Statement testing and coverage
- 4.2 Decision testing and coverage
- 4.3 Other structure-based techniques

5. Experience-based techniques

6. Choosing test techniques

Equivalence partitioning

Typical problem

A **savings account** in a bank earns a different rate of interest depending on the balance in the account.
In order to test the software that calculates the interest due, we can identify the ranges of balance values that earn the different rates of interest.
If a balance in the range $0 up to $100.00 has a 3% interest rate, a balance over $100.00 and up to $1000.00 has a 5% interest rate, and balances of $1000.00 and over have a 7% interest rate, we would initially identify three valid equivalence partitions and one invalid partition as shown below.

Invalid partition	Valid (for 3% interest)		Valid (for 5%)		Valid (for 7%)
–$0.01	$0.00	$100.00	$100.01	$999.99	$1000.00

1. The test development process
- *1.1 Background*
- *1.2 Test analysis*
- *1.3 Test design*
- *1.4 Test implementation*

2. Categories of test design techniques

3. Specification-based techniques (black-box)
- *3.1 Equivalence partitioning*
- *3.2 Boundary value analysis*
- *3.3 Decision table testing*
- *3.4 State transition testing*
- *3.5 Use case testing*

4. Structure-based techniques (white-box)
- *4.1 Statement testing and coverage*
- *4.2 Decision testing and coverage*
- *4.3 Other structure-based techniques*

5. Experience-based techniques

6. Choosing test techniques

Equivalence partitioning

We choose to calculate the interest of one balance in each partition:
-$10.00, $50.00, $260.00, $1348.00

Invalid partition	Valid (for 3% interest)		Valid (for 5%)		Valid (for 7%)
–$0.01	$0.00	$100.00	$100.01	$999.99	$1000.00

1. The test development process
- *1.1 Background*
- *1.2 Test analysis*
- *1.3 Test design*
- *1.4 Test implementation*

2. Categories of test design techniques

3. Specification-based techniques (black-box)
- *3.1 Equivalence partitioning*
- *3.2 Boundary value analysis*
- *3.3 Decision table testing*
- *3.4 State transition testing*
- *3.5 Use case testing*

4. Structure-based techniques (white-box)
- *4.1 Statement testing and coverage*
- *4.2 Decision testing and coverage*
- *4.3 Other structure-based techniques*

5. Experience-based techniques

6. Choosing test techniques

Equivalence partitioning

Technique

Inputs/outputs/internal values of the software are divided into groups that are expected to exhibit similar behavior.

1. The test development process
- *1.1 Background*
- *1.2 Test analysis*
- *1.3 Test design*
- *1.4 Test implementation*

2. Categories of test design techniques

3. Specification-based techniques (black-box)
- *3.1 Equivalence partitioning*
- *3.2 Boundary value analysis*
- *3.3 Decision table testing*
- *3.4 State transition testing*
- *3.5 Use case testing*

4. Structure-based techniques (white-box)
- *4.1 Statement testing and coverage*
- *4.2 Decision testing and coverage*
- *4.3 Other structure-based techniques*

5. Experience-based techniques

6. Choosing test techniques

Equivalence partitioning

Technique

Inputs/outputs/internal values of the software are divided into groups that are expected to exhibit similar behavior.

Equivalence partitions can be found for both valid data and invalid data, i.e. values that should be rejected.

1. The test development process
- *1.1 Background*
- *1.2 Test analysis*
- *1.3 Test design*
- *1.4 Test implementation*

2. Categories of test design techniques

3. Specification-based techniques (black-box)
- *3.1 Equivalence partitioning*
- *3.2 Boundary value analysis*
- *3.3 Decision table testing*
- *3.4 State transition testing*
- *3.5 Use case testing*

4. Structure-based techniques (white-box)
- *4.1 Statement testing and coverage*
- *4.2 Decision testing and coverage*
- *4.3 Other structure-based techniques*

5. Experience-based techniques

6. Choosing test techniques

Equivalence partitioning

Technique

Inputs/outputs/internal values of the software are divided into groups that are expected to exhibit similar behavior.

Equivalence partitions can be found for both valid data and invalid data, i.e. values that should be rejected.

Notes

- Tests can be designed to cover more than one partitions.
- Equivalence partitioning is applicable at all levels of testing.
- Equivalence partitioning as a technique can be used to achieve input and output coverage.

1. The test development process

- *1.1 Background*
- *1.2 Test analysis*
- *1.3 Test design*
- *1.4 Test implementation*

2. Categories of test design techniques

3. Specification-based techniques (black-box)

- *3.1 Equivalence partitioning*
- *3.2 Boundary value analysis*
- *3.3 Decision table testing*
- *3.4 State transition testing*
- *3.5 Use case testing*

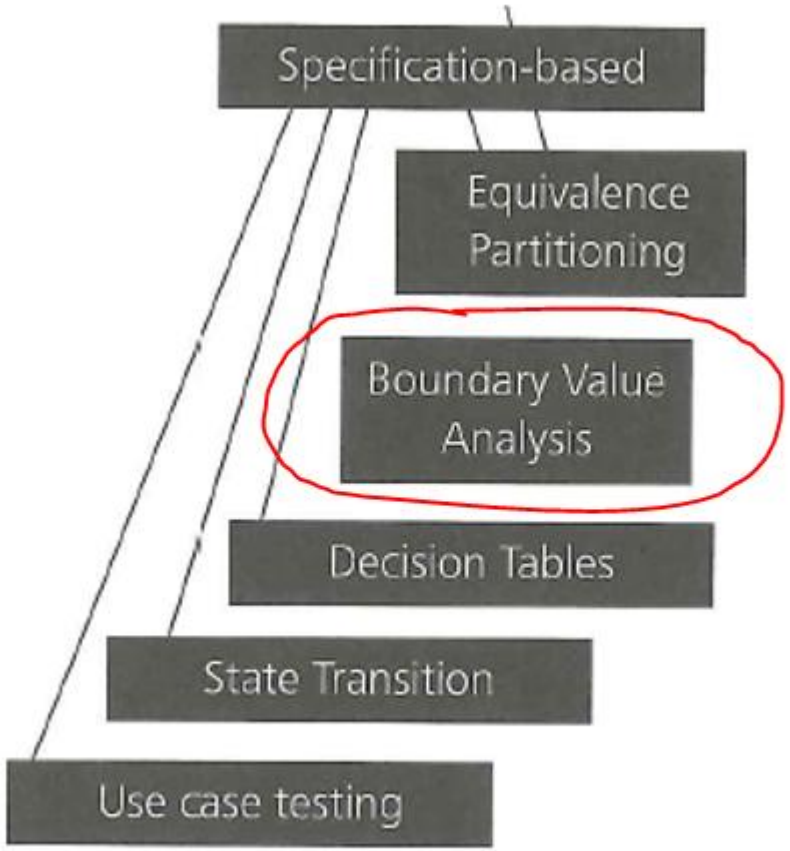
4. Structure-based techniques (white-box)

- *4.1 Statement testing and coverage*
- *4.2 Decision testing and coverage*
- *4.3 Other structure-based techniques*

5. Experience-based techniques

6. Choosing test techniques

Black-box techniques



1. The test development process
- *1.1 Background*
- *1.2 Test analysis*
- *1.3 Test design*
- *1.4 Test implementation*

2. Categories of test design techniques

3. Specification-based techniques (black-box)
- *3.1 Equivalence partitioning*
- *3.2 Boundary value analysis*
- *3.3 Decision table testing*
- *3.4 State transition testing*
- *3.5 Use case testing*

4. Structure-based techniques (white-box)
- *4.1 Statement testing and coverage*
- *4.2 Decision testing and coverage*
- *4.3 Other structure-based techniques*

5. Experience-based techniques

6. Choosing test techniques

Boundary value analysis

Behavior at the edge of each equivalence partition is more likely to be incorrect than behavior within the partition.

Boundary are an area where testing is likely to yield defects.

Example

A printer has an input option of number of copies to be made:

Invalid partition	Valid partition	Invalid partition
0	1 99	100

1. The test development process
- 1.1 Background
- 1.2 Test analysis
- 1.3 Test design
- 1.4 Test implementation

2. Categories of test design techniques

3. Specification-based techniques (black-box)
- 3.1 Equivalence partitioning
- 3.2 Boundary value analysis
- 3.3 Decision table testing
- 3.4 State transition testing
- 3.5 Use case testing

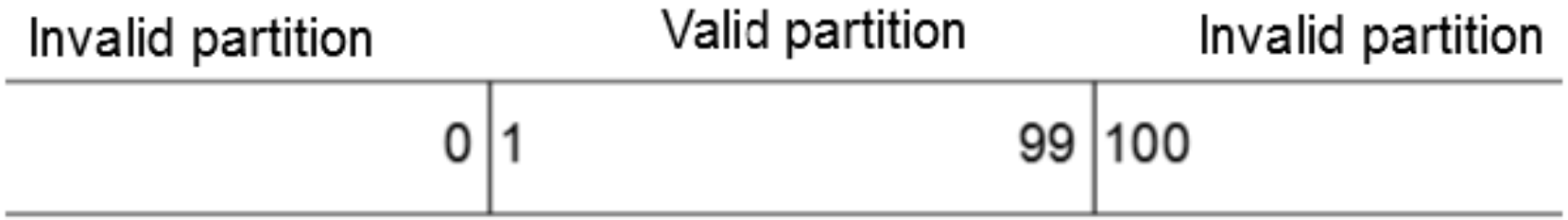
4. Structure-based techniques (white-box)
- 4.1 Statement testing and coverage
- 4.2 Decision testing and coverage
- 4.3 Other structure-based techniques

5. Experience-based techniques

6. Choosing test techniques

Boundary value analysis

The maximum and minimum values of a partition are its boundary values.



Valid boundary values: 1, 99
Invalid boundary values: 0, 100

When designing test cases, a test for each boundary value is chosen.

1. The test development process
- 1.1 Background
- 1.2 Test analysis
- 1.3 Test design
- 1.4 Test implementation

2. Categories of test design techniques

3. Specification-based techniques (black-box)
- 3.1 Equivalence partitioning
- 3.2 Boundary value analysis
- 3.3 Decision table testing
- 3.4 State transition testing
- 3.5 Use case testing

4. Structure-based techniques (white-box)
- 4.1 Statement testing and coverage
- 4.2 Decision testing and coverage
- 4.3 Other structure-based techniques

5. Experience-based techniques

6. Choosing test techniques

Boundary value analysis

Another example with two decimals:

Invalid partition	Valid partition		Invalid partition
- 0.01	0.00	100.00	100.01

Valid boundary values: 0.00, 100.00
Invalid boundary values: - 0.01, 100.01

1. The test development process
- *1.1 Background*
- *1.2 Test analysis*
- *1.3 Test design*
- *1.4 Test implementation*

2. Categories of test design techniques

3. Specification-based techniques (black-box)
- *3.1 Equivalence partitioning*
- *3.2 Boundary value analysis*
- *3.3 Decision table testing*
- *3.4 State transition testing*
- *3.5 Use case testing*

4. Structure-based techniques (white-box)
- *4.1 Statement testing and coverage*
- *4.2 Decision testing and coverage*
- *4.3 Other structure-based techniques*

5. Experience-based techniques

6. Choosing test techniques

Boundary value analysis

Boundary analysis

Analysis at the edge of each equivalence partition.

Why? Because there, the results are more likely to be incorrect.

1. The test development process
- *1.1 Background*
- *1.2 Test analysis*
- *1.3 Test design*
- *1.4 Test implementation*

2. Categories of test design techniques

3. Specification-based techniques (black-box)
- *3.1 Equivalence partitioning*
- *3.2 Boundary value analysis*
- *3.3 Decision table testing*
- *3.4 State transition testing*
- *3.5 Use case testing*

4. Structure-based techniques (white-box)
- *4.1 Statement testing and coverage*
- *4.2 Decision testing and coverage*
- *4.3 Other structure-based techniques*

5. Experience-based techniques

6. Choosing test techniques

Boundary value analysis

Boundary analysis

Analysis at the edge of each equivalence partition.

Why? Because there, the results are more likely to be incorrect.

The maximum and minimum values of a partition are its boundary values.

1. The test development process
- *1.1 Background*
- *1.2 Test analysis*
- *1.3 Test design*
- *1.4 Test implementation*

2. Categories of test design techniques

3. Specification-based techniques (black-box)
- *3.1 Equivalence partitioning*
- *3.2 Boundary value analysis*
- *3.3 Decision table testing*
- *3.4 State transition testing*
- *3.5 Use case testing*

4. Structure-based techniques (white-box)
- *4.1 Statement testing and coverage*
- *4.2 Decision testing and coverage*
- *4.3 Other structure-based techniques*

5. Experience-based techniques

6. Choosing test techniques

Boundary value analysis

Boundary analysis

Analysis at the edge of each equivalence partition.

Why? Because there, the results are more likely to be incorrect.

The maximum and minimum values of a partition are its boundary values.

Valid and invalid boundary

- A boundary value for a valid partition is a valid boundary value.
- A boundary value for an invalid partition is an invalid boundary value.
- Tests can be designed to cover both valid and invalid boundary values.

1. The test development process
- *1.1 Background*
- *1.2 Test analysis*
- *1.3 Test design*
- *1.4 Test implementation*

2. Categories of test design techniques

3. Specification-based techniques (black-box)
- *3.1 Equivalence partitioning*
- *3.2 Boundary value analysis*
- *3.3 Decision table testing*
- *3.4 State transition testing*
- *3.5 Use case testing*

4. Structure-based techniques (white-box)
- *4.1 Statement testing and coverage*
- *4.2 Decision testing and coverage*
- *4.3 Other structure-based techniques*

5. Experience-based techniques

6. Choosing test techniques

Boundary value analysis

Notes

- Boundary values are used for test data selection.
- Boundary value analysis can be applied at all test levels.
- Its relatively easy to apply and its defect finding capability is high.
- Detailed specifications are helpful.
- This technique is often considered as an extension of equivalence partitioning.

1. The test development process
- *1.1 Background*
- *1.2 Test analysis*
- *1.3 Test design*
- *1.4 Test implementation*

2. Categories of test design techniques

3. Specification-based techniques (black-box)
- *3.1 Equivalence partitioning*
- *3.2 Boundary value analysis*
- *3.3 Decision table testing*
- *3.4 State transition testing*
- *3.5 Use case testing*

4. Structure-based techniques (white-box)
- *4.1 Statement testing and coverage*
- *4.2 Decision testing and coverage*
- *4.3 Other structure-based techniques*

5. Experience-based techniques

6. Choosing test techniques

Equivalence partitioning and boundary

Why do both equivalence partitioning and boundary value analysis?

1. The test development process
- *1.1 Background*
- *1.2 Test analysis*
- *1.3 Test design*
- *1.4 Test implementation*

2. Categories of test design techniques

3. Specification-based techniques (black-box)
- *3.1 Equivalence partitioning*
- *3.2 Boundary value analysis*
- *3.3 Decision table testing*
- *3.4 State transition testing*
- *3.5 Use case testing*

4. Structure-based techniques (white-box)
- *4.1 Statement testing and coverage*
- *4.2 Decision testing and coverage*
- *4.3 Other structure-based techniques*

5. Experience-based techniques

6. Choosing test techniques

Equivalence partitioning and boundary

Why do both equivalence partitioning and boundary value analysis?

Boundary values are usually extreme values

1. The test development process
- *1.1 Background*
- *1.2 Test analysis*
- *1.3 Test design*
- *1.4 Test implementation*

2. Categories of test design techniques

3. Specification-based techniques (black-box)
- *3.1 Equivalence partitioning*
- *3.2 Boundary value analysis*
- *3.3 Decision table testing*
- *3.4 State transition testing*
- *3.5 Use case testing*

4. Structure-based techniques (white-box)
- *4.1 Statement testing and coverage*
- *4.2 Decision testing and coverage*
- *4.3 Other structure-based techniques*

5. Experience-based techniques

6. Choosing test techniques

Equivalence partitioning and boundary

Why do both equivalence partitioning and boundary value analysis?

Boundary values are usually extreme values.

To gain confidence to the system we also want to test it under normal circumstances.

1. The test development process
- *1.1 Background*
- *1.2 Test analysis*
- *1.3 Test design*
- *1.4 Test implementation*

2. Categories of test design techniques

3. Specification-based techniques (black-box)
- *3.1 Equivalence partitioning*
- *3.2 Boundary value analysis*
- *3.3 Decision table testing*
- *3.4 State transition testing*
- *3.5 Use case testing*

4. Structure-based techniques (white-box)
- *4.1 Statement testing and coverage*
- *4.2 Decision testing and coverage*
- *4.3 Other structure-based techniques*

5. Experience-based techniques

6. Choosing test techniques

Equivalence partitioning and boundary

Why do both equivalence partitioning and boundary value analysis?

Boundary values are usually extreme values.

To gain confidence to the system we also want to test it under normal circumstances.

Rule of thumb (Closed, valid partitions)
Pick two boundary values (min and max), and one value from the middle of the partition.

1. The test development process
- *1.1 Background*
- *1.2 Test analysis*
- *1.3 Test design*
- *1.4 Test implementation*

2. Categories of test design techniques

3. Specification-based techniques (black-box)
- *3.1 Equivalence partitioning*
- *3.2 Boundary value analysis*
- *3.3 Decision table testing*
- *3.4 State transition testing*
- *3.5 Use case testing*

4. Structure-based techniques (white-box)
- *4.1 Statement testing and coverage*
- *4.2 Decision testing and coverage*
- *4.3 Other structure-based techniques*

5. Experience-based techniques

6. Choosing test techniques

Equivalence partitioning

We can also apply equivalence partitioning and boundary value analysis more than once to the same specification item. For example, if an internal telephone system for a company with 200 telephones has 3-digit extension numbers from 100 to 699, we can identify the following partitions and boundaries:

1. The test development process
- *1.1 Background*
- *1.2 Test analysis*
- *1.3 Test design*
- *1.4 Test implementation*

2. Categories of test design techniques

3. Specification-based techniques (black-box)
- *3.1 Equivalence partitioning*
- *3.2 Boundary value analysis*
- *3.3 Decision table testing*
- *3.4 State transition testing*
- *3.5 Use case testing*

4. Structure-based techniques (white-box)
- *4.1 Statement testing and coverage*
- *4.2 Decision testing and coverage*
- *4.3 Other structure-based techniques*

5. Experience-based techniques

6. Choosing test techniques

Equivalence partitioning

We can also apply equivalence partitioning and boundary value analysis more than once to the same specification item. For example, if an internal telephone system for a company with 200 telephones has 3-digit extension numbers from 100 to 699, we can identify the following partitions and boundaries:

- digits (characters 0 to 9) with the invalid partition containing non-digits

1. The test development process
- *1.1 Background*
- *1.2 Test analysis*
- *1.3 Test design*
- *1.4 Test implementation*

2. Categories of test design techniques

3. Specification-based techniques (black-box)
- *3.1 Equivalence partitioning*
- *3.2 Boundary value analysis*
- *3.3 Decision table testing*
- *3.4 State transition testing*
- *3.5 Use case testing*

4. Structure-based techniques (white-box)
- *4.1 Statement testing and coverage*
- *4.2 Decision testing and coverage*
- *4.3 Other structure-based techniques*

5. Experience-based techniques

6. Choosing test techniques

Equivalence partitioning

We can also apply equivalence partitioning and boundary value analysis more than once to the same specification item. For example, if an internal telephone system for a company with 200 telephones has 3-digit extension numbers from 100 to 699, we can identify the following partitions and boundaries:

- digits (characters 0 to 9) with the invalid partition containing non-digits
- number of digits, 3 (so invalid boundary values of 2 digits and 4 digits)

1. The test development process
- *1.1 Background*
- *1.2 Test analysis*
- *1.3 Test design*
- *1.4 Test implementation*

2. Categories of test design techniques

3. Specification-based techniques (black-box)
- *3.1 Equivalence partitioning*
- *3.2 Boundary value analysis*
- *3.3 Decision table testing*
- *3.4 State transition testing*
- *3.5 Use case testing*

4. Structure-based techniques (white-box)
- *4.1 Statement testing and coverage*
- *4.2 Decision testing and coverage*
- *4.3 Other structure-based techniques*

5. Experience-based techniques

6. Choosing test techniques

Equivalence partitioning

We can also apply equivalence partitioning and boundary value analysis more than once to the same specification item. For example, if an internal telephone system for a company with 200 telephones has 3-digit extension numbers from 100 to 699, we can identify the following partitions and boundaries:

- digits (characters 0 to 9) with the invalid partition containing non-digits
- number of digits, 3 (so invalid boundary values of 2 digits and 4 digits)
- range of extension numbers, 100 to 699 (so invalid boundary values of 099 and 700)

1. The test development process
- *1.1 Background*
- *1.2 Test analysis*
- *1.3 Test design*
- *1.4 Test implementation*

2. Categories of test design techniques

3. Specification-based techniques (black-box)
- *3.1 Equivalence partitioning*
- *3.2 Boundary value analysis*
- *3.3 Decision table testing*
- *3.4 State transition testing*
- *3.5 Use case testing*

4. Structure-based techniques (white-box)
- *4.1 Statement testing and coverage*
- *4.2 Decision testing and coverage*
- *4.3 Other structure-based techniques*

5. Experience-based techniques

6. Choosing test techniques

Equivalence partitioning

We can also apply equivalence partitioning and boundary value analysis more than once to the same specification item. For example, if an internal telephone system for a company with 200 telephones has 3-digit extension numbers from 100 to 699, we can identify the following partitions and boundaries:

- digits (characters 0 to 9) with the invalid partition containing non-digits
- number of digits, 3 (so invalid boundary values of 2 digits and 4 digits)
- range of extension numbers, 100 to 699 (so invalid boundary values of 099 and 700)
- extensions that are in use and those that are not (two valid partitions, no boundaries)

1. The test development process
- 1.1 Background
- 1.2 Test analysis
- 1.3 Test design
- 1.4 Test implementation

2. Categories of test design techniques

3. Specification-based techniques (black-box)
- 3.1 Equivalence partitioning
- 3.2 Boundary value analysis
- 3.3 Decision table testing
- 3.4 State transition testing
- 3.5 Use case testing

4. Structure-based techniques (white-box)
- 4.1 Statement testing and coverage
- 4.2 Decision testing and coverage
- 4.3 Other structure-based techniques

5. Experience-based techniques

6. Choosing test techniques

Equivalence partitioning

We can also apply equivalence partitioning and boundary value analysis more than once to the same specification item. For example, if an internal telephone system for a company with 200 telephones has 3-digit extension numbers from 100 to 699, we can identify the following partitions and boundaries:

- digits (characters 0 to 9) with the invalid partition containing non-digits
- number of digits, 3 (so invalid boundary values of 2 digits and 4 digits)
- range of extension numbers, 100 to 699 (so invalid boundary values of 099 and 700)
- extensions that are in use and those that are not (two valid partitions, no boundaries)
- the lowest and highest extension numbers that are in use could also be used as boundary values

1. The test development process
- *1.1 Background*
- *1.2 Test analysis*
- *1.3 Test design*
- *1.4 Test implementation*

2. Categories of test design techniques

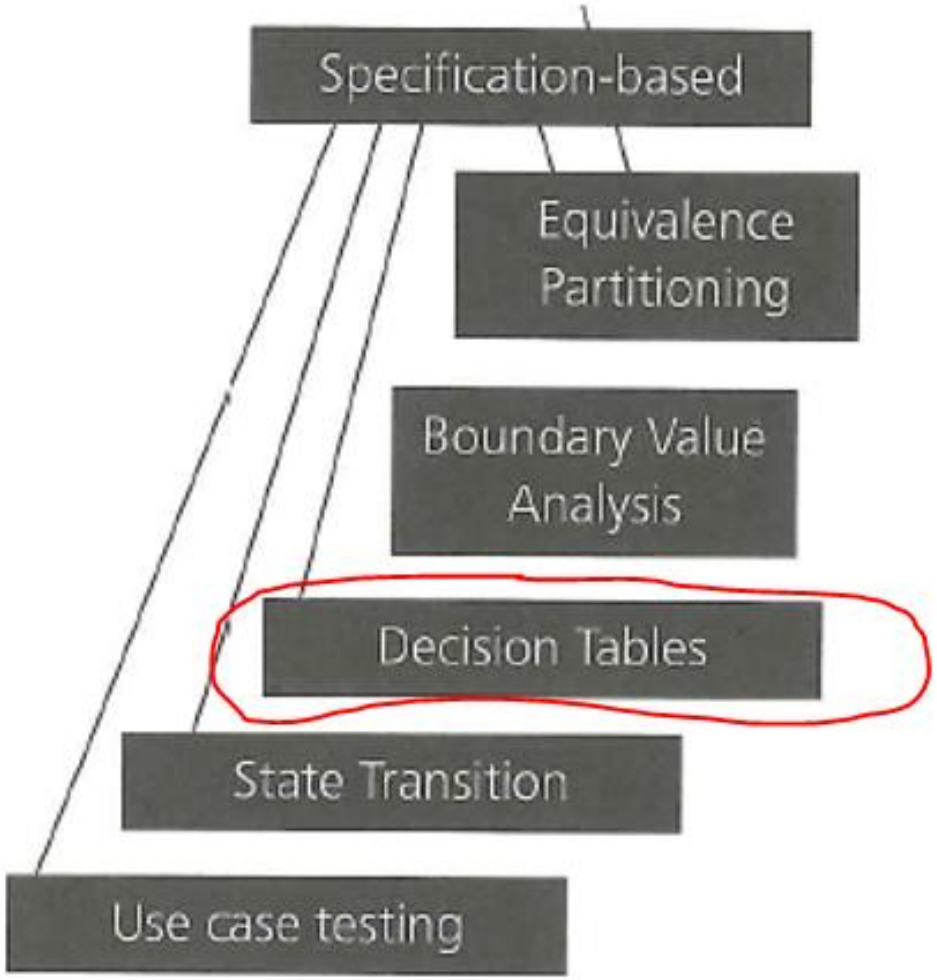
3. Specification-based techniques (black-box)
- *3.1 Equivalence partitioning*
- *3.2 Boundary value analysis*
- *3.3 Decision table testing*
- *3.4 State transition testing*
- *3.5 Use case testing*

4. Structure-based techniques (white-box)
- *4.1 Statement testing and coverage*
- *4.2 Decision testing and coverage*
- *4.3 Other structure-based techniques*

5. Experience-based techniques

6. Choosing test techniques

Black-box techniques



1. The test development process
- *1.1 Background*
- *1.2 Test analysis*
- *1.3 Test design*
- *1.4 Test implementation*

2. Categories of test design techniques

3. Specification-based techniques (black-box)
- *3.1 Equivalence partitioning*
- *3.2 Boundary value analysis*
- *3.3 Decision table testing*
- *3.4 State transition testing*
- *3.5 Use case testing*

4. Structure-based techniques (white-box)
- *4.1 Statement testing and coverage*
- *4.2 Decision testing and coverage*
- *4.3 Other structure-based techniques*

5. Experience-based techniques

6. Choosing test techniques

Decision table testing

Decision tables are a good way

- to capture system requirements that contain *logical conditions*
- to document internal system design
- to record complex business rules that a system is to implement

1. The test development process
- *1.1 Background*
- *1.2 Test analysis*
- *1.3 Test design*
- *1.4 Test implementation*

2. Categories of test design techniques

3. Specification-based techniques (black-box)
- *3.1 Equivalence partitioning*
- *3.2 Boundary value analysis*
- *3.3 Decision table testing*
- *3.4 State transition testing*
- *3.5 Use case testing*

4. Structure-based techniques (white-box)
- *4.1 Statement testing and coverage*
- *4.2 Decision testing and coverage*
- *4.3 Other structure-based techniques*

5. Experience-based techniques

6. Choosing test techniques

Decision table testing

Recall truth tables in mathematical logic :

	P	Q	P ∧ Q
Case 1	T	T	T
Case 2	T	F	F
Case 3	F	T	F
Case 4	F	F	F

1. The test development process
- 1.1 Background
- 1.2 Test analysis
- 1.3 Test design
- 1.4 Test implementation

2. Categories of test design techniques

3. Specification-based techniques (black-box)
- 3.1 Equivalence partitioning
- 3.2 Boundary value analysis
- 3.3 Decision table testing
- 3.4 State transition testing
- 3.5 Use case testing

4. Structure-based techniques (white-box)
- 4.1 Statement testing and coverage
- 4.2 Decision testing and coverage
- 4.3 Other structure-based techniques

5. Experience-based techniques

6. Choosing test techniques

Decision table testing

Decision tables

Cause-effect table

Used when inputs and actions can be expressed as Boolean values

Systematic way of stating complex business rules

Help testers identify effects of combinations of different input

Effective approach to reveal faults in the requirements

Conditions	R1	R2	R3	R4
Condition_1	T	T	F	F
Condition_2	T	F	T	F
Actions				
Action_1	?	?	?	?
Action_2	?	?	?	?

1. The test development process
- *1.1 Background*
- *1.2 Test analysis*
- *1.3 Test design*
- *1.4 Test implementation*

2. Categories of test design techniques

3. Specification-based techniques (black-box)
- *3.1 Equivalence partitioning*
- *3.2 Boundary value analysis*
- *3.3 Decision table testing*
- *3.4 State transition testing*
- *3.5 Use case testing*

4. Structure-based techniques (white-box)
- *4.1 Statement testing and coverage*
- *4.2 Decision testing and coverage*
- *4.3 Other structure-based techniques*

5. Experience-based techniques

6. Choosing test techniques

Decision table testing

- When creating decision tables, the specification is analyzed, and actions of the system are identified.
- If the input conditions and actions are stated in a way where they are either be true or false (Boolean), decision tables can be useful.
- The decision table contains the triggering conditions, i.e. all combinations of **true** and **false** for all input conditions, and the resulting actions for each combination of conditions.

1. The test development process
- 1.1 Background
- 1.2 Test analysis
- 1.3 Test design
- 1.4 Test implementation

2. Categories of test design techniques

3. Specification-based techniques (black-box)
- 3.1 Equivalence partitioning
- 3.2 Boundary value analysis
- 3.3 Decision table testing
- 3.4 State transition testing
- 3.5 Use case testing

4. Structure-based techniques (white-box)
- 4.1 Statement testing and coverage
- 4.2 Decision testing and coverage
- 4.3 Other structure-based techniques

5. Experience-based techniques

6. Choosing test techniques

Decision table testing

	Rule 1	Rule 2	Rule 3	Rule 4	Rule 5	Rule 6	Rule 7	Rule 8
Conditions								
Condition 1	True	True	True	True	False	False	False	False
Condition 2	True	True	False	False	True	True	False	False
Condition 3	True	False	True	False	True	False	True	False
Actions								
Action 1	**False**	**False**	True	**False**	**False**	**False**	**False**	**False**
Action 2	**False**	**False**	**False**	True	**False**	**False**	**False**	**False**
Action 3	True	**False**	**False**	**False**	**False**	**False**	**False**	**False**
Action 4	**False**	**False**	**False**	**False**	True	**False**	**False**	**False**

1. The test development process
- 1.1 Background
- 1.2 Test analysis
- 1.3 Test design
- 1.4 Test implementation

2. Categories of test design techniques

3. Specification-based techniques (black-box)
- 3.1 Equivalence partitioning
- 3.2 Boundary value analysis
- 3.3 Decision table testing
- 3.4 State transition testing
- 3.5 Use case testing

4. Structure-based techniques (white-box)
- 4.1 Statement testing and coverage
- 4.2 Decision testing and coverage
- 4.3 Other structure-based techniques

5. Experience-based techniques

6. Choosing test techniques

Decision table testing

	Rule 1	Rule 2	Rule 3	Rule 4	Rule 5	Rule 6	Rule 7	Rule 8
Conditions								
Condition 1	*True*	*True*	*True*	*True*	*False*	*False*	*False*	*False*
Condition 2	*True*	*True*	*False*	*False*	*True*	*True*	*False*	*False*
Condition 3	*True*	*False*	*True*	*False*	*True*	*False*	*True*	*False*
Actions	***Action 3***	**x**	***Action 1***	***Action 2***	***Action 4***	**x**	**x**	**x**

The **coverage standard** commonly used with decision table testing is to have at least one test per column, which typically involves covering all combinations of triggering conditions.

1. The test development process
- 1.1 Background
- 1.2 Test analysis
- 1.3 Test design
- 1.4 Test implementation

2. Categories of test design techniques

3. Specification-based techniques (black-box)
- 3.1 Equivalence partitioning
- 3.2 Boundary value analysis
- 3.3 Decision table testing
- 3.4 State transition testing
- 3.5 Use case testing

4. Structure-based techniques (white-box)
- 4.1 Statement testing and coverage
- 4.2 Decision testing and coverage
- 4.3 Other structure-based techniques

5. Experience-based techniques

6. Choosing test techniques

Decision table testing

	Rule 1	Rule 2 & 6	Rule 3	Rule 4	Rule 5	Rule 7 & 8
Conditions						
Condition 1	*True*	----	*True*	*True*	*False*	*False*
Condition 2	*True*	*True*	*False*	*False*	*True*	*False*
Condition 3	*True*	*False*	*True*	*False*	*True*	---
Actions	***Action 3***	**x**	***Action 1***	***Action 2***	***Action 4***	**x**

Columns with identical outcomes may be merged, enabling one test to cover multiple condition combinations and reducing the overall number of tests.

1. The test development process
- *1.1 Background*
- *1.2 Test analysis*
- *1.3 Test design*
- *1.4 Test implementation*

2. Categories of test design techniques

3. Specification-based techniques (black-box)
- *3.1 Equivalence partitioning*
- *3.2 Boundary value analysis*
- *3.3 Decision table testing*
- *3.4 State transition testing*
- *3.5 Use case testing*

4. Structure-based techniques (white-box)
- *4.1 Statement testing and coverage*
- *4.2 Decision testing and coverage*
- *4.3 Other structure-based techniques*

5. Experience-based techniques

6. Choosing test techniques

Decision table testing

Example - Admission to the Computer Science programme

- Betingelser:
  GSK: Generell studiekompetanse
  R1: Bestått matematikk R1 fra videregående skole
  KP: Konkurransepoeng over årets grense

- Aksjoner:
  SP: Tilbud om studieplass
  V: Settes på venteliste
  FK: Tilbud om forberedende kurs i R1.
  A: Avslag, dvs. avslag på studieplass og forkurs, samt heller ikke på venteliste.

1. The test development process
- *1.1 Background*
- *1.2 Test analysis*
- *1.3 Test design*
- *1.4 Test implementation*

2. Categories of test design techniques

3. Specification-based techniques (black-box)
- *3.1 Equivalence partitioning*
- *3.2 Boundary value analysis*
- *3.3 Decision table testing*
- *3.4 State transition testing*
- *3.5 Use case testing*

4. Structure-based techniques (white-box)
- *4.1 Statement testing and coverage*
- *4.2 Decision testing and coverage*
- *4.3 Other structure-based techniques*

5. Experience-based techniques

6. Choosing test techniques

Decision table testing

Example - Admission to the Computer Science programme

- Betingelser:
  GSK: Generell studiekompetanse
  R1: Bestått matematikk R1 fra videregående skole
  KP: Konkurransepoeng over årets grense

- Aksjoner:
  SP: Tilbud om studieplass
  V: Settes på venteliste
  FK: Tilbud om forberedende kurs i R1.
  A: Avslag, dvs. avslag på studieplass og forkurs, samt heller ikke på venteliste.

	Regel 1	Regel 2	Regel 3	Regel 4	Regel 5	Regel 6	Regel 7	Regel 8
GSK	true	true	true	true	false	false	false	false
R1	true	true	false	false	true	true	false	false
KP	true	false	true	false	true	false	true	false
Aksjon	**SP**	**V**	**FK**	**A**	**A**	**A**	**A**	**A**

1. The test development process
- *1.1 Background*
- *1.2 Test analysis*
- *1.3 Test design*
- *1.4 Test implementation*

2. Categories of test design techniques

3. Specification-based techniques (black-box)
- *3.1 Equivalence partitioning*
- *3.2 Boundary value analysis*
- *3.3 Decision table testing*
- *3.4 State transition testing*
- *3.5 Use case testing*

4. Structure-based techniques (white-box)
- *4.1 Statement testing and coverage*
- *4.2 Decision testing and coverage*
- *4.3 Other structure-based techniques*

5. Experience-based techniques

6. Choosing test techniques

Decision table testing

Svar: Vi ser av tabellen at når søkeren ikke har generell studiekompetanse, så får vedkommende avslag, uavhengig av om vedkommende har R1 eller ikke, eller om vedkommende har konkurransepoeng over eller under årets inntaksgrense. Følgelig kan regel 5, 6, 7 og 8 slås sammen, og vi får følgende tabell:

	Regel 1	Regel 2	Regel 3	Regel 4	Regel 5
GSK	true	true	true	true	false
R1	true	true	false	false	----
KP	true	false	true	false	----
Aksjon	**SP**	**V**	**FK**	**A**	**A**

Vi har nå redusert antall regler fra 8 til uten å miste noen testtilfeller. Regel 5 er uavhengig av verdiene til betingelsene R1 og K.

1. The test development process
- *1.1 Background*
- *1.2 Test analysis*
- *1.3 Test design*
- *1.4 Test implementation*

2. Categories of test design techniques

3. Specification-based techniques (black-box)
- *3.1 Equivalence partitioning*
- *3.2 Boundary value analysis*
- *3.3 Decision table testing*
- *3.4 State transition testing*
- *3.5 Use case testing*

4. Structure-based techniques (white-box)
- *4.1 Statement testing and coverage*
- *4.2 Decision testing and coverage*
- *4.3 Other structure-based techniques*

5. Experience-based techniques

6. Choosing test techniques

Decision table testing

Example - Decision table for credit-card

- If you are a new customer opening a credit card account, you will get a 15% discount on all your purchases today.

- If are an existing customer and you hold a loyalty card, you get 10% discount.

- If you have a coupon, you can get 20% off today (but it can't be used with a 'new-customer' discount)

- Discount are added, if applicable.

1. The test development process
- 1.1 Background
- 1.2 Test analysis
- 1.3 Test design
- 1.4 Test implementation

2. Categories of test design techniques

3. Specification-based techniques (black-box)
- 3.1 Equivalence partitioning
- 3.2 Boundary value analysis
- 3.3 Decision table testing
- 3.4 State transition testing
- 3.5 Use case testing

4. Structure-based techniques (white-box)
- 4.1 Statement testing and coverage
- 4.2 Decision testing and coverage
- 4.3 Other structure-based techniques

5. Experience-based techniques

6. Choosing test techniques

Decision table testing

	Rule 1	Rule 2	Rule 3	Rule 4	Rule 5	Rule 6	Rule 7	Rule 8
Conditions								
New customer (15%)	*True*	*True*	*True*	*True*	*False*	*False*	*False*	*False*
Loyalty card (10%)	*True*	*True*	*False*	*False*	*True*	*True*	*False*	*False*
Coupon (20%)	*True*	*False*	*True*	*False*	*True*	*False*	*True*	*False*
Actions								
Discount	x	x	**20 %**	**15%**	**30%**	**10%**	**20%**	**0%**

- If you are a new customer opening a credit card account, you will get a 15% discount on all your purchases today.
- If are an existing customer and you hold a loyalty card, you get 10% discount.
- If you have a coupon, you can get 20% off today (but it can't be used with a 'new-customer' discount)
- Discount are added, if applicable.

1. The test development process
- 1.1 Background
- 1.2 Test analysis
- 1.3 Test design
- 1.4 Test implementation

2. Categories of test design techniques

3. Specification-based techniques (black-box)
- 3.1 Equivalence partitioning
- 3.2 Boundary value analysis
- 3.3 Decision table testing
- 3.4 State transition testing
- 3.5 Use case testing

4. Structure-based techniques (white-box)
- 4.1 Statement testing and coverage
- 4.2 Decision testing and coverage
- 4.3 Other structure-based techniques

5. Experience-based techniques

6. Choosing test techniques

Decision table testing

	Rule 1 & 2	Rule 3 & 7	Rule 4	Rule 5	Rule 6	Rule 8
Conditions						
New customer (15%)	*True*	---	*True*	*False*	*False*	*False*
Loyalty card (10%)	*True*	*False*	*False*	*True*	*True*	*False*
Coupon (20%)	---	*True*	*False*	*True*	*False*	*False*
Actions						
Discount	x	**20 %**	**15%**	**30%**	**10%**	**0%**

- If you are a new customer opening a credit card account, you will get a 15% discount on all your purchases today.
- If are an existing customer and you hold a loyalty card, you get 10% discount.
- If you have a coupon, you can get 20% off today (but it can't be used with a 'new-customer' discount)
- Discount are added, if applicable.

1. The test development process
- *1.1 Background*
- *1.2 Test analysis*
- *1.3 Test design*
- *1.4 Test implementation*

2. Categories of test design techniques

3. Specification-based techniques (black-box)
- *3.1 Equivalence partitioning*
- *3.2 Boundary value analysis*
- *3.3 Decision table testing*
- *3.4 State transition testing*
- *3.5 Use case testing*

4. Structure-based techniques (white-box)
- *4.1 Statement testing and coverage*
- *4.2 Decision testing and coverage*
- *4.3 Other structure-based techniques*

5. Experience-based techniques

6. Choosing test techniques

Decision table testing

- The coverage standard commonly used with decision table testing is to have at least one test per column, which typically involves covering all combinations of triggering conditions.

1. The test development process
- *1.1 Background*
- *1.2 Test analysis*
- *1.3 Test design*
- *1.4 Test implementation*

2. Categories of test design techniques

3. Specification-based techniques (black-box)
- *3.1 Equivalence partitioning*
- *3.2 Boundary value analysis*
- *3.3 Decision table testing*
- *3.4 State transition testing*
- *3.5 Use case testing*

4. Structure-based techniques (white-box)
- *4.1 Statement testing and coverage*
- *4.2 Decision testing and coverage*
- *4.3 Other structure-based techniques*

5. Experience-based techniques

6. Choosing test techniques

Decision table testing

Decision tables are a good way to:

- capture system requirements that contain *logical conditions*
- to document internal system design.

The input conditions and actions are most often stated in such a way that they can be either true or false (Boolean).

The strength of decision table testing is that it creates combinations of conditions that might not otherwise have been exercised during testing.

It may be applied to all situations when the action of the software depends on several logical decisions.