IN3240/IN4240 - Software Testing

Test Design Techniques

Chapter 4 – Part 2

3. Specification-based testing (Black-box)

4. Structure-based testing (White-box)

5. Experience-based testing

6. Choosing test 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

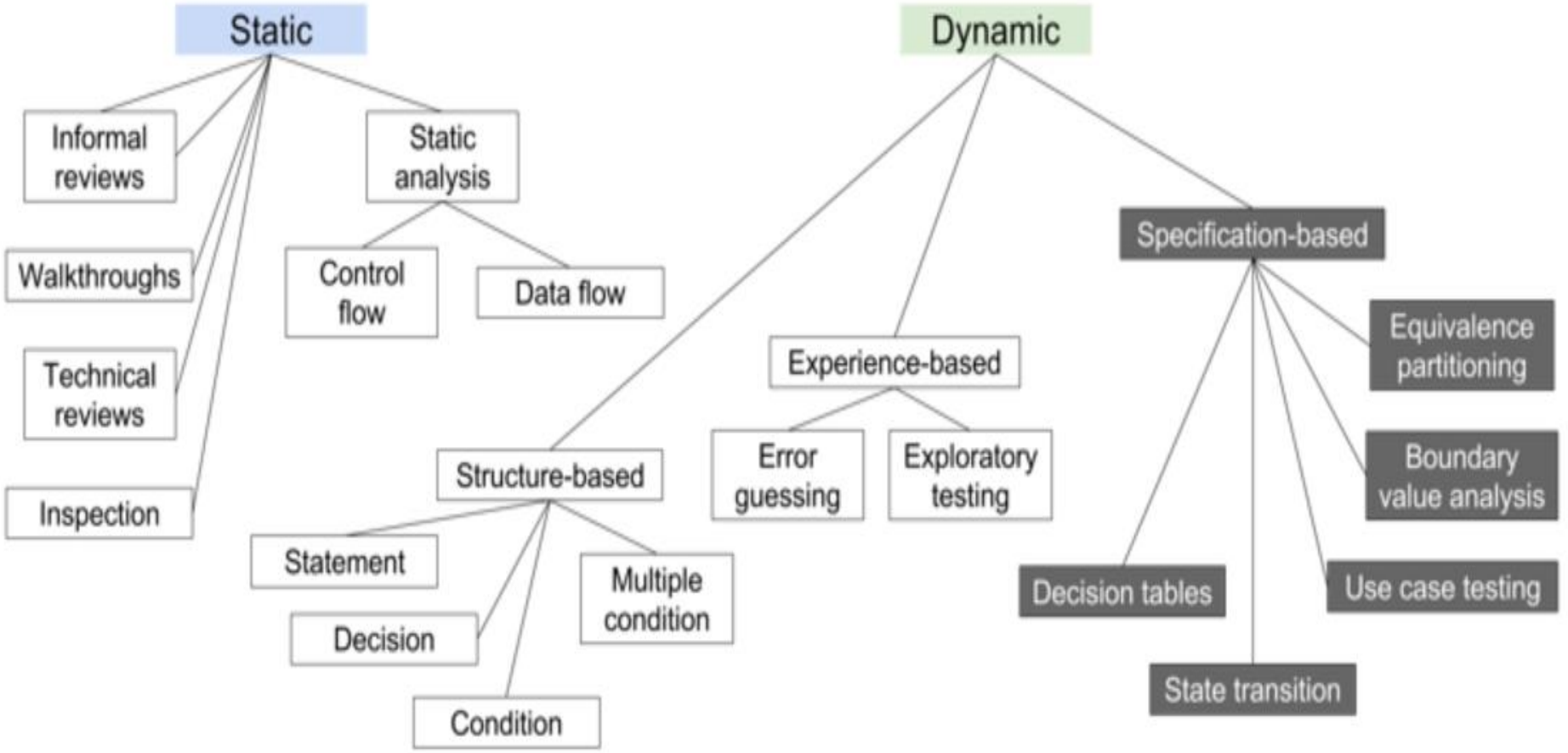
3. Specification-based techniques (black-box)
- *3.1 Equivalence partitioning*
- *3.2 Boundary value analysis*
- *3.3 Decision table testing*
- *3.4 State transition testing*
- *3.5 Use case testing*

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

Test 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

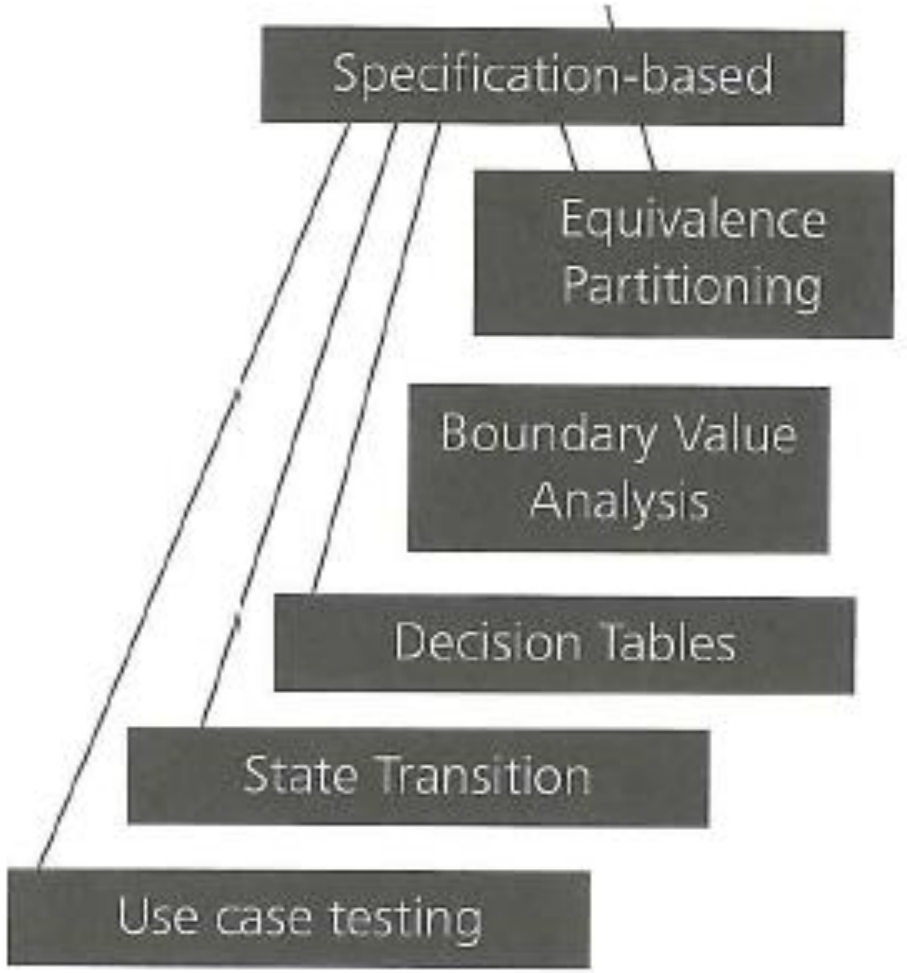
3. Specification-based techniques (black-box)
- *3.1 Equivalence partitioning*
- *3.2 Boundary value analysis*
- *3.3 Decision table testing*
- *3.4 State transition testing*
- *3.5 Use case testing*

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

Specification-based testing 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

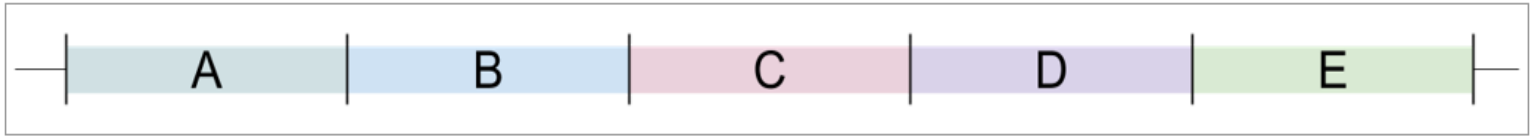
6. Choosing test techniques

EP and BVA

Equivalence partitioning

Identify variables who will be treated the same

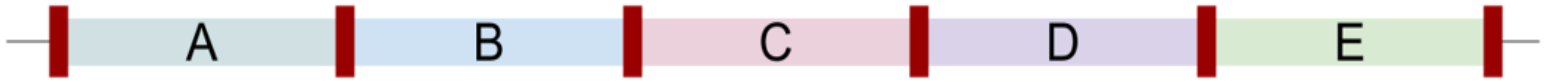
Group values into equivalence classes



Boundary value analysis (BVA)

Testing at the edges of each equivalence class

Things are more likely to go wrong here!



The techniques are often used in combination

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

	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**

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

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 you 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

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.

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

State transition testing

A system can be in a finite number of different states. This aspects of the system can be described as a *'finite state machine' ; a state diagram*.

Any system where you get a different output for the same input, depending on what has happened before, is a finite state system.

The transition from one state to another are determined by the rules of the 'machine'.

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

State transition testing

State transition testing

System can be in a finite number of different states

- Elements of state transition models

States → The SW may occupy

E.g. open / closed, active / inactive

Transitions → From one state to another

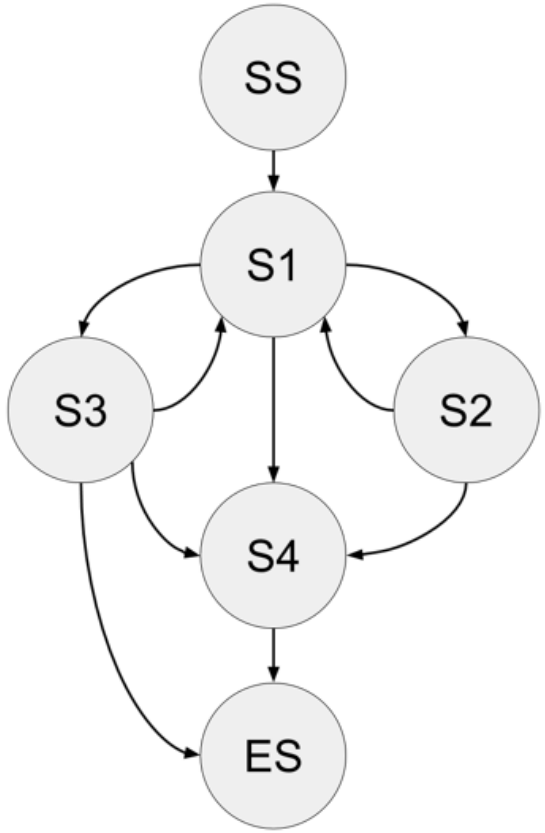
Not all transitions are allowed

Events → Causing state transitions

E.g. closing a file, withdrawing money

Actions → Actions resulting from transitions

E.g. error message



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

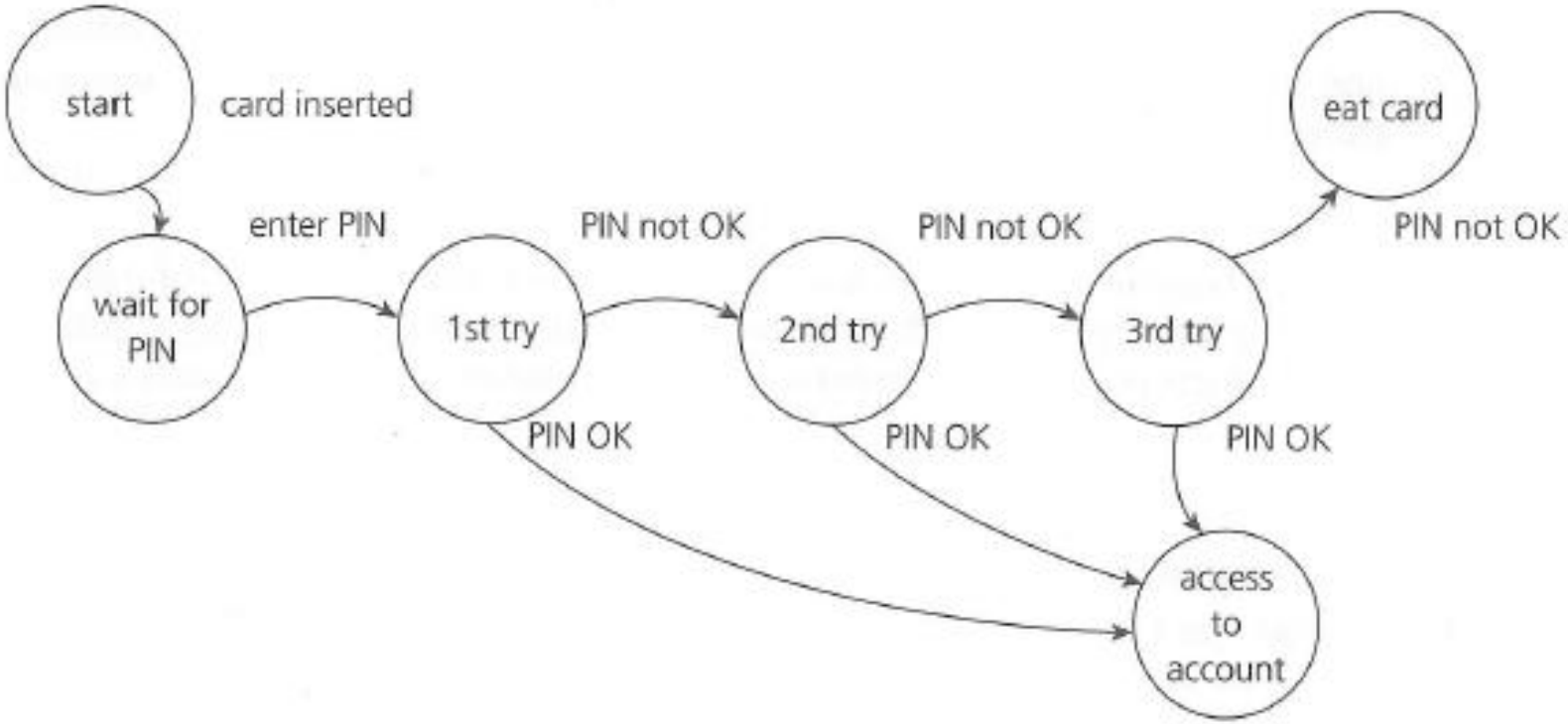
4. Structure-based techniques (white-box)
- 4.1 Statement testing and coverage
- 4.2 Decision testing and coverage
- 4.3 Other structure-based techniques

5. Experience-based techniques

6. Choosing test techniques

State transition testing

A *'finite state machine'* is often shown as a state diagram:
AMT PIN example.



The states of the system under test are separate, identifiable and finite in number.

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*

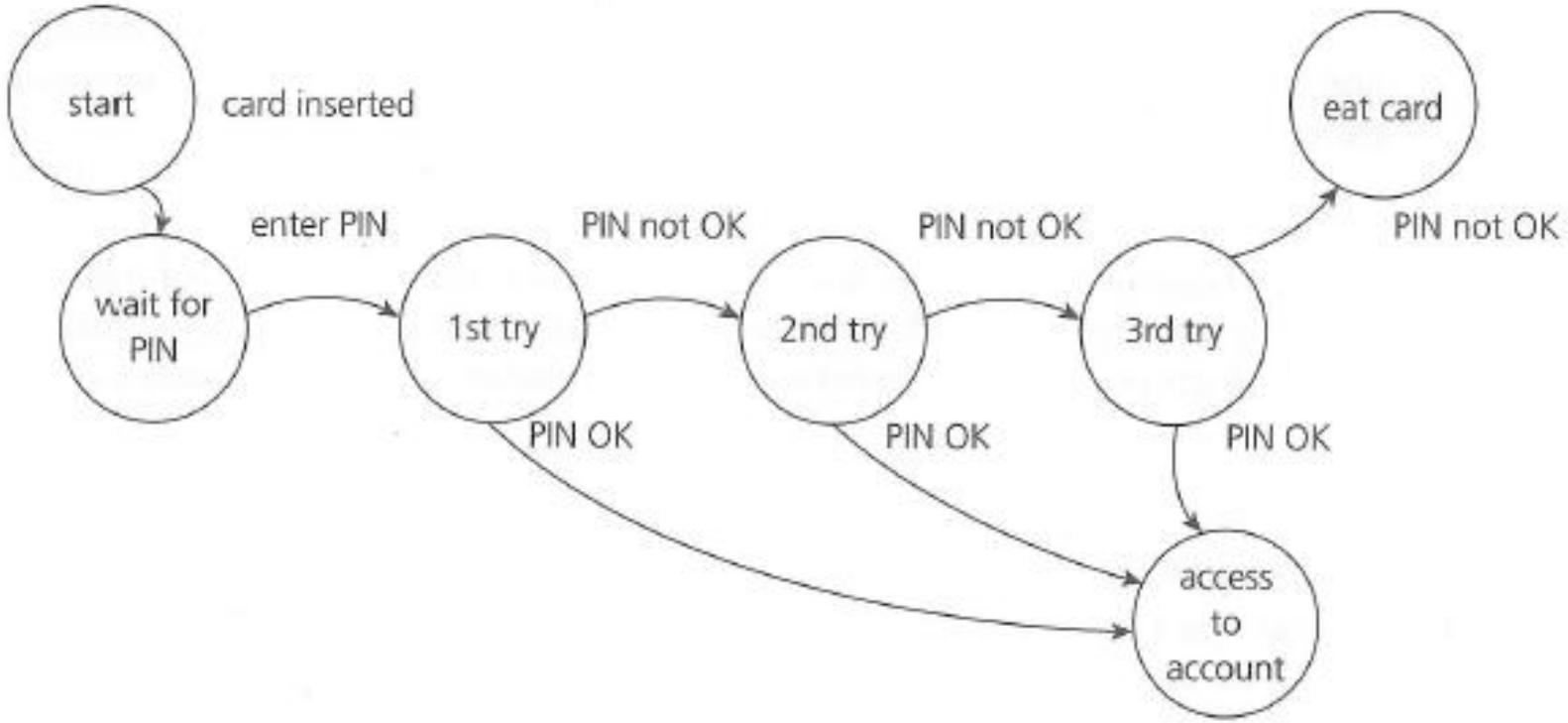
4. Structure-based techniques (white-box)
- *4.1 Statement testing and coverage*
- *4.2 Decision testing and coverage*
- *4.3 Other structure-based techniques*

5. Experience-based techniques

6. Choosing test techniques

State transition testing

How many tests do we need to exercise every state?



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*

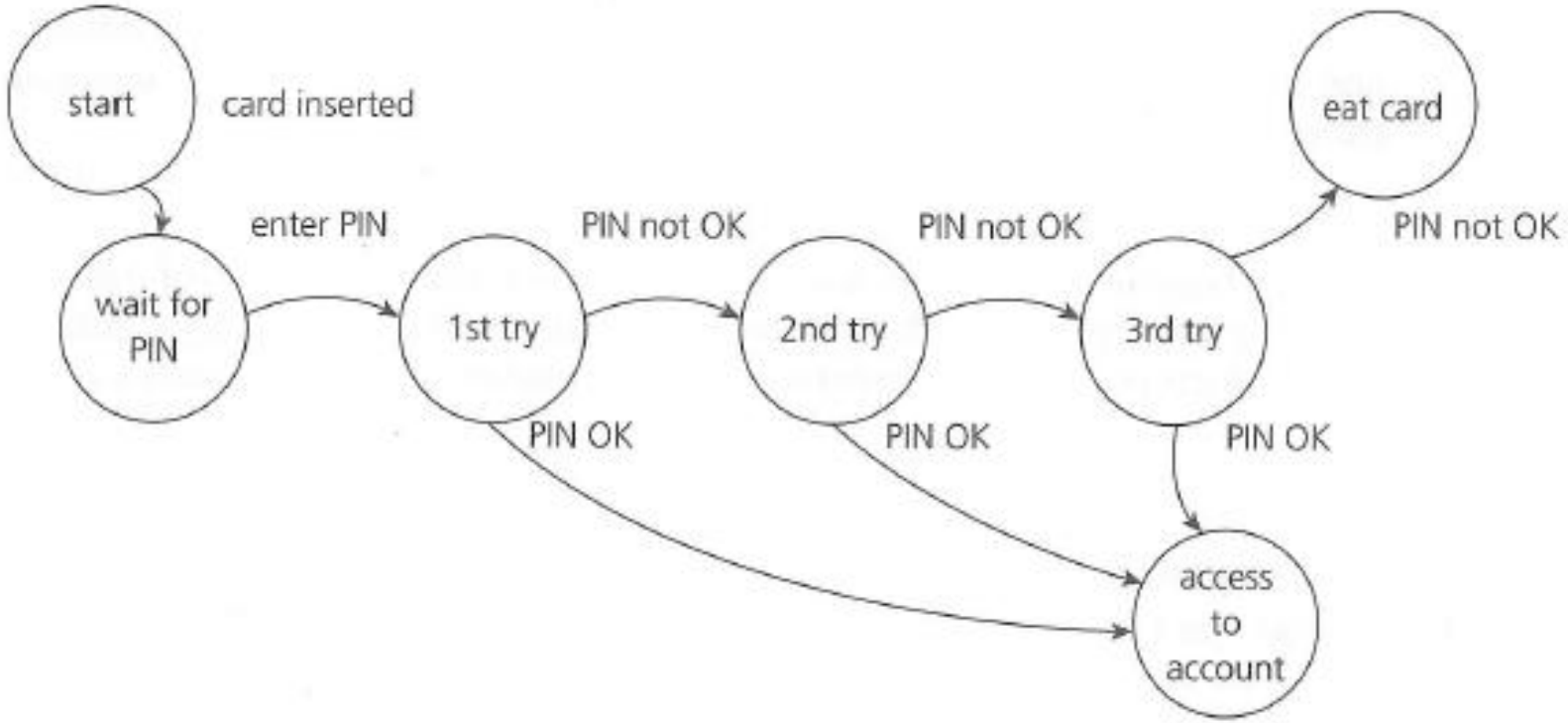
4. Structure-based techniques (white-box)
- *4.1 Statement testing and coverage*
- *4.2 Decision testing and coverage*
- *4.3 Other structure-based techniques*

5. Experience-based techniques

6. Choosing test techniques

State transition testing

How many tests do we need to exercise every state? 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*

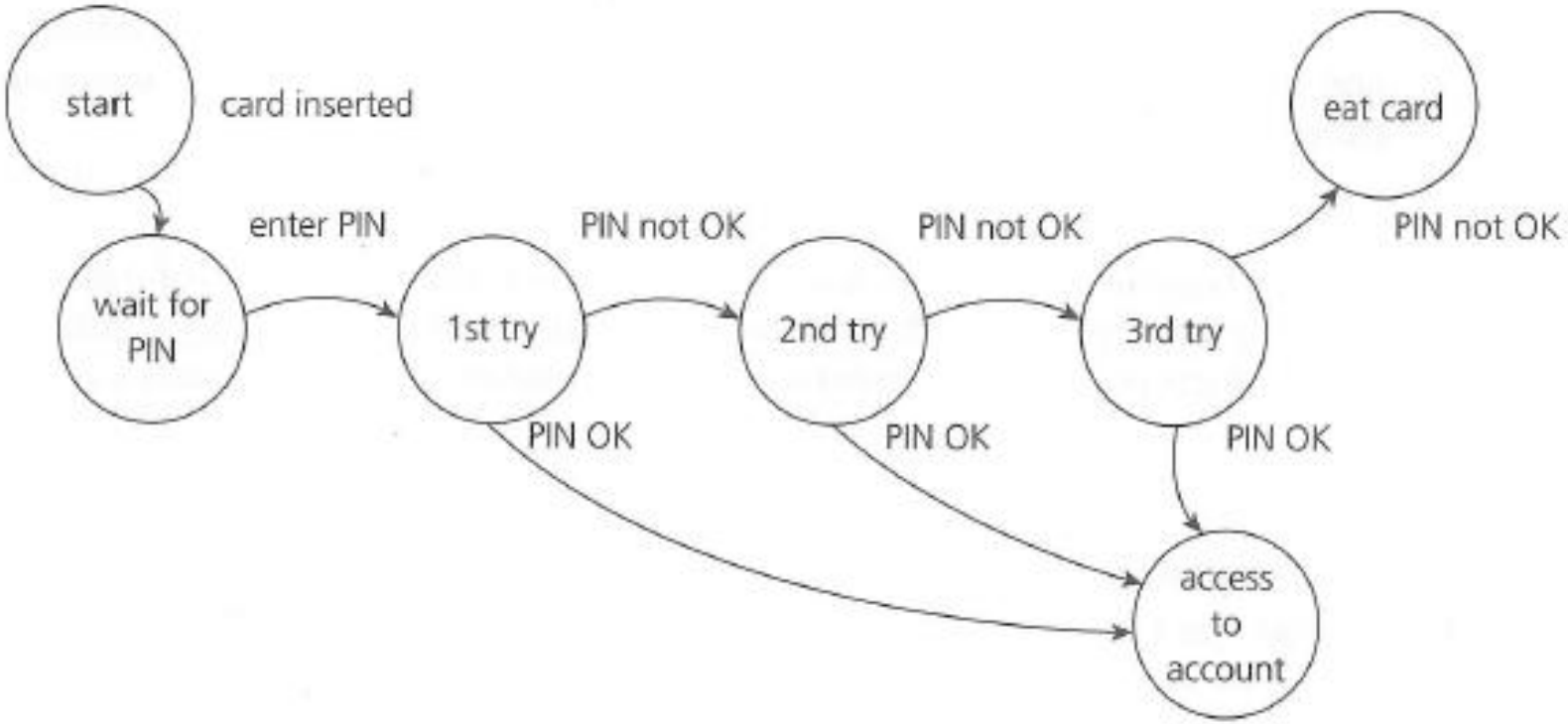
4. Structure-based techniques (white-box)
- *4.1 Statement testing and coverage*
- *4.2 Decision testing and coverage*
- *4.3 Other structure-based techniques*

5. Experience-based techniques

6. Choosing test techniques

State transition testing

How many tests do we need to exercise every transition?



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*

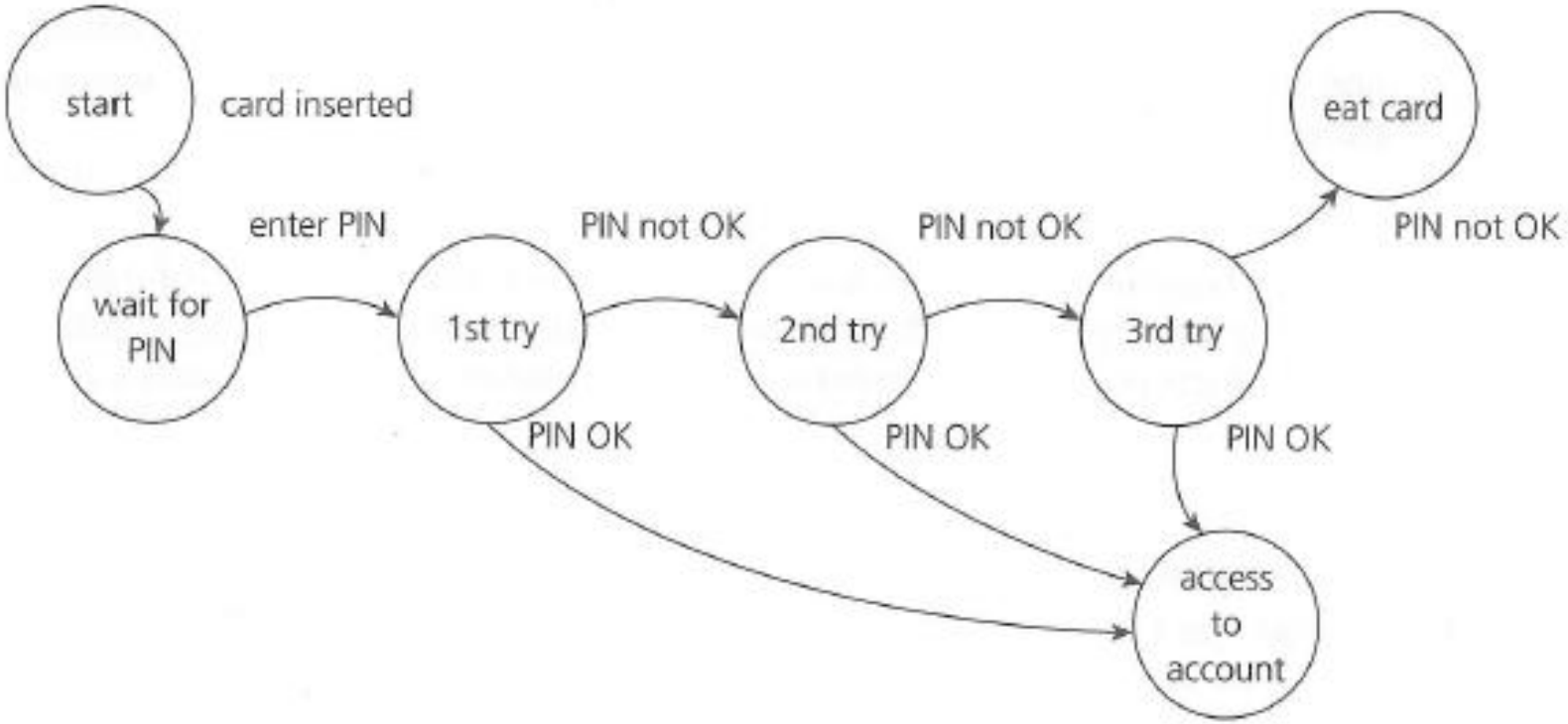
4. Structure-based techniques (white-box)
- *4.1 Statement testing and coverage*
- *4.2 Decision testing and coverage*
- *4.3 Other structure-based techniques*

5. Experience-based techniques

6. Choosing test techniques

State transition testing

How many tests do we need to exercise every transition? 4



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

State transition testing

Why state transition testing?

Because a system may exhibit a different response depending on current conditions or previous history.

State transition testing allows the tester to view:

- the software in terms of its states
- transitions between states
- the inputs or events that trigger state changes (transitions)
- the actions which may result from those transitions

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

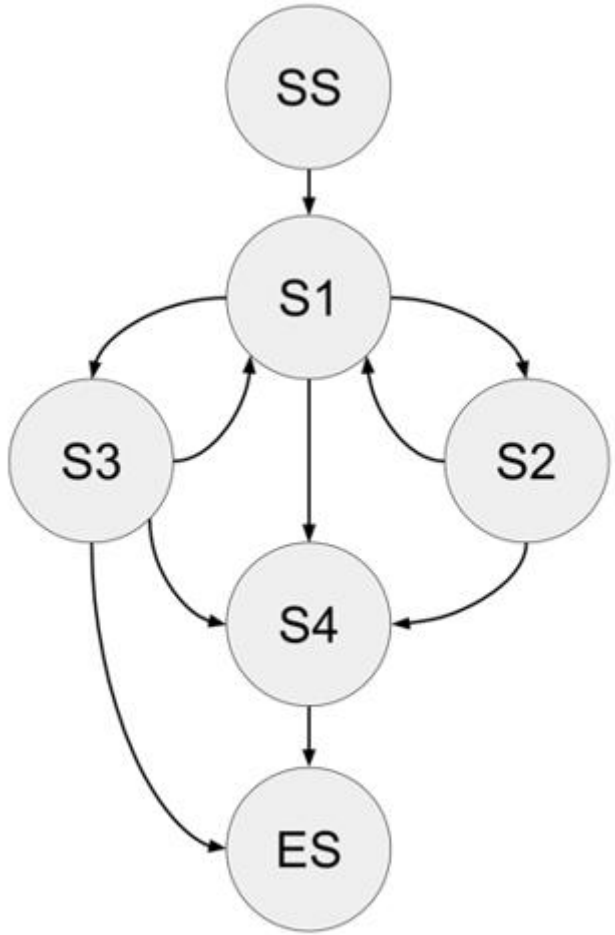
6. Choosing test techniques

State transition testing

Tests can be designed

- to cover a typical sequence of states
- to exercise specific sequences of transitions
- to cover every state
- to exercise every transition
- to test invalid transitions

State transition testing is much used within the software industry and technical automation in general.



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

Use case testing

Use case - describes interactions between actors (users and the system), which produce a result of value to a system user.

Example

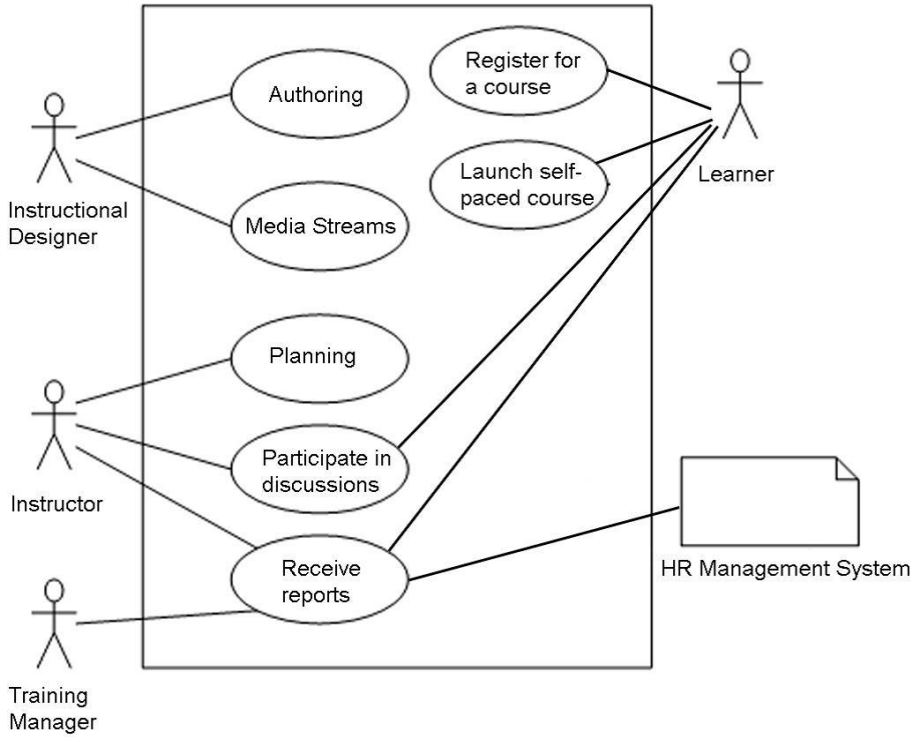
An on-line training website:

User 1: the learner

User 2: the tutor (instructor)

User 3: the training manager

User 4: the instructional designer



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

Use case testing

Use case testing

- Identify test cases that exercise the whole system
  - Transaction by transaction basis
  - From start to finish
- Describes interactions between actor and system
  - Achieve a specific task
  - Produce something of value to the user
- Defined in terms of the actor, not the system
  - Describes process flows through a system
    - Based on its actual use
  - Can uncover integration defects

Use case name	<name>	
Actor(s)	<actor1>, ...	
Pre-conditions	<cond1>, ...	
Post-conditions	<cond1>, ...	
Main Success Scenario	Step	Description
	1	A: <action>
	2	S: <response>
	3	A: <action>
	4	S: <response>
	...	...
Extensions	Step	Description
	S.X	<cause> S: <response>
	S.Y	<cause> S: <response>

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

Use case testing

Each use case has pre-conditions, which need to be met for a use case to work successfully.

Each use case terminates with post-conditions, which are the observable results and final state of the system after the use case has been completed.

A use case usually has a *mainstream* (i.e. most likely) scenario, and sometimes *alternative branches*.

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

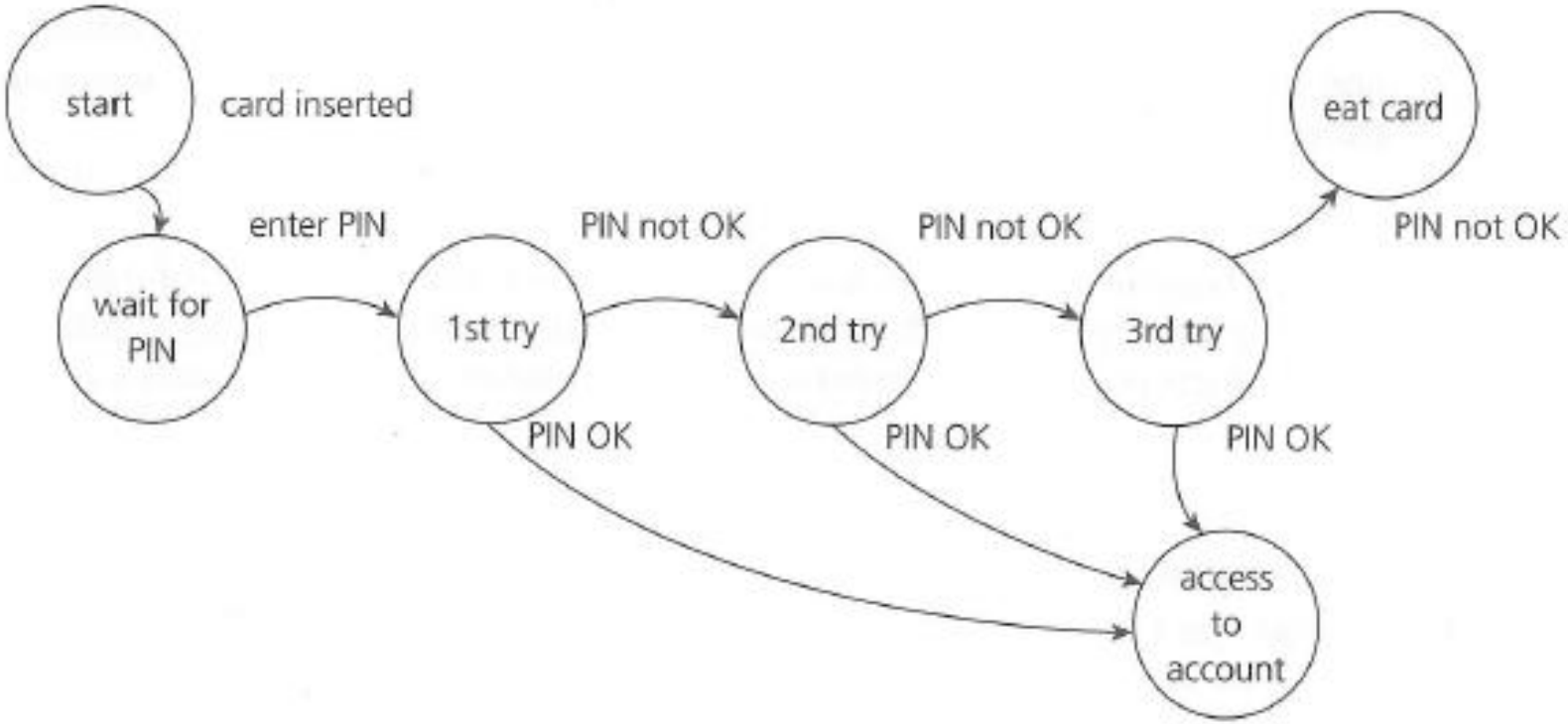
3. Specification-based techniques (black-box)
- *3.1 Equivalence partitioning*
- *3.2 Boundary value analysis*
- *3.3 Decision table testing*
- *3.4 State transition testing*
- *3.5 Use case testing*

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

Use case testing versus state transition testing



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

Use case testing

The PIN-Example

	Step	Description
Main Success Scenario **A: Actor** **S: System**	1	A: Inserts card
	2	S: Validates card and asks for PIN
	3	A: *Enters PIN*
	4	S: Validates PIN
	5	S: Allows access to account
Extensions	2a	Card not valid S: Display message and reject card
	4a	PIN not valid S: Display message and ask for re-try (twice)
	4b	PIN invalid 3 times S: Eat card and exit

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

Use case testing

- Use cases are very useful for designing acceptance tests with customer/user participation.
- Use cases describe the 'process flows' through a system base on its actual likely use.
- The test cases derived from use cases are most useful in uncovering defects in the process flows during real-world use of the system.
- They also help uncover integration defects caused by the integration and interference of different components, which individual testing would not see.
- Designing test cases from use cases may be combined with other specification-based test 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

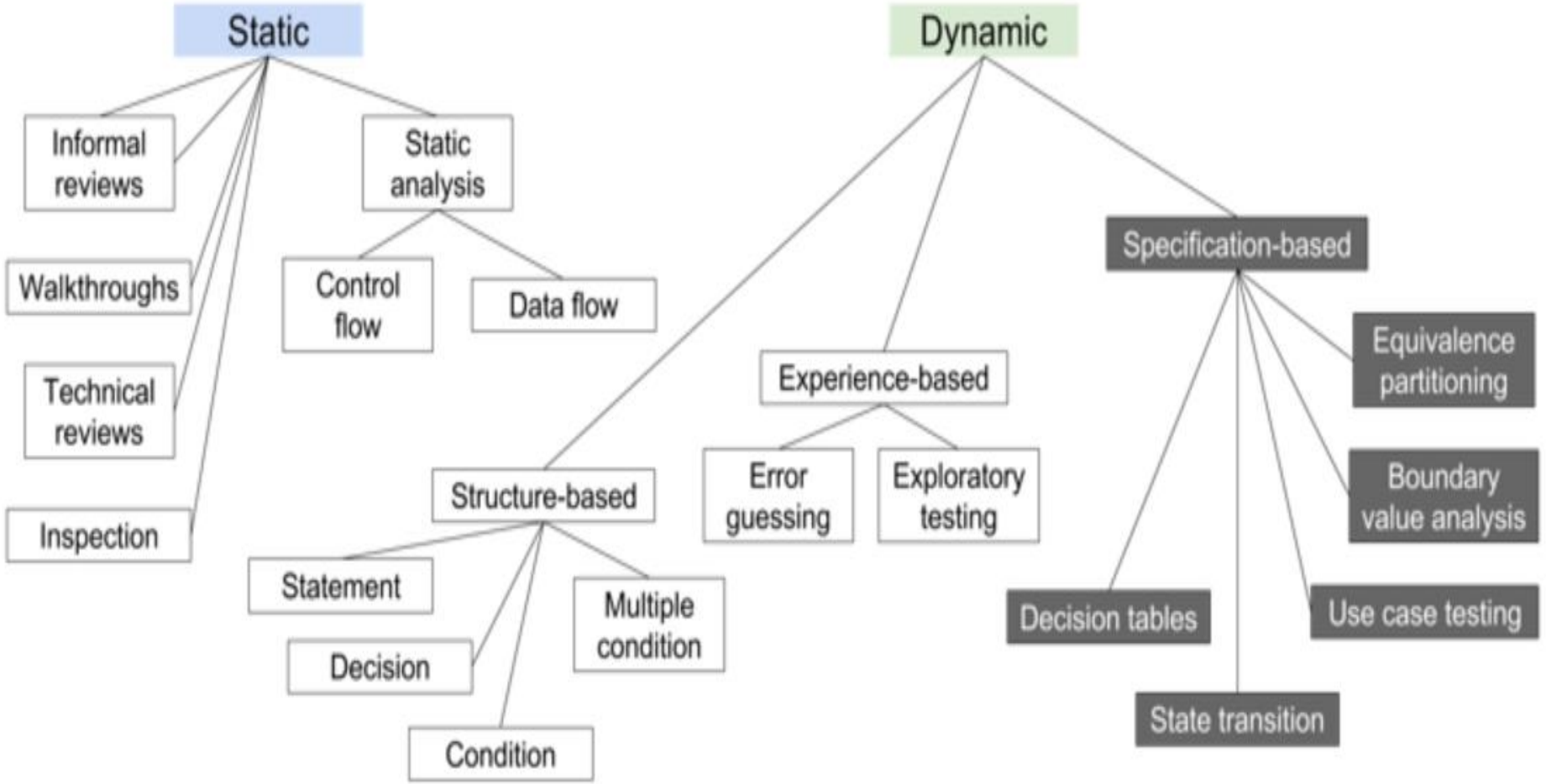
3. Specification-based techniques (black-box)
- *3.1 Equivalence partitioning*
- *3.2 Boundary value analysis*
- *3.3 Decision table testing*
- *3.4 State transition testing*
- *3.5 Use case testing*

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

Structure-based testing
White-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

Structure-based testing
White-box techniques

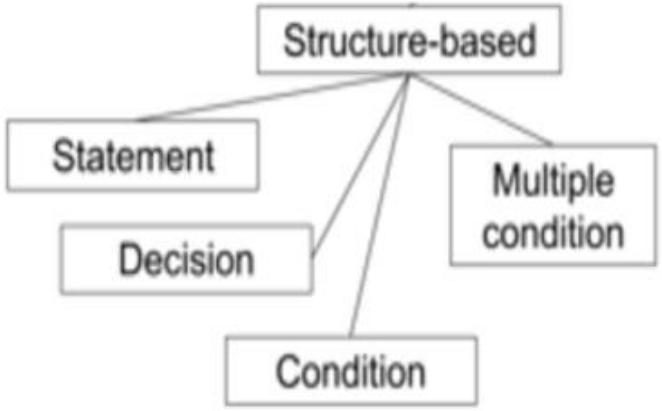
Structure-based techniques serve two purposes:

Test coverage measurement

We can assess the amount of testing performed by tests derived from e.g. specification-based technique to asses coverage.

Structural test case design

We can generate additional test cases with the aim of increasing 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

What is coverage?

$$\text{Coverage} = \frac{\text{Number of coverage items exercised}}{\text{Total number of coverage items}} \times 100\%$$

A coverage item is whatever we have been able to count and see whether a test has exercised or used this item.

NB! 100% coverage does *not* mean that 100% tested!

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

Structure-based testing
White-box techniques

Structure-based testing (white-box) is based on an identified **structure** of the software.

Component level - the structure is that of the code itself:
- statements, decisions/branches.

Integration level - the structure may be a call tree
(a diagram in which modules call other modules).

System level - the structure may be a menu structure, business process or web page structure.

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

How to measure coverage?

We can measure coverage from a number of different structure elements in a system or component.

We can also measure coverage of each of the *specification based techniques*:

- EP: percentage of equivalence partitions exercised
- BVA: percentage of boundaries exercised
- Decision table: percentage business rules or decision table column tested
- State transition testing: percentage of
  - states visited
  - transitions exercised
  - legal transitions exercised
  - invalid transitions exercised

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

How to measure coverage?

The steps typically taken to measuring coverage is useful in understanding the relative merits of each technique:

1. Decide on the structural element used, i.e. the coverage items to be counted.
2. Count the structural elements or items.
3. Instrument the code.
4. Run the tests for which the coverage measurement is required.
5. Using the output from the instrumentation, determining the percentage of elements or items exercised.

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

Statement testing & coverage

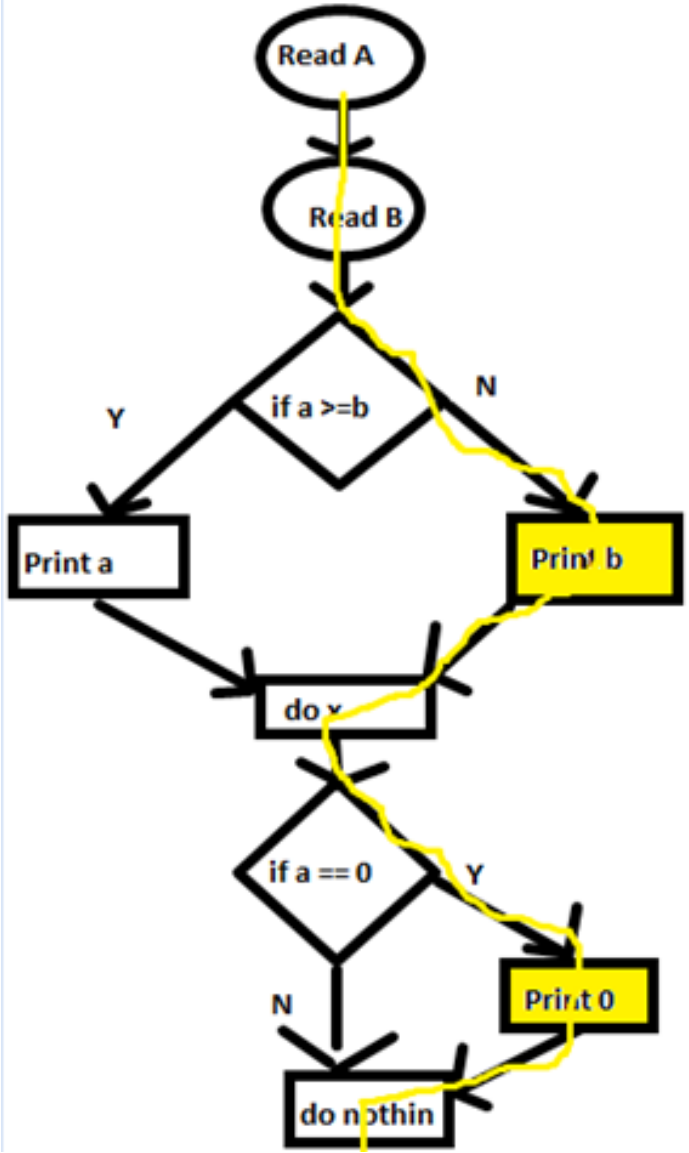
In component testing

Statement coverage is the percentage of executable statements that have been exercised by a test case suite.

The statement testing technique derives test cases to execute specific statements, normally to increase statement coverage.

Statement coverage =

$$\frac{\text{Number of statements exercised}}{\text{Total number of statements}} \times 100\ \%$$



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

Statement coverage - example

Example

If you are flying with an economy ticket, there is a possibility that you may get upgraded to business class, especially if you hold a gold card in the airline's frequent flyer program.

If you don't hold a gold card, there is a possibility that you will get "bumped" off the flight if it is full and you check in late.

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

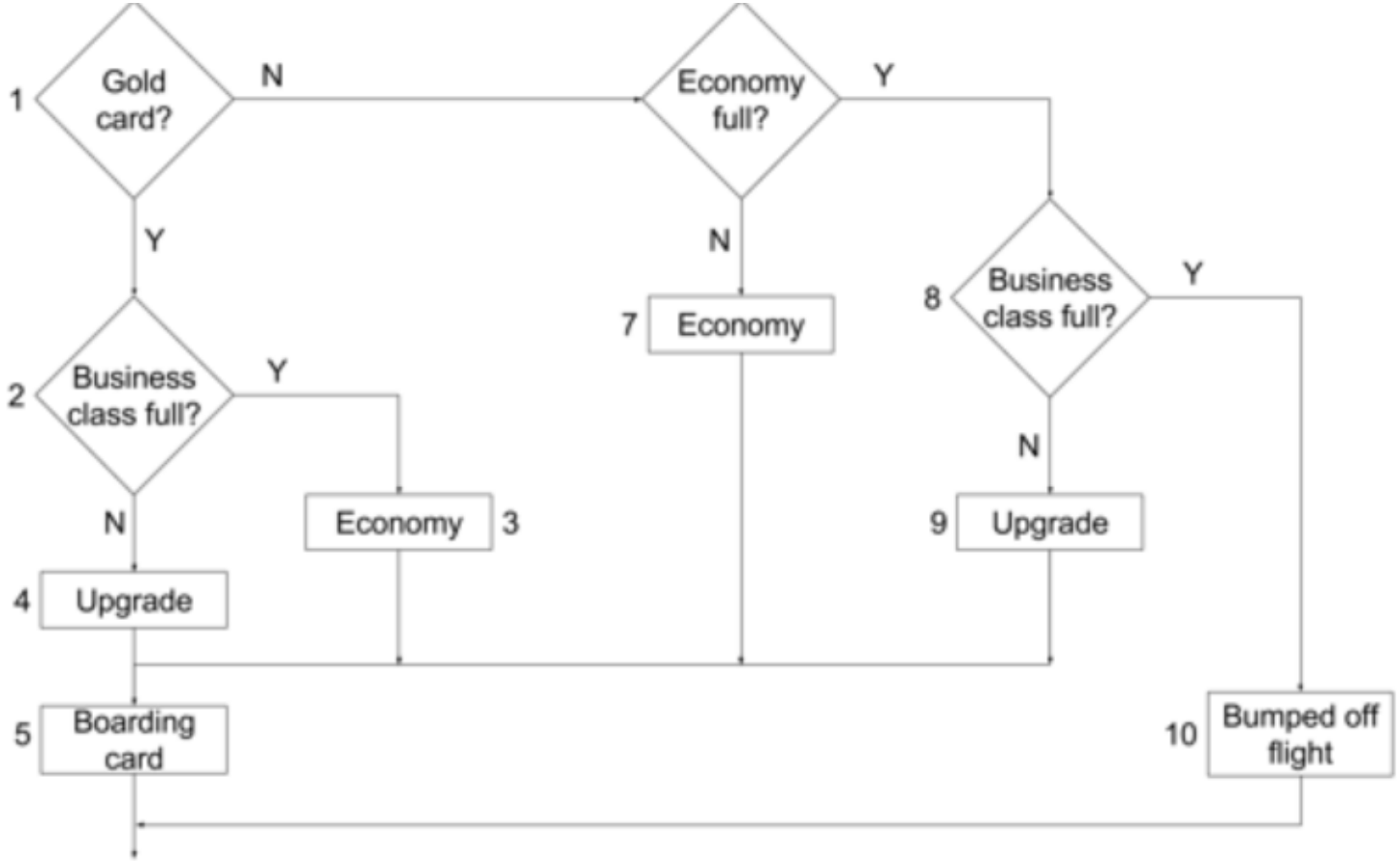
Statement coverage - example

Tests run:

Test 1
Gold card holder who gets upgraded to business class

Test 2
Non-gold card holder who stays in economy

Test 3
A person who is bumped off the flight



What is the statement coverage of these three 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

Statement coverage - example

What is the statement coverage of these three tests?

Calculating statement coverage

$$Statement\ coverage = \frac{Number\ of\ statements\ exercised}{Total\ number\ of\ statements} \times 100$$

After running all three tests:

Numerator: How many statements have we exercised?

Denominator: How many statements exist in total?

Multiply by a hundred to get percentage

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

Statement coverage - example

What is the statement coverage of these three tests?

Test 1

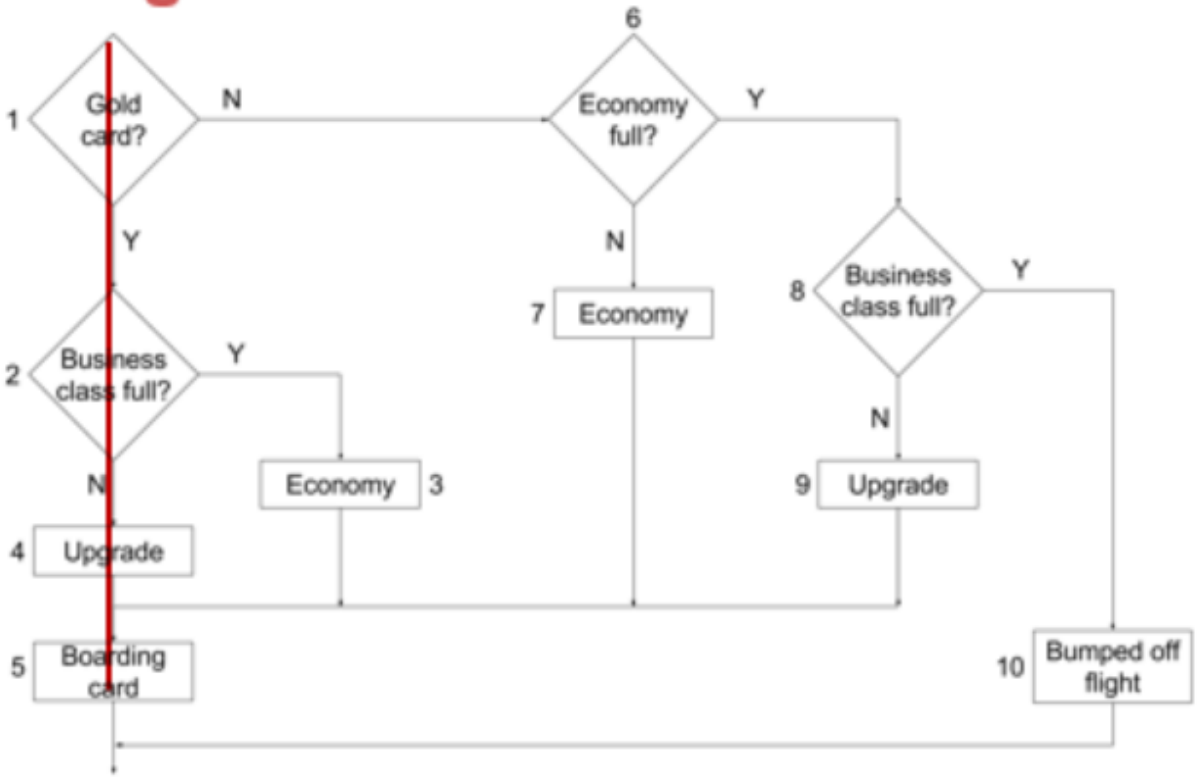
Gold card holder

Upgraded to business class

Coverage

Total statements: 10

Statements so far: 4



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

Statement coverage - example

What is the statement coverage of these three tests?

Test 2

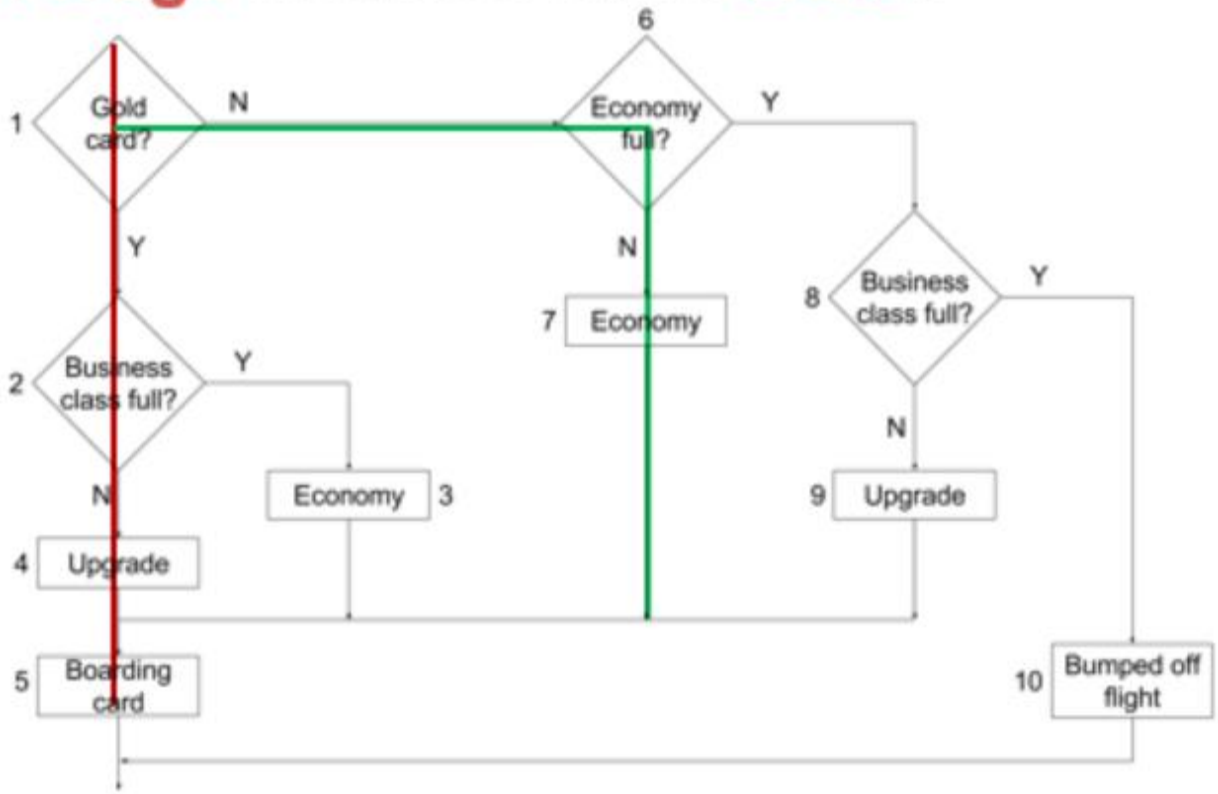
Non-gold card holder

Stays in economy

Coverage

Total statements: 10

Statements so far: 6



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

Statement coverage - example

What is the statement coverage of these three tests?

Test 3

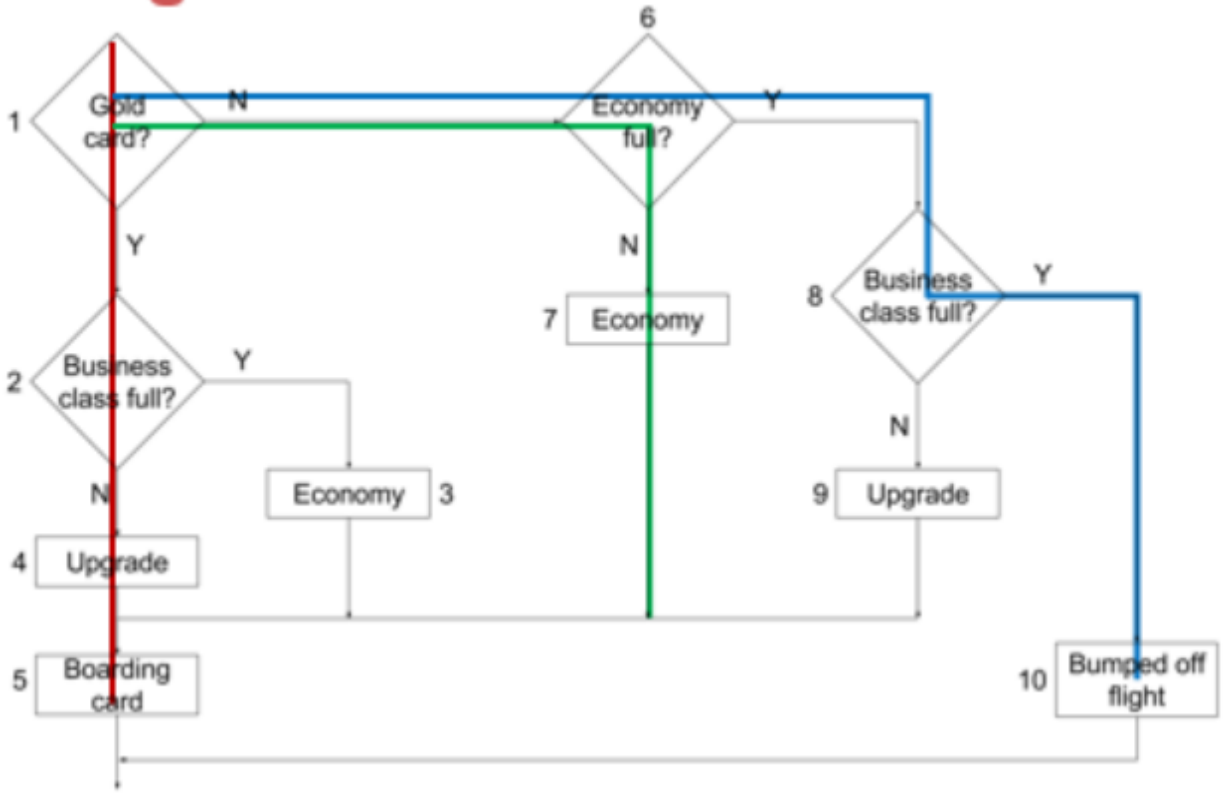
Any person

Bumped off the flight

Coverage

Total statements: 10

Statements so far: 8



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

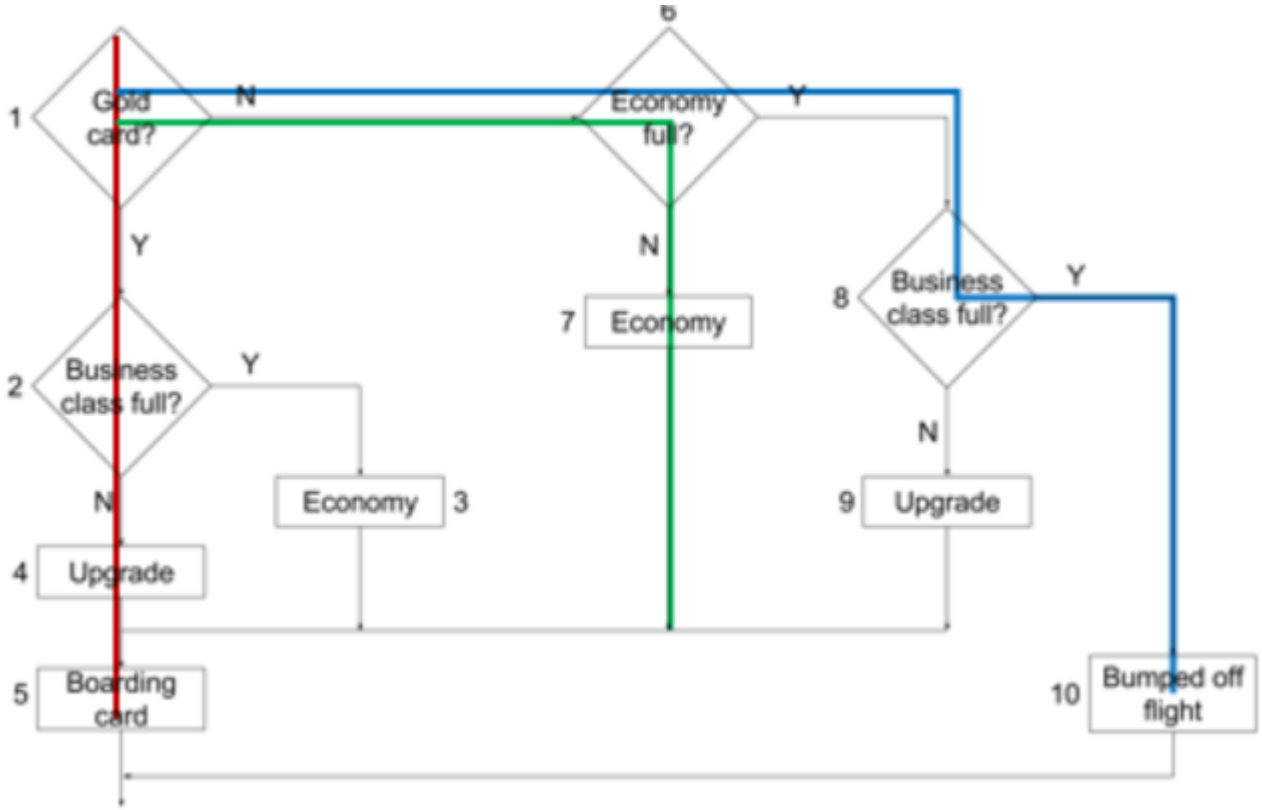
6. Choosing test techniques

Statement coverage - example

How many statements have we exercised? 8

How many statements exist in total? 10

Statement coverage? 80%



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

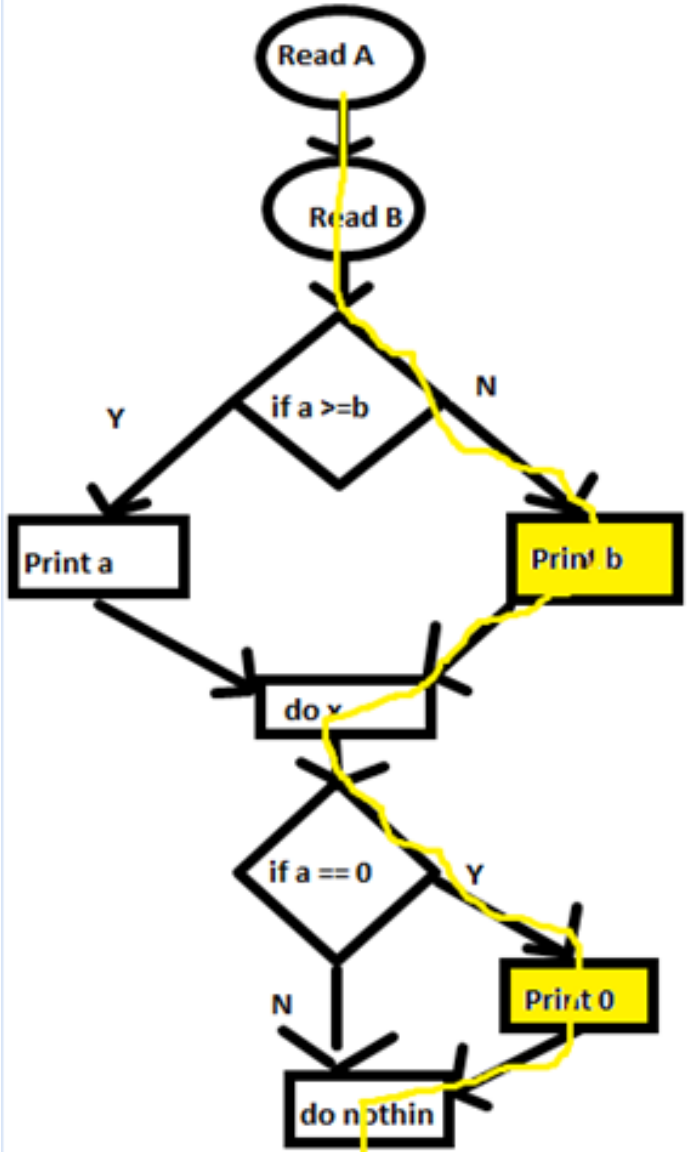
6. Choosing test techniques

Decision testing & coverage

Decision coverage - is the assessment of the percentage of decision outcomes

(e.g. the True and False options of an IF statement) that have been exercised by a test case suite.

$$\frac{\text{Number of decision outcomes exercised}}{\text{Total number of decision outcomes}} \times 100\ \%$$



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

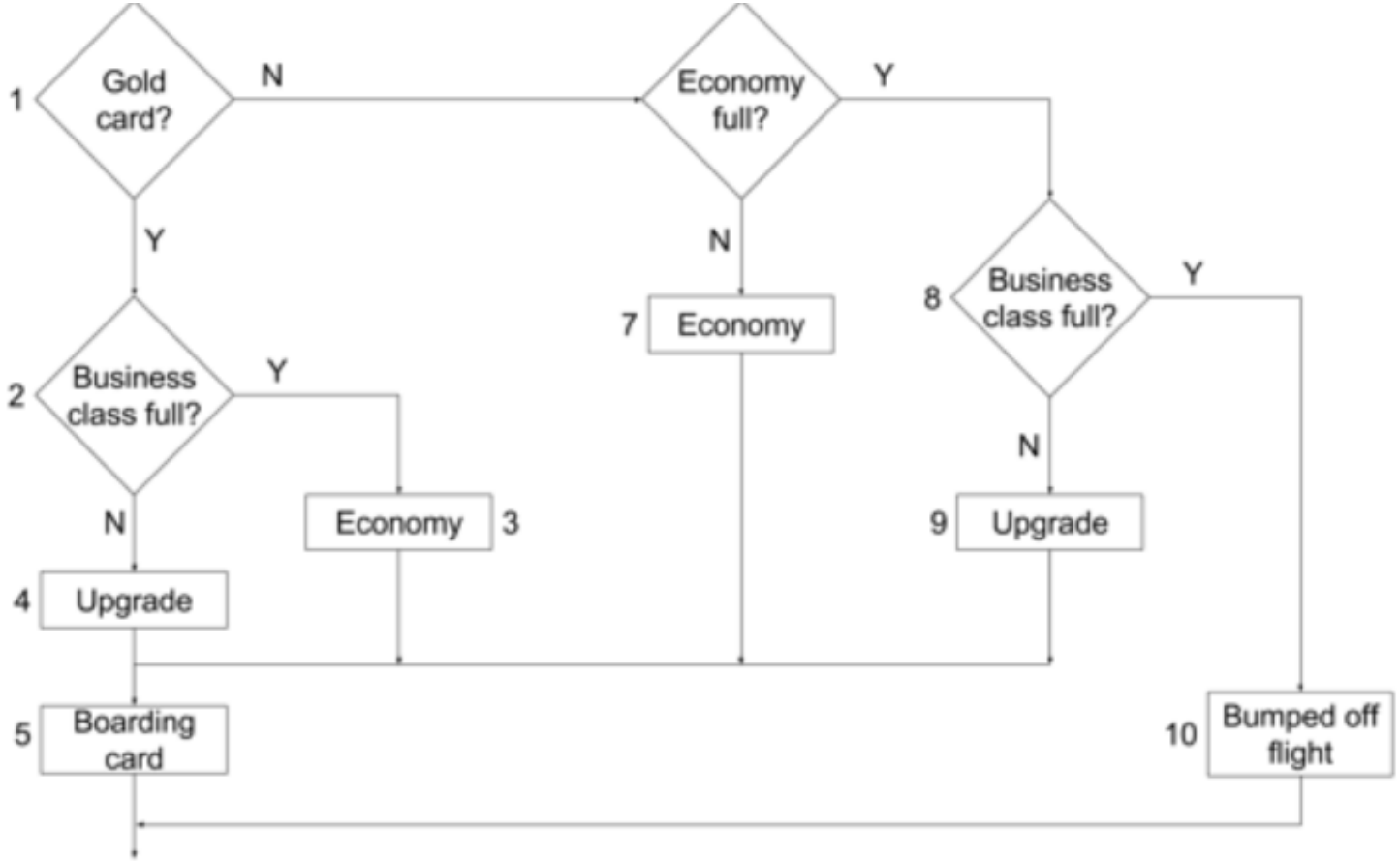
Decision coverage - example

Tests run:

Test 1
Gold card holder who gets upgraded to business class

Test 2
Non-gold card holder who stays in economy

Test 3
A person who is bumped off the flight



What is the decision coverage of these three 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*

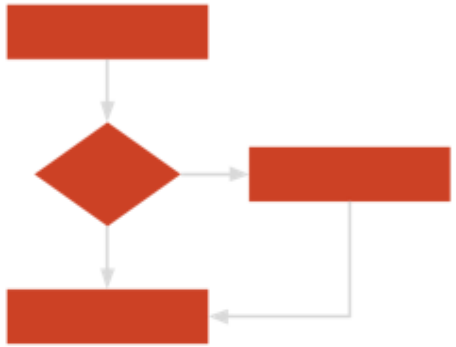
5. Experience-based techniques

6. Choosing test techniques

statement/decision coverage

Statement coverage

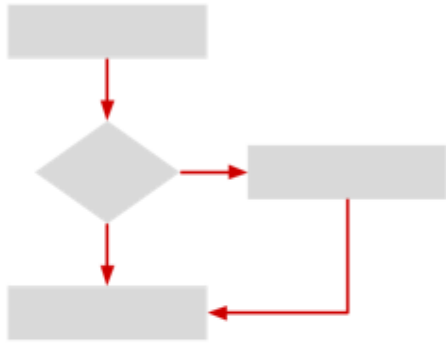
Percentage of statements exercised



$$\text{Statement coverage} = \frac{\text{Number of statements exercised}}{\text{Total number of statements}} \times 100$$

Decision coverage

Percentage of decisions exercised



$$\text{Decision coverage} = \frac{\text{Number of decision outcomes exercised}}{\text{Total number of decision outcomes}} \times 100$$

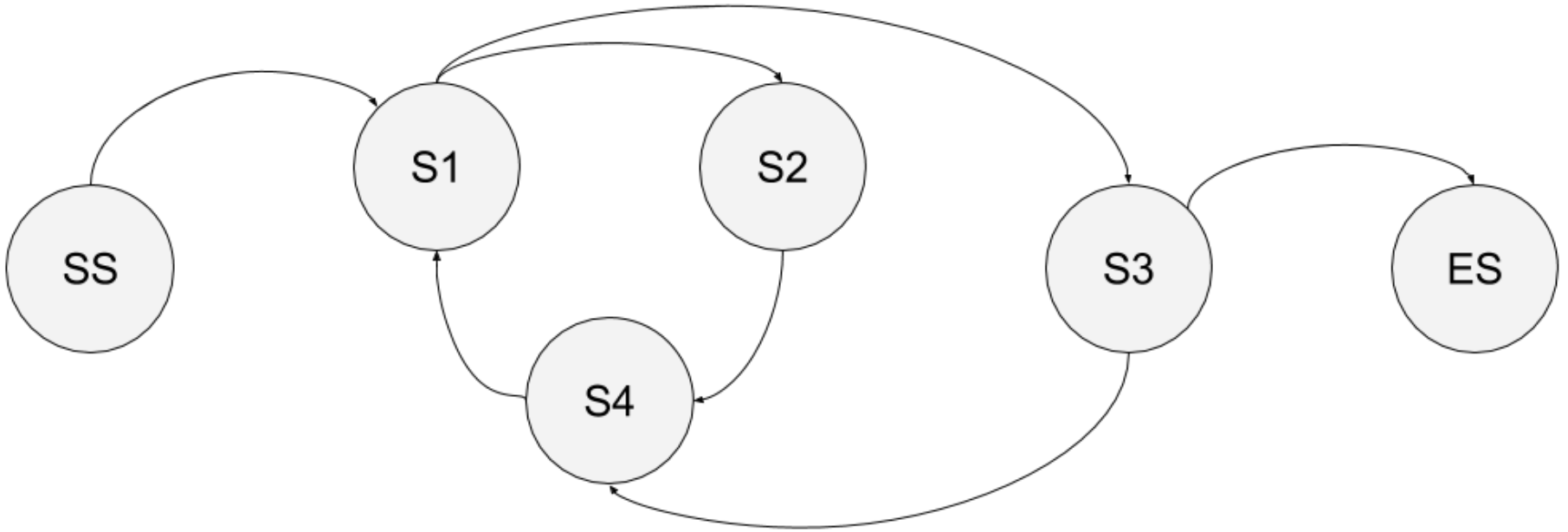
Decision coverage is stronger than statement coverage

100% decision coverage guarantees 100% statement coverage

Not the other way around!

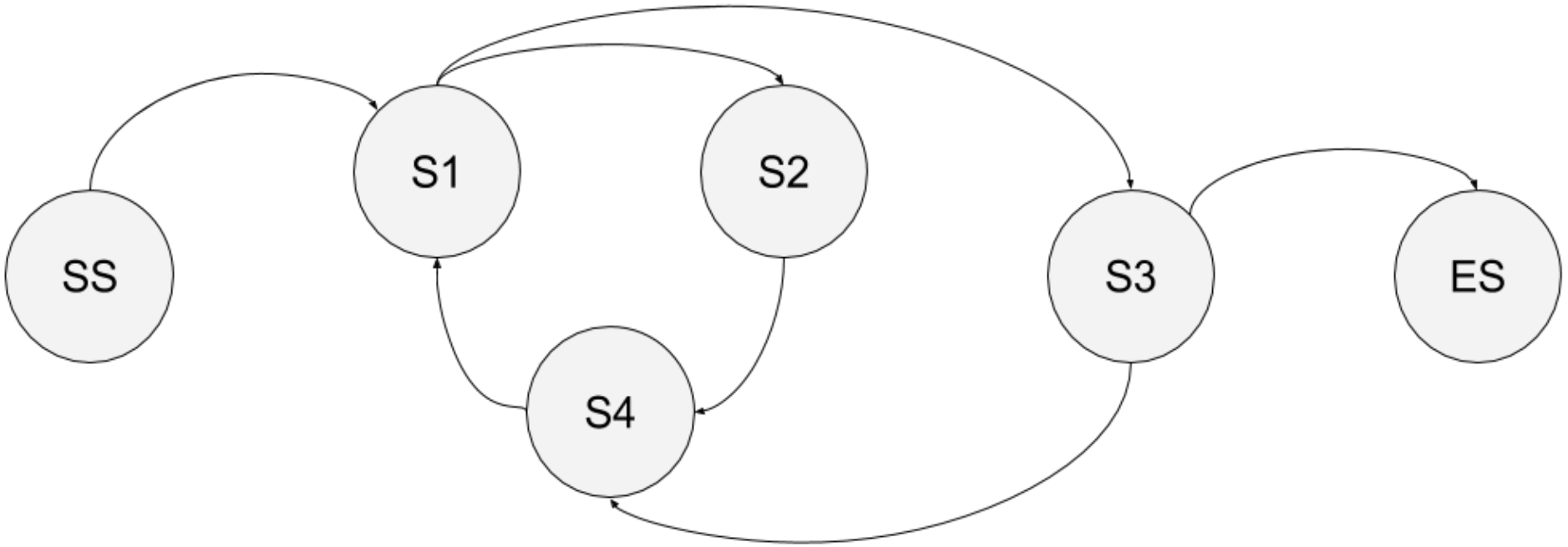
State transition testing and coverage

Given the state diagram below, which test case is the minimum series of valid transitions to cover every state?



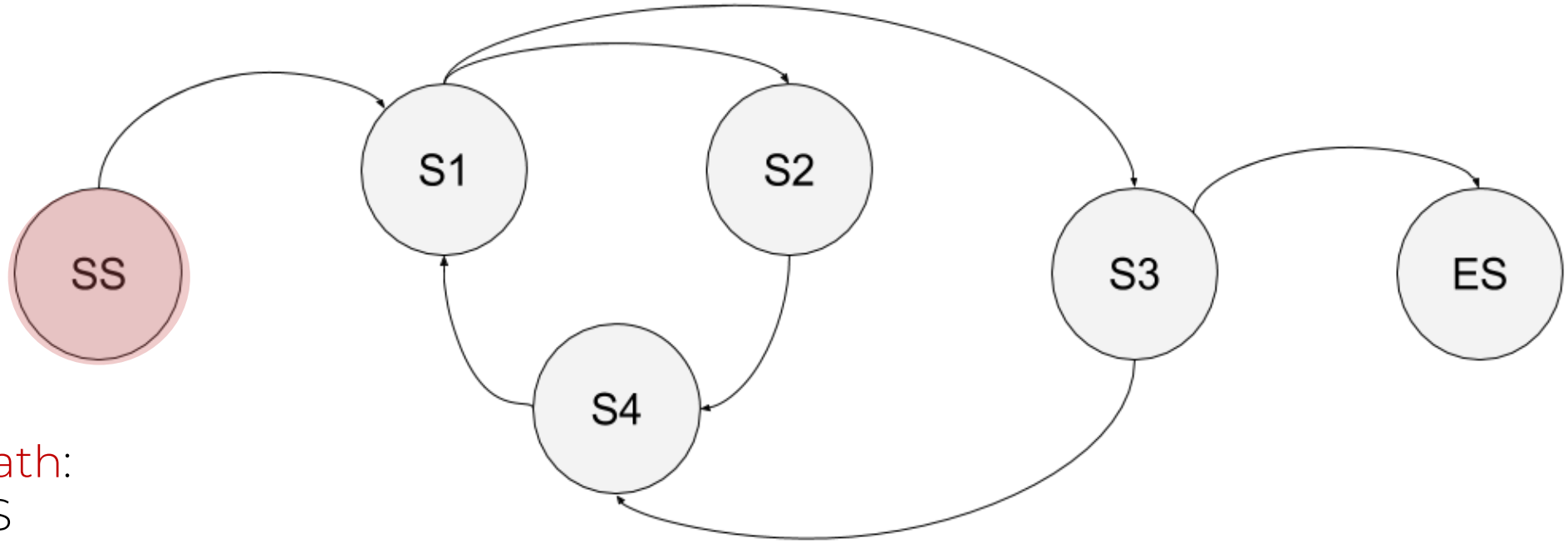
State transition testing and coverage

Want the minimum path from SS to ES, visiting each state at least once, i.e. 100% state coverage.



State transition testing and coverage

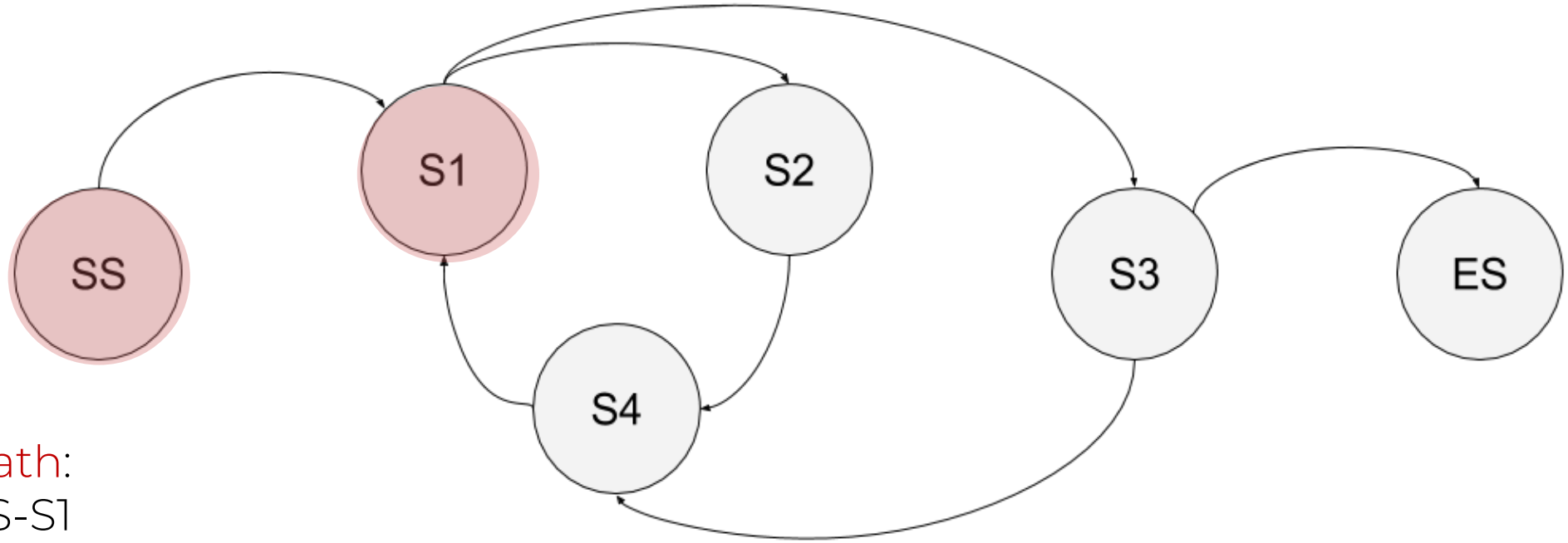
Want the minimum path from SS to ES, visiting each state at least once, i.e. 100% state coverage.



Path:
SS

State transition testing and coverage

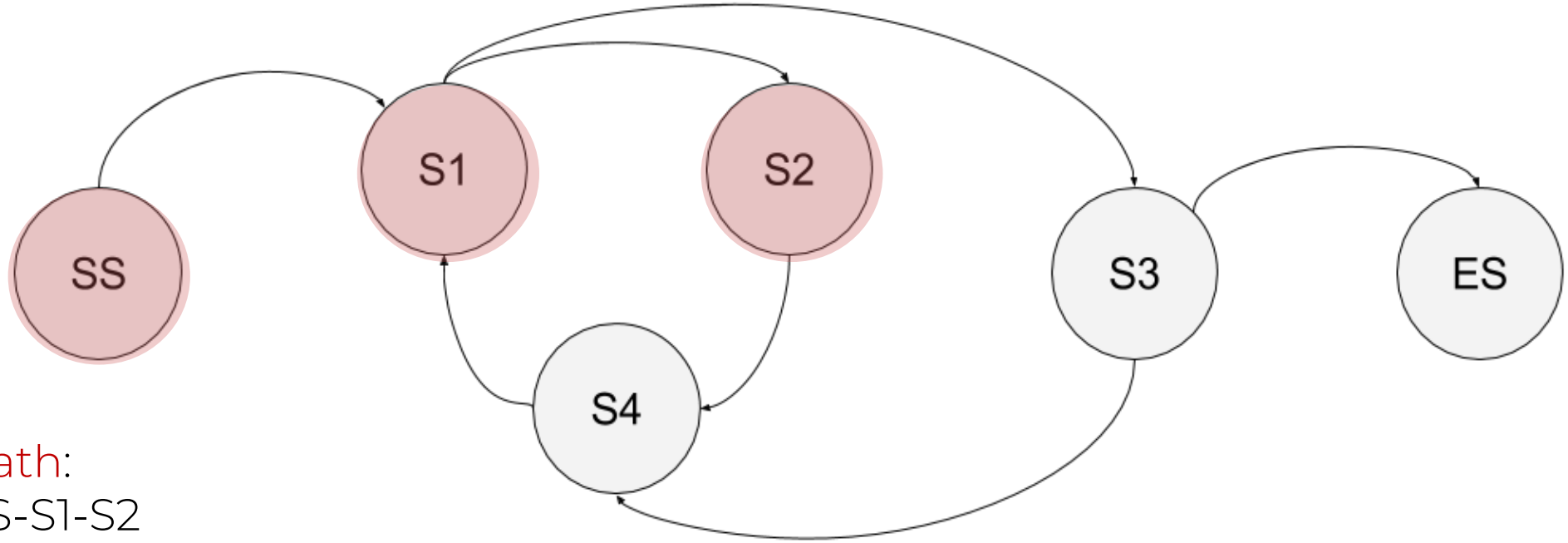
Want the minimum path from SS to ES, visiting each state at least once, i.e. 100% state coverage.



Path:
SS-S1

State transition testing and coverage

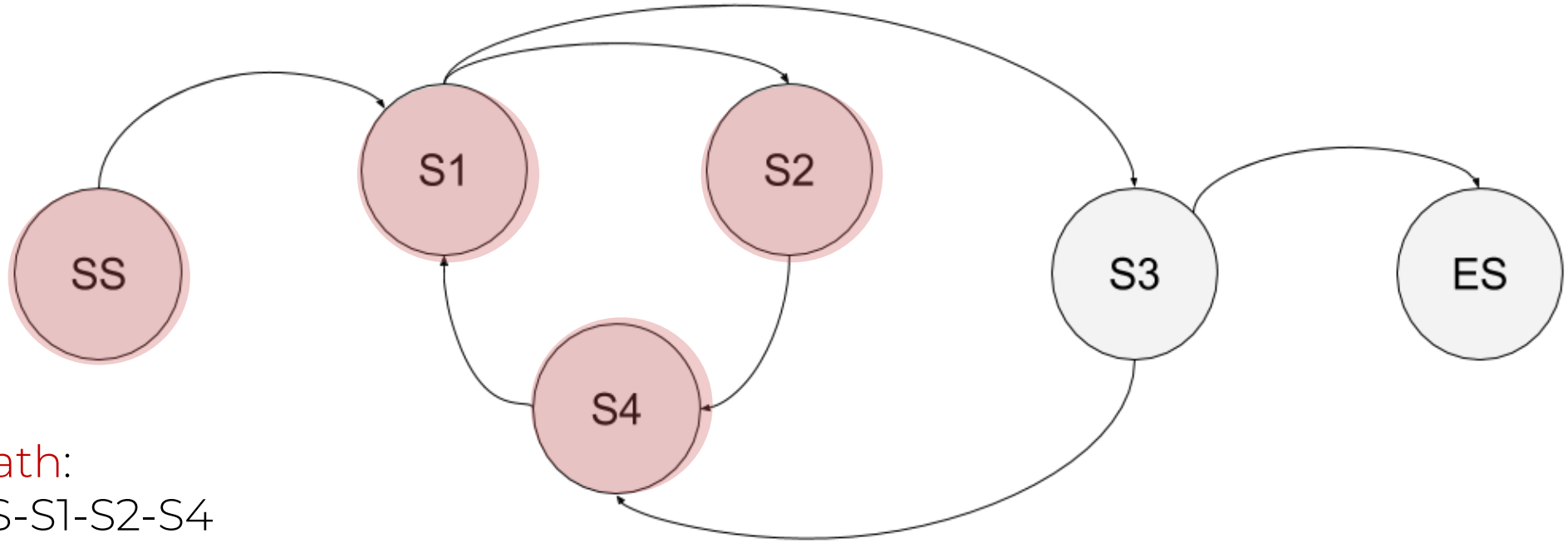
Want the minimum path from SS to ES, visiting each state at least once, i.e. 100% state coverage.



Path:
SS-S1-S2

State transition testing and coverage

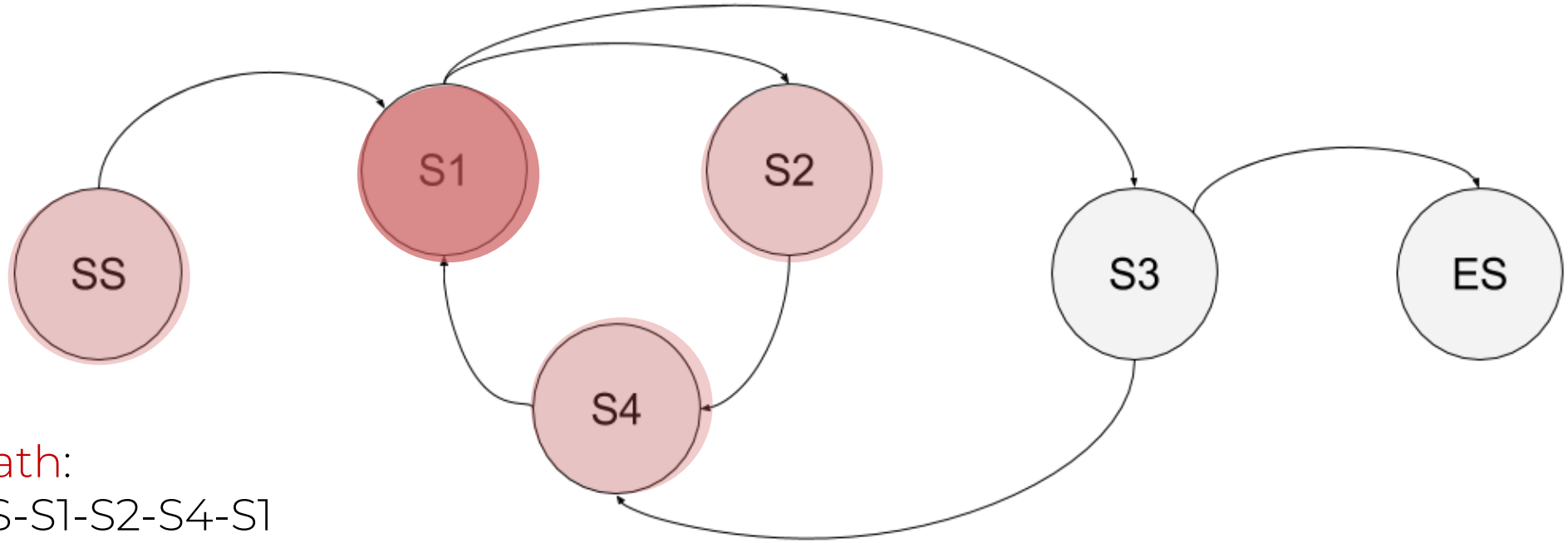
Want the minimum path from SS to ES, visiting each state at least once, i.e. 100% state coverage.



Path:
SS-S1-S2-S4

State transition testing and coverage

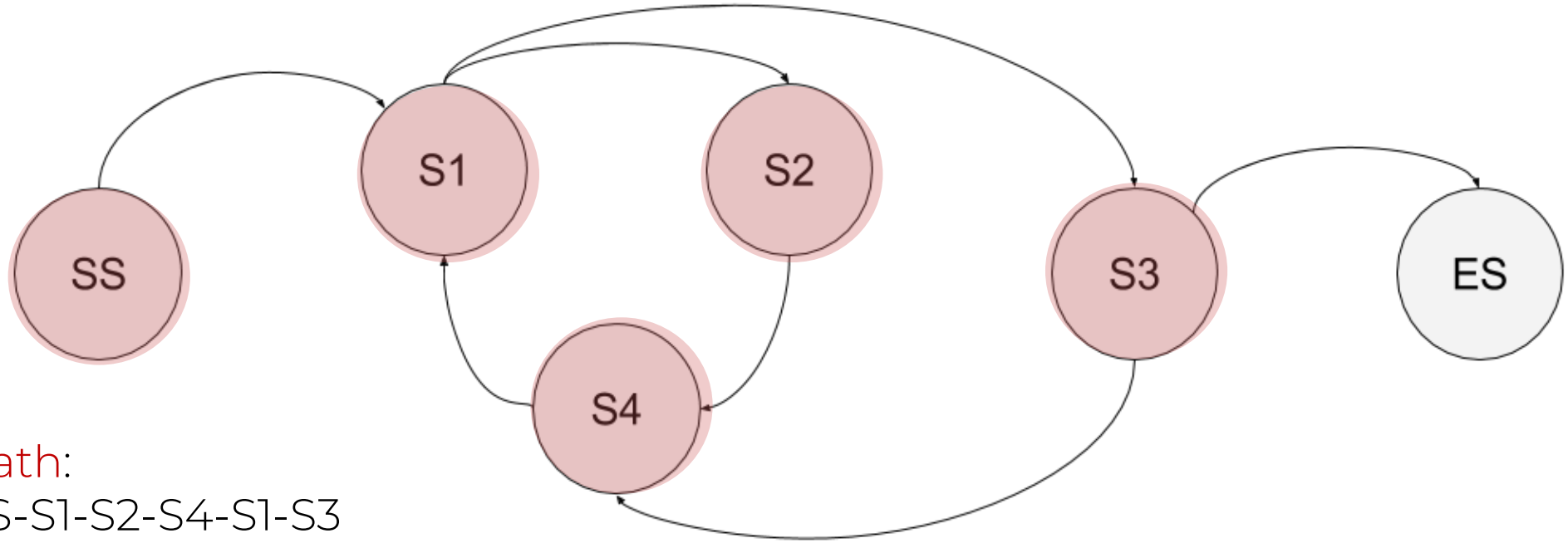
Want the minimum path from SS to ES, visiting each state at least once, i.e. 100% state coverage.



Path:
SS-S1-S2-S4-S1

State transition testing and coverage

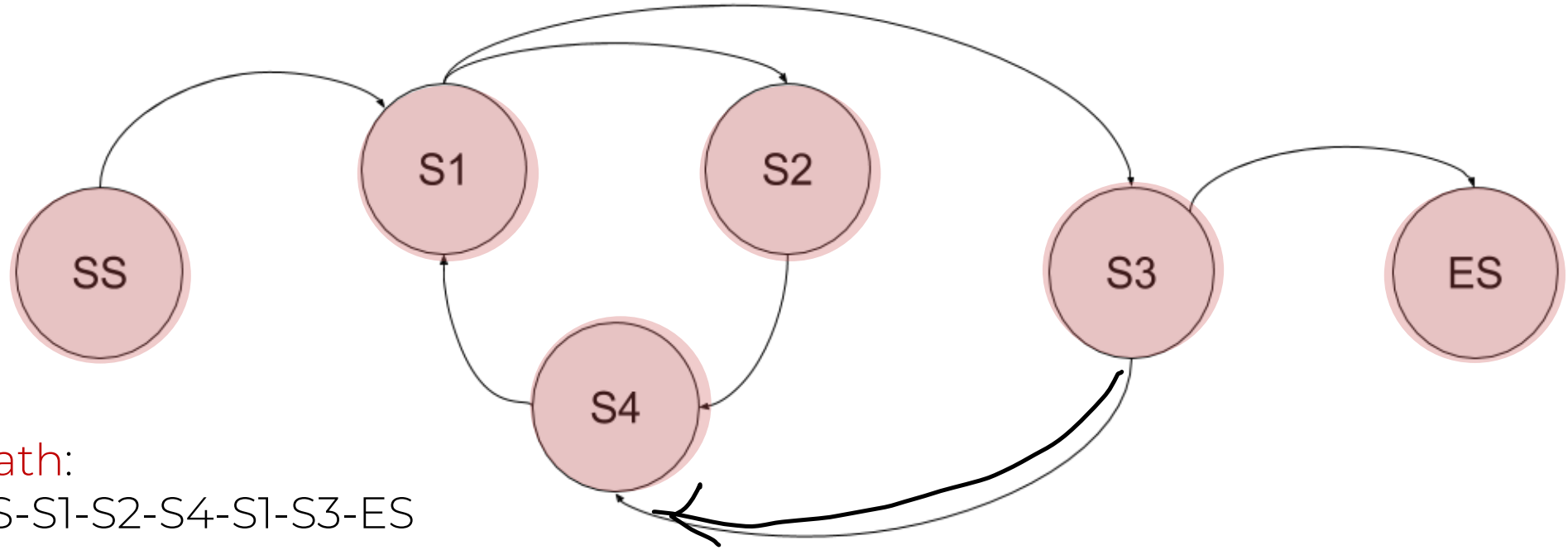
Want the minimum path from SS to ES, visiting each state at least once, i.e. 100% state coverage.



Path:
SS-S1-S2-S4-S1-S3

State transition testing and coverage

Want the minimum path from SS to ES, visiting each state at least once, i.e. 100% state coverage.



Path:
SS-S1-S2-S4-S1-S3-ES

State transition testing and coverage

Transition coverage is stronger than state coverage

100% transition coverage guarantees 100% state coverage

Not the other way around!

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

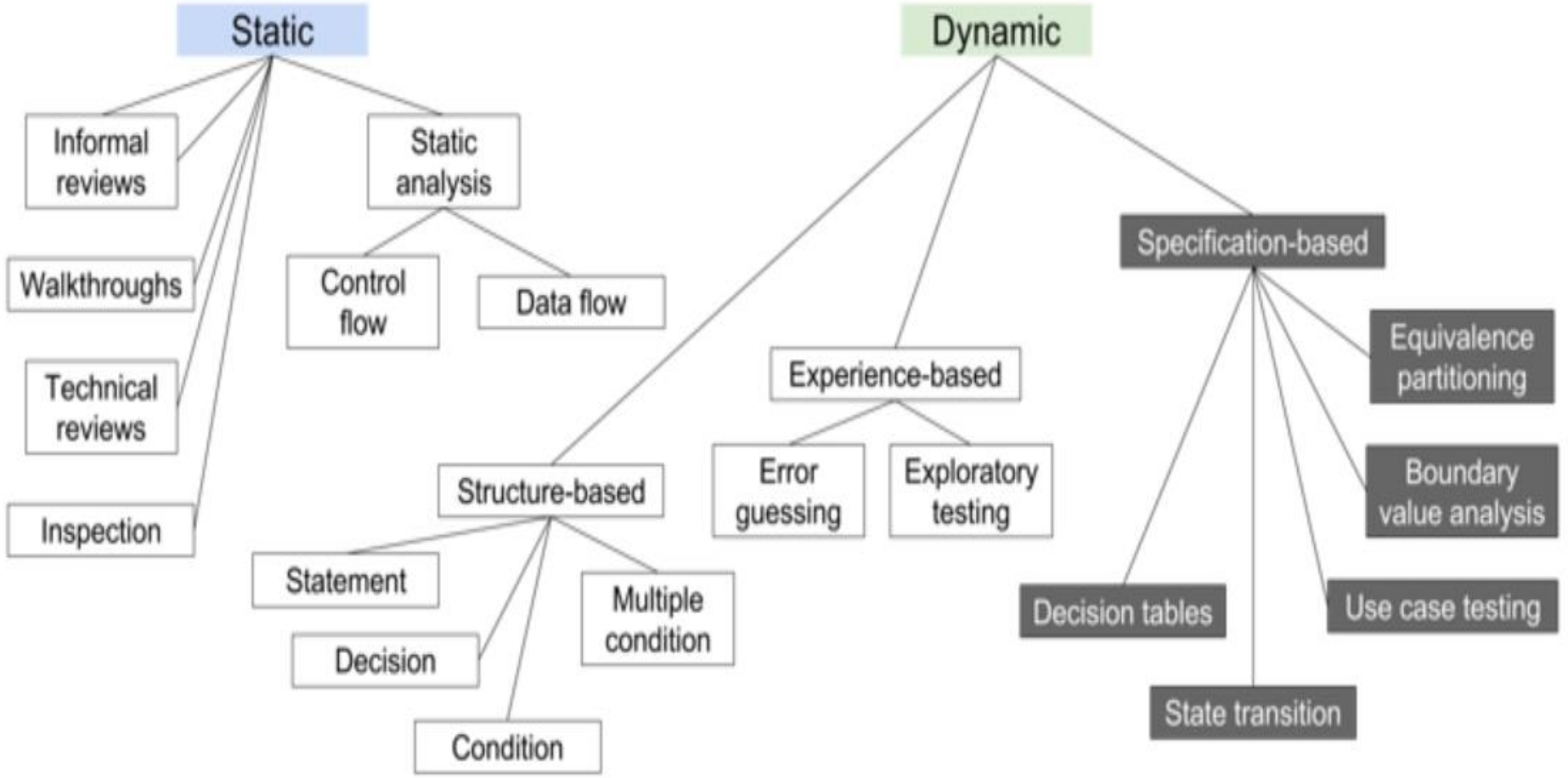
3. Specification-based techniques (black-box)
- *3.1 Equivalence partitioning*
- *3.2 Boundary value analysis*
- *3.3 Decision table testing*
- *3.4 State transition testing*
- *3.5 Use case testing*

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

Experience-based testing



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

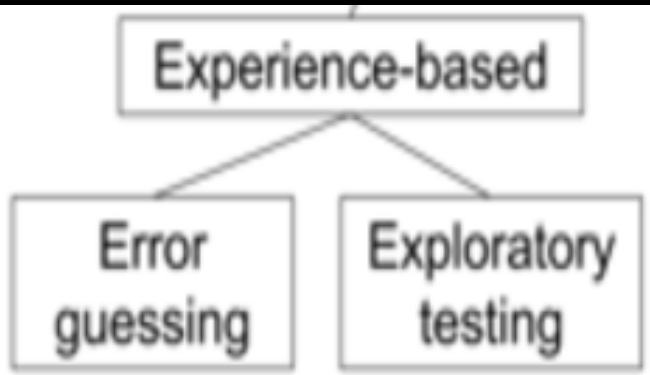
3. Specification-based techniques (black-box)
- 3.1 Equivalence partitioning
- 3.2 Boundary value analysis
- 3.3 Decision table testing
- 3.4 State transition testing
- 3.5 Use case testing

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

Experience-based testing



- Tests are derived from the tester's skill and intuition and their experience with similar applications and technologies.
- When used to augment systematic techniques, experienced-based testing can be useful in identifying special tests not easily captured by formal techniques, especially when applied after more formal approaches.
- May yield widely varying degrees of effectiveness, depending on the testers experience.

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

Experience-based testing

Error guessing = a commonly used experienced-based technique.

Generally testers anticipate defects based on experience.

A structured approach to the error guessing technique is to enumerate a list of possible errors and to design tests that attack these errors.

This systematic approach is called fault attack.

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

Experience-based testing

Exploratory testing = **concurrent** **test design**, **test execution**, **test logging and learning**,

based on a test charter containing test objectives,
and *carried out within time-boxes*.

It is most useful …

- where there are few or inadequate specifications
- under severe time pressure
- to complement other, more formal testing
- It can serve to help ensure that the most serious defects are found.

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

Choosing test techniques

The choice of which test techniques to use depends on a number of factors, including:

