Техніки тест дизайну

План лекції

Тест дизайн та його категорії

Specification based or Black box techniques

Experience based techniques

Вибір тестових технік

Тест дизайн та його категорії

Test design

Test design - методика, що використовується для створення та/або вибору тестових сценаріїв.

В цілому можна розділити тестування на дві великі категорії.

1

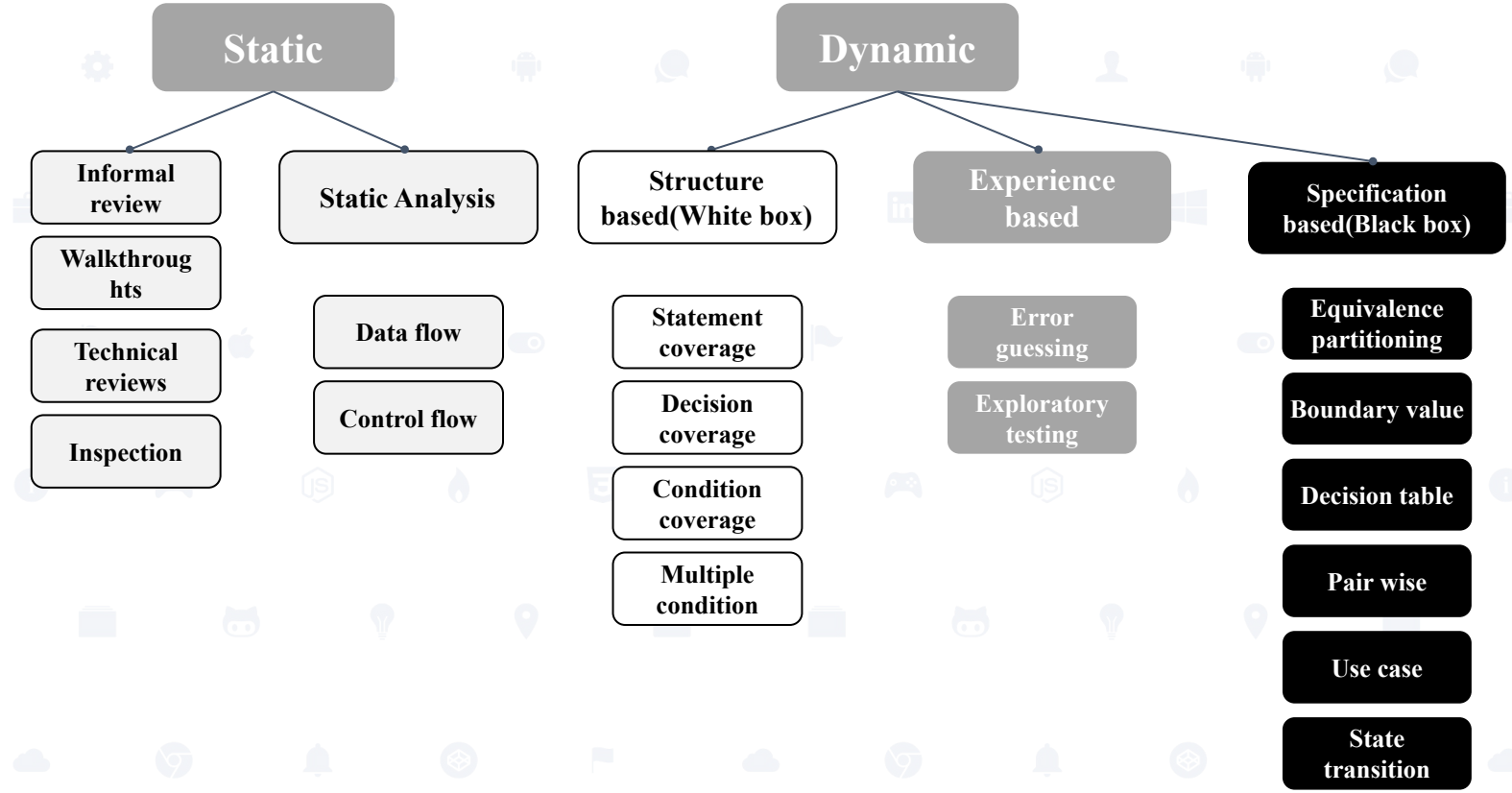
Static testing

тип тестування, який передбачає, що програмний код під час тестування не виконуватиметься.

2

Dynamic testing

тип тестування, що передбачає запуск програмного коду.



Це базові техніки, які описані в foundation level. Їх може бути більше

Specification based or Black box techniques

Specification based

Black box техніки - цей вид техніки базується на основі (документації, специфікації, use cases, бізнес процесів). Ці методи застосовні як для функціонального, так і для нефункціонального тестування. Методи тестування чорної скриньки зосереджені на входах і виходах об’єкта тестування без посилання на його внутрішню структуру.

Equivalence partitioning

Equivalence partitioning

Equivalence partition - це метод тестування програмного забезпечення, який ділить вхідні дані програмного блоку на розділи еквівалентних даних, з яких можна отримати контрольні приклади. Існують класи еквівалентності як для позитивних, так і для негативних значень.

Equivalence partitioning - розробка тестів методом чорної скриньки, тестові сценарії створюються для перевірки елементів еквівалентної області. Як правило, тестові сценарії розробляються для покриття кожної області як мінімум один раз.

1. **Позитивні значення** – це значення, які повинні бути прийняті компонентом або системою. Еквівалентний клас, що містить дійсні значення, називається «дійсним розділом еквівалентності».

2. **Негативне значення** – це значення, які повинні бути відхилені компонентом або системою. Еквівалентний клас, що містить недійсні значення, називається «недійсним розділом еквівалентності».

3. Розділи можна ідентифікувати для будь-якого елемента даних, пов’язаного з тестовим об’єктом, включаючи вхідні дані, виходи, внутрішні значення, значення, пов’язані з часом (наприклад, до або після події) та параметри для інтерфейсу (наприклад, інтегровані компоненти, що тестуються під час інтеграційного тестування).

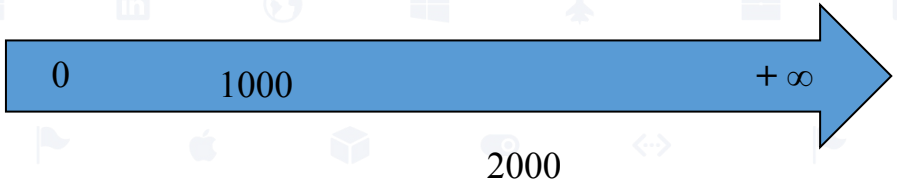
4. Якщо в тестових випадках використовуються негативні розділи еквівалентності, їх слід тестувати індивідуально, тобто не поєднувати з іншими негативними розділами еквівалентності, щоб гарантувати, що помилки не маскуються.
Збої можуть бути замасковані, коли кілька збоїв відбуваються одночасно, але помітен лише один, через що інші збої залишаються непоміченими.

Як працює техніка на практиці

Задача

Фітнес трекер рахує кількість кроків, які ви проходите за день. Після цього в кінці дня надсилає вам результат:

- Якщо ви пройшли 1000м і менше - меседж “Картопля”
- Якщо результат 1000 до 2000 - “Непогано”
- Якщо більше - “Дуже добре”



Класи:

- 0 - 1000
- 1001 - 2000
- 2001 - + ∞

Практика

Для складання іспиту студент повинен скласти тест з 40 питань, з них відповісти мінімум на 25, щоб отримати 3 бали. Якщо студент відповів до 90%, оцінка складає 4 бали. Вище — 5 балів. Визначте еквівалентні класи у цій задачі?

Boundary values analysis

Boundary value analysis

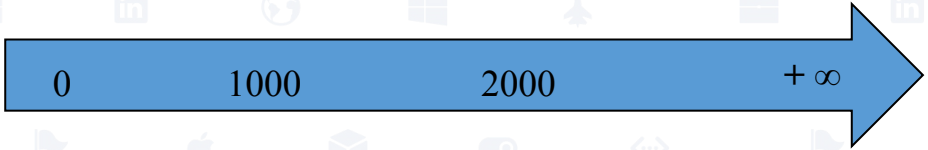
Boundary value - вхідне значення або вихідні дані, що знаходяться на межі еквівалентної області або на найменшій відстані від обох сторін грані, наприклад, мінімальне чи максимальне значення області.

Boundary value analysis - Розробка тестів методом чорної скриньки, при якому тестові сценарії проектуються виходячи з граничних значень.

Задача

Фітнес трекер рахує кількість кроків, які ви проходите за день. Після цього в кінці дня надсилає вам результат:

- Якщо ви пройшли 1000м і менше - меседж “Картопля”
- Якщо результат більше 1000 - 2000 - “Непогано”
- Якщо більше 2000 - “Дуже добре”



Значения:

- 0 - 1000
- 1001 - 2000
- 2001

Ще для більшої впевненості використовують проміжні значення:

- 0 - 500 - 1000
- 1001 - 1500 - 2000
- 2001

Практика

Якщо температура опускається нижче 18 градусів, включається обігрів. Коли температура досягає 21 градуса, опалення відключається. Який мінімальний набір тестових вхідних значень для покриття всіх дійсних розділів еквівалентності?

Decision tables testing

Decision table

Decision table - таблиця, що відображає комбінації вхідних даних та/або причини з відповідними вихідними даними та/або діями (наслідками), які можуть бути використані для проектування тестових сценаріїв.

Приклад

Conditions/Input	Rule 1	Rule 2	Rule3	Rule4
Age > 23	T	T	F	F
Clean Driving record	T	F	T	F
Action/Output				
Supply rental car	Y	N	N	N

Практика

Для отримання акцій від компанії існують наступні умови. Необхідно пропрацювати більше 5 років, та бути QA чи девелопер. Якщо ти тестувальник, отримуеш 10000 акцій, якщо розробник - 5000!))) Побудуйте Decision table.

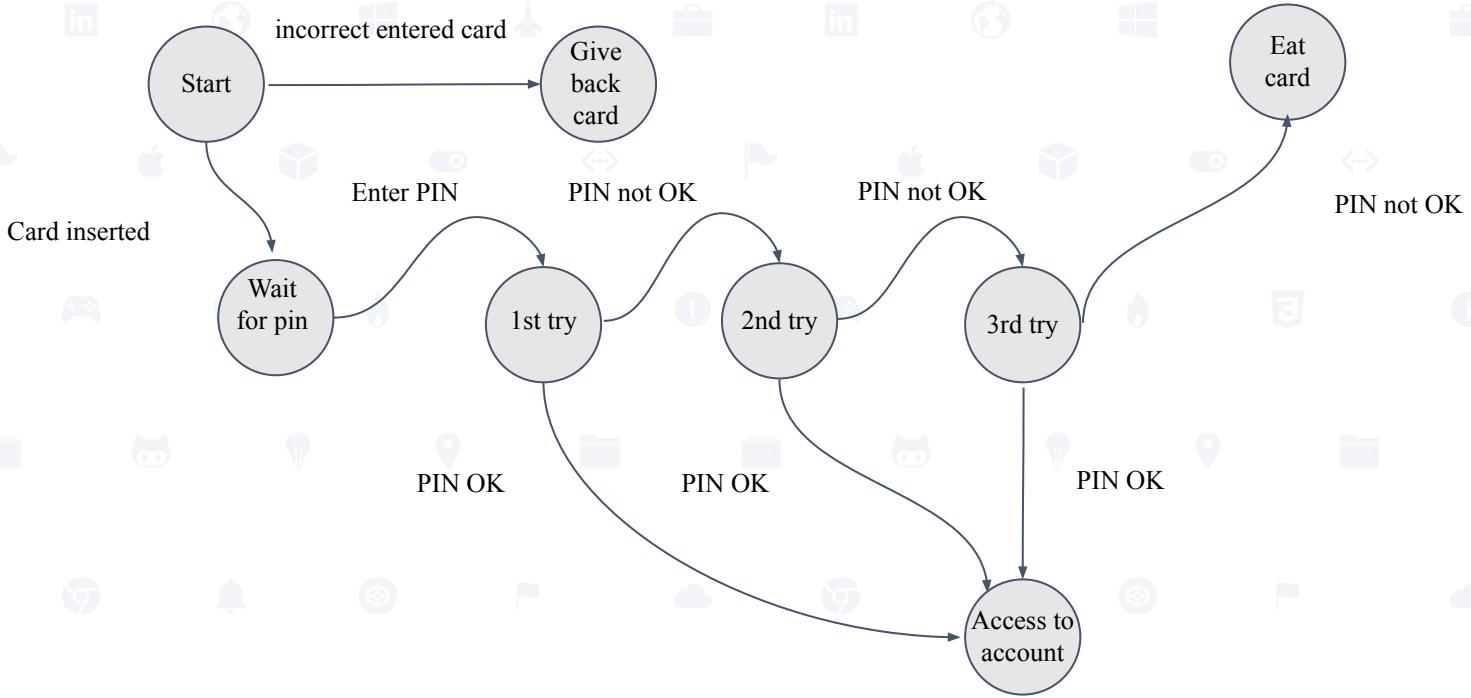
State transition testing

State transition testing

State transition testing - розробка тестів методом чорного ящика, при якому сценарії тестування будуються на основі виконання коректних та некоректних переходів станів.

State transition - перехід між двома станами компонентів чи системи.

State transition diagram



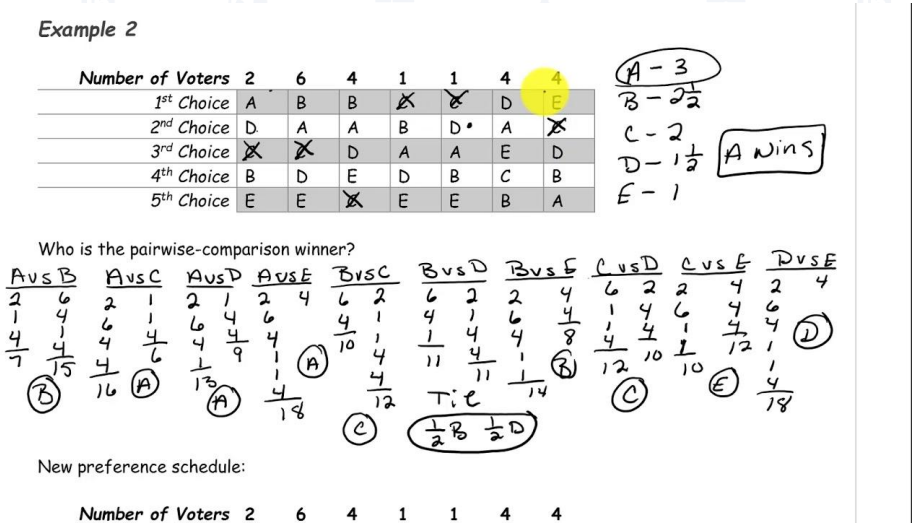
Ще один приклад, тільки за допомогою таблиці

Поточний стан	Подія	Дія	Наступний стан
Start	Enter card	Error(incorrect entered card)	Give back card
Start	Enter card	Correct card	Wait for pin
Wait for pin	Enter pin	Incorrect pin	Expectation for new pin(1 incorrect)
Wait for pin	Enter pin	Correct pin	Access to account
1 incorrect	Enter pin	Incorrect pin	Expectation for new pin(2 incorrect)
1 incorrect	Enter pin	Correct pin	Access to account
2 incorrect	Enter pin	Incorrect pin	Expectation for new pin(3 incorrect)
2 incorrect	Enter pin	Correct pin	Access to account
3 incorrect	Enter pin	Incorrect pin	Eat card
3 incorrect	Enter pin	Correct pin	Access to account

Pairwise testing

Pairwise testing

Pairwise testing - Розробка тестів методом чорної скриньки, в якій тестові сценарії розробляються таким чином, щоб виконати всі можливі окремі комбінації кожної пари вхідних властивостей.



Приклад

Після встановлення додатку необхідно встановити три параметри:

- операційна система (Mac, Linux або Windows),
- мова (German, Norwegian, English),
- розмір екрана (Small, Large).

Якщо для перевірки були обрані всі комбінаціїв результатів, вийде 3 × 3 × 2 = 18 тестів.

Однак після використання техніки в нас виходить 9 комбінацій.

Test case	OS	Language	Screen
1	Mac		
2	Mac		
3	Mac		
4	Linux		
5	Linux		
6	Linux		
7	Windows		
8	Windows		
9	Windows		

Test case	OS	Language	Screen
1	Mac	German	
2	Mac	Norwegian	
3	Mac	English	
4	Linux	German	
5	Linux	Norwegian	
6	Linux	English	
7	Windows	German	
8	Windows	Norwegian	
9	Windows	English	

Test case	OS	Language	Screen
1	Mac	German	Small
2	Mac	Norwegian	Large
3	Mac	English	Small
4	Linux	German	Large
5	Linux	Norwegian	Small
6	Linux	English	Large
7	Windows	German	Small
8	Windows	Norwegian	Large
9	Windows	English	Small

Use case testing

Use case testing

Use case testing - розробка тестів методом чорного ящика, при якому тестові сценарії створюються для виконання сценаріїв використання.

Use case - це перелік дій, сценарій, за яким користувач взаємодіє з додатком, програмою для виконання будь-якої дії для досягнення конкретної мети. Тестування за допомогою use case проводиться для того, щоб виявити додаткові логічні дірки та баги в додатку, які складно знайти в тестуванні індивідуальних модулів, частин програми окремо один від одного.

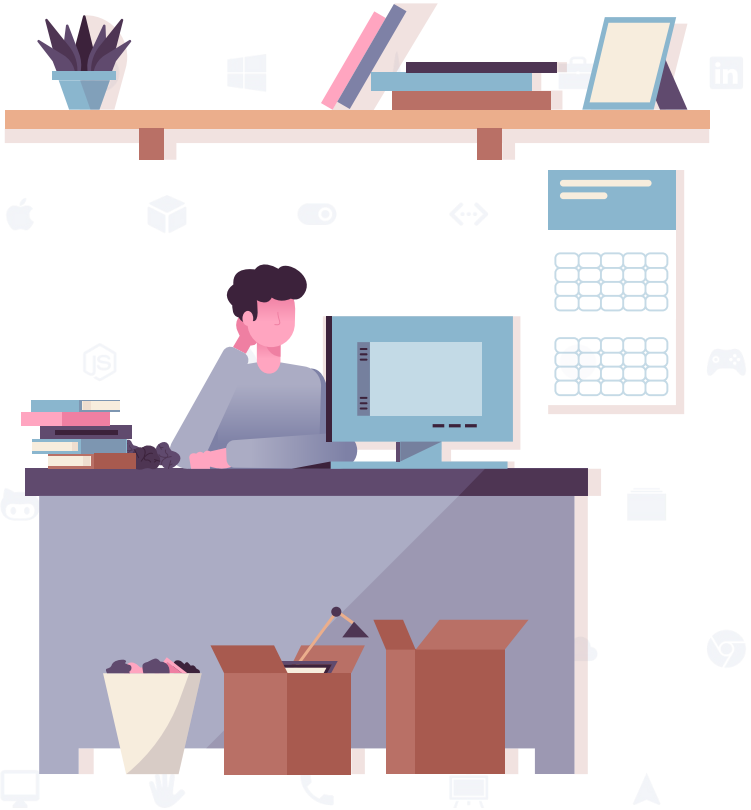
Приклад

	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

Experience based techniques

Experience based

Experience based technique - процедура отримання та/або вибору тестових випадків на основі досвіду, знань та інтуїції тестувальника



Error guessing

Error guessing - техніка проектування тестів, за якою досвід тестувальника використовується для передбачення того, які дефекти можуть бути присутніми в компоненті чи системі під час тестування в результаті допущених помилок і для розробки тестів спеціально для викриття їх.

- Техніка доповнюється формальними підходами.
- Успішність техніки дуже багато в чому залежить від майстерності тестера
- Тестування залежить від проблем, які в нас були до цього

Exploratory testing

Exploratory testing - у дослідницькому тестуванні розробляються, виконуються, реєструються та оцінюються неформальні (не визначені заздалегідь) тести динамічно під час виконання тесту. Результати тесту використовуються, щоб дізнатися більше про компонент або систему та створювати тести для областей, які можуть потребувати додаткового тестування.

.

Коли ми будемо використовувати exploratory testing ?

1. Вам потрібно забезпечити швидкий зворотний зв'язок про новий продукт або фічу
2. Вивчення продукту
3. Ви вже провели скриптове тестування і хочете урізноманітнити своє тестування
4. Вам потрібно знайти найважливіший баг у найкоротші терміни
5. Погана або відсутня документація

Вибір тестових технік

ANY QUESTIONS ?