Testing strategies
Стратегии тестирования

Specification-based or Black-box Techniques

Метод чёрного ящика (***blackbox|closedbox|specification-based testing***) — у тестировщика либо нет доступа к внутренней структуре и коду приложения, либо недостаточно знаний для их понимания, либо он сознательно не обращается к ним в процессе тестирования.

Тестировщик концентрируется на том, что делает программное обеспечение, а не как он это делает.

Structure-based or White-box Techniques

Метод белого ящика (***whitebox*|*openbox*|clear*box*|*glassbox testing***) — у тестировщика есть доступ к внутренней структуре и коду приложения, а также есть достаточно знаний для понимания увиденного. Некоторые авторы склонны жёстко связывать этот метод со статическим тестированием, но ничто не мешает тестировщику запустить код на выполнение и при этом периодически обращаться к самому коду.

Grey-box testing

Метод серого ящика (***graybox testing***) — комбинация методов белого ящика и чёрного ящика, состоящая в том, что к части кода и архитектуры у тестировщика доступ есть, а к части — нет. Обычно говорят о методах белого или чёрного ящика в применении к тем или иным частям приложения, при этом понимая, что «приложение целиком» тестируется по методу серого ящика.

White-box

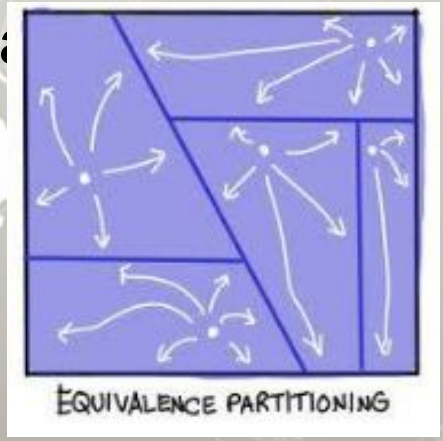
Преимущества	Недостатки
• Показывает скрытые проблемы и упрощает их диагностику. • Допускает достаточно простую автоматизацию тест-кейсов и их выполнение на самых ранних стадиях развития проекта. • Обладает развитой системой метрик, сбор и анализ которых легко автоматизируется. • Стимулирует разработчиков к написанию качественного кода. • Многие техники этого метода являются проверенными, хорошо себя зарекомендовавшими решениями, базирующимися на строгом техническом подходе.	• Не может выполняться тестировщиками, не обладающими достаточными знаниями в области программирования. • Тестирование сфокусировано на реализованной функциональности, что повышает вероятность пропуска нереализованных требований. • Поведение приложения исследуется в отрыве от реальной среды выполнения и не учитывает её влияние. • Поведение приложения исследуется в отрыве от реальных пользовательских

Black-box

Преимущества	Недостатки
• Тестировщик не обязан обладать (глубокими) знаниями в области программирования. • Поведение приложения исследуется в контексте реальной среды выполнения и учитывает её влияние. • Поведение приложения исследуется в контексте реальных пользовательских сценариев. • Тест-кейсы можно создавать уже на стадии появления стабильных требований. • Процесс создания тест-кейсов позволяет выявить дефекты в требованиях. • Допускает создание тест-кейсов, которые можно многократно использовать на разных проектах.	• Возможно повторение части тест-кейсов, уже выполненных разработчиками. • Высока вероятность того, что часть возможных вариантов поведения приложения останется непротестированной. • Для разработки высокоэффективных тест-кейсов необходима качественная документация. • Диагностика обнаруженных дефектов более сложна. • В связи с широким выбором техник и подходов затрудняется планирование и оценка трудозатрат. • В случае автоматизации могут потребоваться сложные дорогостоящие инструментальные средства.

Equivalence Partitioning

Эквивалентное разбиение (***equivalence partitioning***) - техника тестирования, направленная на сокращение количества разрабатываемых и выполняемых тест-кейсов при сохранении достаточного тестового покрытия. Основывается на предположении, что входы и выходы компонента могут быть разделены на классы, которые, в соответствии со спецификацией компонента, будут обрабатываться компонентом аналогично. Таким образом, результат тестирования одного значения из класса эквивалентности покрывает весь кл



Equivalence Partitioning

In a system designed to work out the tax to be paid:

An employee has £4000 of salary tax free.

The next £1500 is taxed at 10%.

The next £28000 after that is taxed at 22%.

Any further amount is taxed at 40%.

*To the nearest whole pound, which of these groups of numbers fall into three **DIFFERENT** equivalence classes?*

A £28000; £28001; £32001.

B £4000;£5000;£5500.

C £4000;£4200;£5600.

D £32001;£34000;£36500.

X≤ £4000 → 0%

£4000 < X ≤ £5500 → 10%

£5500 < X ≤ £33500 → 22%

X > £33500 → 40%

Equivalence Partitioning

*Given the following specification, which of the following values for age are in the **SAME** equivalence partition?*

If you are less than 18, you are too young to be insured.

Between 18 and 30 inclusive, you will receive a 20% discount.

Anyone over 30 is not eligible for a discount.

A 29, 30, 31

B 17,29, 31

C 18, 29, 30

D 17,18,19.

X< 18 → 0%

18 ≤ X ≤ 30 → 20%

X > 30 → 0%

Boundary Value Analysis

Тестирование на основе **граничных условий** (***boundary value analysis***) – это техника проверки ошибок на границах классов эквивалентности. Если техника анализа классов эквивалентности ориентирована на тестовое покрытие, то эта техника основана на рисках. Эта техника начинается с идеи о том, что программа может сломаться в области граничных значений.

Boundary Value Analysis

In a system designed to work out the tax to be paid:
An employee has £4000 of salary tax free.
The next £1500 is taxed at 10%.
The next £28000 after that is taxed at 22%.
Any further amount is taxed at 40%.

To the nearest whole pound, which of these is a valid Boundary Value Analysis test case?

A £32001
B £28000
C £1500
D £33501.

X≤ £4000 → 0%

£4000 < X ≤ £5500 → 10%

£5500 < X ≤ £33500 → 22%

X > £33500 → 40%