ДОДАТОК А

Графічний матеріал кваліфікаційної роботи

1

Міністерство освіти та науки України

Харківський національний університет радіоелектроніки

Кафедра ЕОМ

Магістерська кваліфікаційна робота

МЕТОДИ ТЕСТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ ПЛАТФОРМИ ANDROID

Виконав:
Ст.гр. КСМм-23-1
Семко В.В.

Керівник:
асистент кафедри
ЕОМ
Кравченко П.О.

Харків
2025

2

Мета роботи та завдання

Об'єктом дослідження є процес тестування програмного забезпечення для платформи Android.

Предметом дослідження є методи тестування програмного забезпечення.

Метою кваліфікаційної роботи є експериментальне дослідження існуючих методів тестування для платформи Android та їх порівняння.

Завдання:

- провести аналіз існуючої класифікації методів тестування ;
- обрати методи, які найбільш доцільно використовувати для тестування застосунків на платформі Android;
- реалізувати обрані методи тестування та провести порівняльний аналіз отриманих даних;
- провести аналіз отриманих результатів.

Дослідження існуючих методів тестування програмного забезпечення для платформи Android

3

Класифікація методів тестування програмного забезпечення зазвичай розглядається з двох точок зору: ручного та автоматизованого.

Ручне тестування включає в себе тестування білого ящика, чорного ящика та сірого ящика. У тестуванні білого ящика тестувальники мають доступ до внутрішньої структури Android-додатку, що дозволяє перевірити код, логіку та архітектуру програми. У тестуванні чорного ящика перевіряється функціональна поведінка додатку без знання його внутрішньої структури, а тестування сірого ящика комбінує обидва підходи, надаючи тестувальникам частковий доступ до внутрішньої логіки.

Автоматизоване тестування застосовується для зменшення ручної праці та збільшення ефективності тестів за рахунок використання спеціальних інструментів, які автоматично виконують тести.

Рисунок 1 – Класифікація методів тестування програмного забезпечення

Вибір методів тестування та інструментів розробки

4

Для досягнення поставлених задач було досліджено існуючі методи тестування програмного забезпечення та описано їх загальні особливості.

В результаті аналізу існуючих методів тестування було обрано три методи, які будуть досліджуватись:

- метод модульного тестування;
- метод інтеграційного тестування;
- метод функціонального тестування;

Для дослідження обраних методів тестування було розроблено програмний застосунок «калькулятор» з використанням мови програмування Python в середі розробки PyCharm, який дає змогу розробити тести згідно з обраними методами та перевірити їх ефективність.

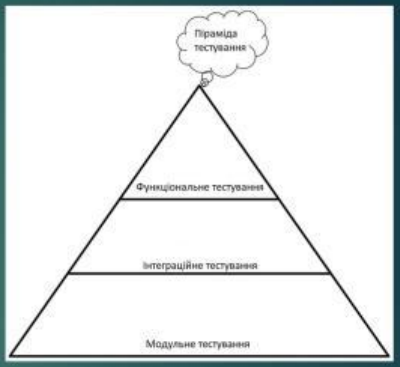


Рисунок 1 – Піраміда тестування

Метод модульного тестування

5

Метод модульного тестування (unit testing) являється основою будь-якого тестування програмного забезпечення, під час якого окремі модулі або компоненти програми перевіряються окремо від інших частин системи.

Основне завдання даного методу тестування – переконатися, що кожен окремий модуль працює правильно і відповідає заданим вимогам.

Основними характеристиками модульного тестування можна вважати наступне:

1) Ізоляція модулів;
2) Локалізація помилок;
3) Документування коду.

Рисунок 1 – послідовність кроків модульного тестування

Математична модель модульного тесту виглядатиме наступним чином:

$z = h(x)$, (1)

де z – результат виконання тесту;
h - функція тесту;
x – вхідні дані.

Метод інтеграційного тестування

6

Метод інтеграційного тестування (integration testing) - це підхід до тестування програмного забезпечення, при якому окремі модулі або компоненти, що пройшли модульне тестування, об'єднуються і тестуються як єдине ціле.

Мета інтеграційного тестування — переконатися, що ці модулі правильно взаємодіють між собою.

Основними характеристиками інтеграційного тестування можна вважати наступне:

1) Перевірка взаємодії;
2) Реальна інтеграція модулів;
3) Різні підходи до інтеграційного тестування.

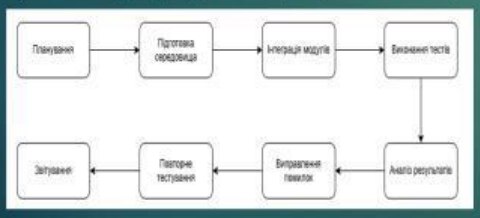


Рисунок 1 - Послідовність кроків інтеграційного тестування

Математична модель інтеграційного тесту виглядатиме наступним чином:

$M(P, C1) = ENij$, (2)

де E – множина переходів (дуг);
Nij – вхідні вершини графа модуля (позначають точки, з яких починається виконання тестів);
$M(P, C1)$ – математична модель тестування програми з урахуванням шляхів тестування P та критерію покриття гілок $C1$;

Метод функціонального тестування

7

Метод функціонального тестування (Functional Testing) — це підхід до тестування програмного забезпечення, який зосереджується на перевірці функціональних аспектів системи, тобто на тому, чи відповідає програмне забезпечення визначеним вимогам і специфікаціям.

Мета функціонального тестування полягає в тому, щоб переконатися, що кожна функція програми працює відповідно до очікувань.

Основними характеристиками функціонального тестування є:

1) Перевірка функціональності;
2) Метод чорного ящика;
3) Перевірка різних сценаріїв;
4) Тестування з точки зору користувача.

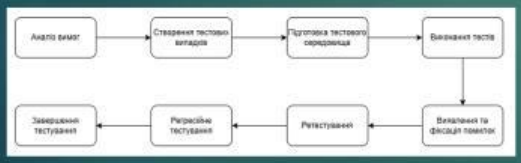


Рисунок 1 - Послідовність кроків функціонального тестування

Математичну модель функціонального тестування можна показати наступним чином:

$$f: T * X > Y, \qquad (2.8)$$

де f – функція, що описує поведінку системи під час тестування;

T – множина тестових сценаріїв для перевірки функціональності системи;

X – множина параметрів в системі;

Y – множина результатів, отриманих після тестування

Експериментальне дослідження методу модульного тестування

8

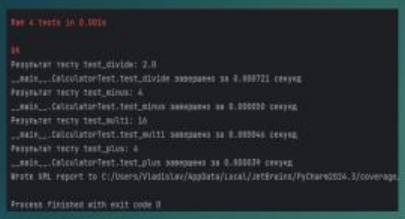


Рисунок 1 - Результат виконання модульних тестів

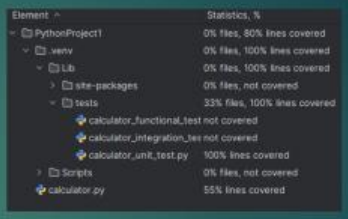


Рисунок 2 – Покриття коду модульними тестами

Загальний час виконання модульних тестів виглядає наступним чином:

- перевірка модуля ділення відбулася за 0.00071 секунду;
- перевірка модуля віднімання відбулася за 0.00005 секунд;
- перевірка модуля множення відбулася за 0.00004 секунди;
- перевірка модуля додавання відбулася за 0.00003 секунди;

Загальне покриття коду розробленого застосунку тестами становить 55%.

Експериментальне дослідження методу інтеграційного тестування

9

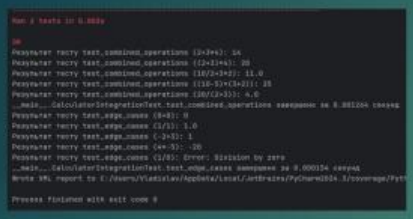


Рисунок 1 - Результат виконання інтеграційних тестів

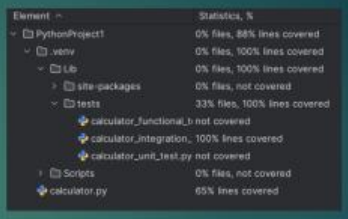


Рисунок 2 – Покриття коду інтеграційними тестами

Було створено два інтеграційні тести, час виконання яких виглядає наступним чином:
- інтеграційний тест з кількома операціями виконано за 0.00126 секунд;
- інтеграційний тест з крайовими випадками виконано за 0.00015 секунд.

Покриття коду розробленого застосунку зросло на 10% у порівнянні з модульними тестами, та становить 65%.

Експериментальне дослідження методу функціонального тестування

10

Рисунок 1 - Результат виконання інтеграційних тестів

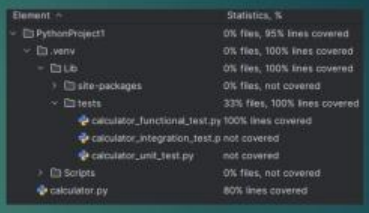


Рисунок 2 – Покриття коду інтеграційними тестами

Було створено 6 тестових сценаріїв, час виконання яких виглядає наступним чином:
- тестування коректності математичних виразів виконано за 0.00005 секунд;
- тестування ділення на нуль виконано за 0.00005 секунд;
- тестування крайових випадків, в тому числі і з нулем виконано за 0.00102 секунди;
- тестування на синтаксичні помилки виконано за 0.00016 секунд;
- тестування на ввод недопустимих символів виконано за 0.00006 секунд;
- тестування на взаємодію з порожніми виразами виконано за 0.00051 секунду;

Покриття коду тестами зросло до 80%, що дає найкращий результат в порівнянні з іншими

Порівняння досліджуваних методів тестування

11

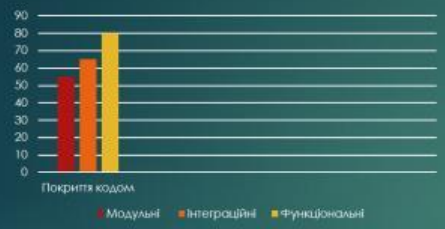


Рисунок 1 – Порівняння методів тестування за критерієм покриття коду

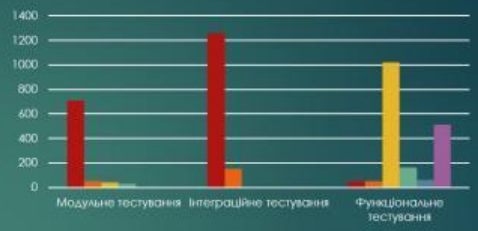


Рисунок 2 – Порівняння методів тестування за часом виконання

Висновки

12

У ході виконання кваліфікаційної роботи були досліджені методи тестування програмного забезпечення для платформи Android, такі як: метод модульного тестування, метод інтеграційного тестування та метод функціонального тестування.

У результаті проведеного дослідження було наведено порівняльний аналіз обраних методів тестування, що дає змогу побачити їх основні переваги та недоліки.

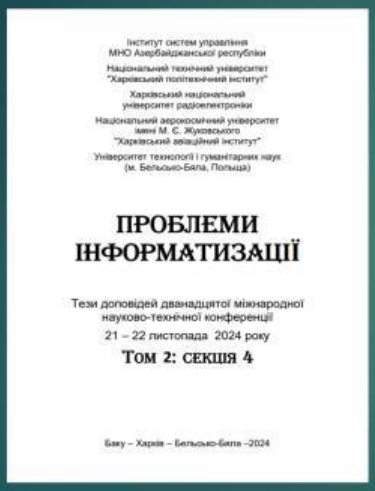
Метод модульного тестування показав найшвидший час виконання (від 0,00003 до 0,00071 секунд) та найменший рівень покриття коду тестами (55%). Даний метод являється найлегшим в реалізації, оскільки не вимагає від розробника знання внутрішньої структури програми, та стосується найменших одиниць коду.

Метод інтеграційного тестування показав велику розбіжність у часі виконання (від 0,00015 до 0,00126 секунд), проте, як правило, він буде другий за швидкодією після модульного тестування. Дана розбіжність під час досліду пояснюється тим, що в інтеграційному тестування в кожному тестовому кейсі знаходиться більший набір тестів, кожен з яких може викликати інші залежності, що в свою чергу впливає на час виконання. Покриття коду тестами зросло до 65%.

Метод функціонального тестування, як правило, найповільніший (від 0,00005 до 0,00102 секунд), оскільки має перевіряти функціональність системи в цілому, не звертаючи увагу на логіку програми чи внутрішній код. Даний мето має найбільше покриття коду тестами (80%), оскільки перевіряє всі очікувані функції в застосунку, проте найскладніший в реалізації.

Апробація результатів кваліфікаційної роботи

13



Інститут систем управління
МНО Азербайджанської республіки
Національний технічний університет
"Харківський політехнічний інститут"
Харківський національний
університет радіоелектроніки
Національний аерокосмічний університет
імені М. Є. Жуковського
"Харківський авіаційний інститут"
Університет технології і гуманітарних наук
(м. Бельсько-Біла, Польща)

ПРОБЛЕМИ ІНФОРМАТИЗАЦІЇ

Тези доповідей дванадцятої міжнародної
науково-технічної конференції
21 – 22 листопада 2024 року
Том 2: секція 4

Баку – Харків – Бельсько-Біла –2024

Проблеми інформатизації дванадцята міжнародна науково-технічна конференція

МЕТОДИ ТЕСТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ТА ЇХ АКТУАЛЬНІСТЬ

Сенко В.В., Кравченко П.О.
Харківський національний університет радіоелектроніки, Харків, Україна

Тестування програмного забезпечення – один з важливих етапів розробки, під час якого проводиться перевірка програмного забезпечення з метою виявлення помилок та дефектів. Воно дає можливість переконатися, що додаток працює правильно та відповідає вимогам користувачів [1].

Використання сучасних методів тестування може бути критично важливим під час розробки програмного забезпечення, оскільки дозволяє виявити більшість проблем ще на етапі розробки.

На сьогодні існує два основних підходи до тестування програмного застосунку: ручний або автоматизований [2]. Вони є прийнятною різними, кожен з яких має свої переваги та недоліки. Однак для досягнення високої якості програмного забезпечення важливо комбінувати ці підходи з використанням різних методів тестування, які охоплюють усі аспекти функціональності застосунку. До найпоширеніших методів тестування належать модульне тестування, інтеграційне тестування, системне тестування та тестування продуктивності. Комбінація різних методів тестування на кожному з етапів розробки забезпечує повноту перевірки та високу якість додатку.

Метою доповіді є дослідження сучасних методів тестування програмного забезпечення та їх актуальності в реаліях сьогодення. Розглянуто різні методи тестування, їх переваги та недоліки, а також сформовано рекомендації для ефективного використання під час розробки.

В доповіді наводяться основні відомості про існуючі методи тестування, їх класифікацію, а також дані про їх застосування розробниками та тестувальниками програмного забезпечення [3].

Наведені дані показують, що найбільш використовуваними методами є функціональне тестування, тестування інтерфейсу та регресійне. Водночас також зазначити, що на сьогоднішній день існує набір методів, які в більшості випадків вважаються неефективними та рідко використовуються. До них можна віднести аутсорсингове тестування та так званий метод мавпи-помічник.

Список літератури

1. Тестування програмного забезпечення: типи, види та застосування [Електронний ресурс]. – 2023. – Режим доступу до ресурсу: https://fozzinsoft.ua/ua/testuvannia-prohramnoho-zabezpechennia/.
2. Types of Software Testing [Електронний ресурс]. – 2024. – Режим доступу до ресурсу: https://www.guru4software.org/types-software-testing/.
3. What Software Test Approaches, Methods, and Techniques are Actually Used in Software Industry? [Електронний ресурс]/ Laura Strandfeat, Gunvi Arnkasen, Faris Bintiveri та ін.]. – 2018. – Режим доступу до ресурсу: https://nure.ua.org/Vol-2158/paper2.pdf.

Посилання на публікацію : https://nure.ua/wp-content/uploads/2024/pi_12_vol_2.pdf