Ф. Б. Гуральник
Р. Н. Квєтний

Тестова об'єктна модель: концепція, застосування та роль в автоматизації тестування

Вінницький національний технічний університет

У даній роботі досліджується концепція тестової об'єктної моделі (ТОМ) та її ключова роль в автоматизації процесів тестування програмного забезпечення. Тестова об'єктна модель є однією з найефективніших методологій, яка дозволяє спростити процес розробки та підтримки тестів завдяки модульності та повторному використанню коду. Робота детально аналізує основні принципи побудови ТОМ, звертаючи увагу на структурування тестів через абстракцію елементів користувацького інтерфейсу. Висвітлюються переваги, що випливають із застосування цієї моделі, зокрема, покращення читабельності коду, можливість швидкої адаптації до змін інтерфейсу та мінімізація ручної роботи при зміні тестових сценаріїв.
Окремий розділ присвячений практичним аспектам реалізації тестової об'єктної моделі. У цьому контексті досліджуються способи побудови каталогу класів, створення діаграм класів для опису тестових сутностей, а також наводяться приклади побудови похідних класів для забезпечення гнучкості в тестуванні. Робота пропонує ефективні підходи до абстрагування елементів інтерфейсу користувача, що дозволяє уникнути дублювання коду та зробити тести більш стійкими до змін в інтерфейсі програми.
Крім того, розглядаються можливості повторного використання існуючих тестових класів для підвищення ефективності роботи команди розробників та тестувальників. У роботі також досліджено вплив тестової об'єктної моделі на загальну структуру проєкту, її здатність робити проєкт більш масштабованим та прозорим для всіх членів команди. Використання ТОМ дозволяє оптимізувати процеси тестування, зменшуючи витрати часу на розробку та підтримку тестових сценаріїв. Це підтверджується на основі результатів дослідження, які свідчать про підвищення стабільності та якості автоматизованих тестів, що, у свою чергу, веде до загального покращення якості програмного забезпечення.

Ключові слова: автоматизація тестування, тестування пз, Page Object pattern, agile тестування

Вступ

Сучасна розробка програмного забезпечення вимагає високої якості та швидкості випуску продуктів. Автоматизація тестування стала ключовим елементом у досягненні цих цілей. Одним із підходів, що підвищують ефективність автоматизованого тестування, є використання тестової об'єктної моделі.

Тестова об'єктна модель (Test Object Model) дозволяє представити елементи користувацького інтерфейсу у вигляді об'єктів зі своїми властивостями та методами. Це створює абстрактний шар між тестовим кодом та реальними елементами інтерфейсу, що спрощує розробку, підтримку та масштабування тестів.

Результати дослідження

У ході дослідження були виявлені такі ключові аспекти тестової об'єктної моделі:

1. Абстрагування елементів інтерфейсу Представлення елементів як об'єктів дозволяє взаємодіяти з ними через зрозумілі методи та властивості, що підвищує читабельність коду.

2. Повторне використання коду: Об'єкти можуть бути використані в різних тестах, що зменшує дублювання коду та спрощує його підтримку.

3. Стійкість до змін: Зміни в інтерфейсі вимагають мінімальних коригувань у тестовому коді, оскільки взаємодія здійснюється через абстрактний шар.

4. Покращення структури проєкту: Використання об'єктної моделі сприяє кращій організації тестового проєкту, робить його більш масштабованим та зрозумілим.

Приклад використання тестової об'єктної моделі в автоматизованому тестуванні веб-додатків показав зменшення часу на розробку тестів на 30% та підвищення їхньої стабільності при оновленнях інтерфейсу.

Практичні приклади каталогу класів

Для кращого розуміння концепції тестової об'єктної моделі розглянемо практичний приклад каталогу класів для автоматизації тестування веб-додатку. На рисунку 1 зображено діаграму класів

Каталог класів:

1. BasePage: базовий клас для всіх сторінок, який містить спільні для всіх сторінок методи та властивості (наприклад, відкриття сторінки, очікування елементів)рис 2.
2. HomePage: клас, що представляє головну сторінку додатку. Успадковується від BasePage рис 3.
3. LoginPage: клас для сторінки авторизації. Містить методи для введення логіну, паролю та натискання кнопки входу. Успадковується від BasePage .
4. DashboardPage: клас для панелі користувача після входу в систему. Успадковується від BasePage .
5. UserProfilePage: клас для сторінки профілю користувача. Містить методи для редагування та збереження інформації. Успадковується від BasePage.

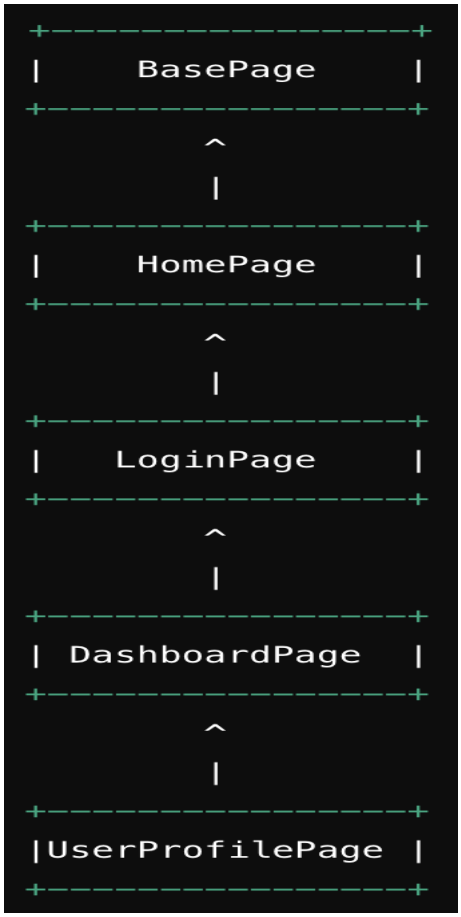

Рисунок 1 діаграма класів

Приклад створення похідних класів

Розглянемо реалізацію цих класів мовою Python з використанням бібліотеки Selenium для автоматизації веб-тестування.

```python
# base_page.py
from selenium.webdriver.support.ui import WebDriverWait

class BasePage:
    def __init__(self, driver):
        self.driver = driver
        self.wait = WebDriverWait(driver, 10)

    def open(self, url):
        self.driver.get(url)

    def find_element(self, locator):
        return self.wait.until(lambda driver: driver.find_element(*locator))
```

Рисунок 2 реалізація BasePage

```python
# home_page.py
from base_page import BasePage
from selenium.webdriver.common.by import By

class HomePage(BasePage):
    LOGIN_BUTTON = (By.ID, 'loginButton')

    def click_login(self):
        self.find_element(self.LOGIN_BUTTON).click()
```

Рисунок 3 приклад перевикористання методів класу в HomePage

Підсумок практичних прикладів

Додавання практичних прикладів каталогу класів та діаграми класів демонструє, як тестова об'єктна модель реалізується на практиці. Створення похідних класів від базового дозволяє:

1. Уніфікувати спільний функціонал: базовий клас містить загальні методи та властивості, що зменшує дублювання коду.
2. Покращити масштабованість: нові сторінки можуть бути легко додані шляхом створення нових класів, що успадковують базовий.
3. Забезпечити легкість підтримки: зміни в базовому класі автоматично застосовуються до всіх похідних класів.

Використання тестової об'єктної моделі з принципами об'єктно-орієнтованого програмування сприяє створенню ефективних, зрозумілих та легких у підтримці автоматизованих тестів.

Висновок:

Тестова об'єктна модель є важливим інструментом в автоматизації тестування, який сприяє підвищенню ефективності та якості процесу тестування. Її використання дозволяє створювати більш стійкі, зрозумілі та легкі у підтримці автоматизовані тести. Рекомендується впровадження тестової об'єктної моделі в практику автоматизованого тестування для досягнення кращих результатів у розробці програмного забезпечення.

Список використаної літератури

1. Фаулер М. Шаблони архітектури корпоративних застосунків. Київ: Діалектика, 2003.
2. Бек К. Розробка через тестування: Практичний посібник. Київ: Видавництво "Перо", 2004.
3. Кріспін Л., Грегорі Д. Гнучке тестування: Практичний посібник для тестувальників і гнучких команд. Харків: Видавництво "Фактор", 2011.
4. Паттон Р. Програмне забезпечення тестування: Практичний посібник* Київ: Видавництво "Діалектика", 2008.
5. Мартін Р. Чистий код: Створення, аналіз і рефакторинг. Київ: Видавництво "Оріон", 2019.
6. Річардсон А. Selenium Simplified: Посібник з автоматизації веб-тестування. London: Compendium Developments, 2012.
7. Месзарош Г. xUnit Test Patterns: Refactoring Test Code. Boston: Addison-Wesley, 2007.
8. Гомер С., Бассі Б., Брук П., Шу А. Автоматизація тестування програмного забезпечення: Методи, інструменти та підходи. Київ: Видавництво "Вільямс", 2011.

9. Ерік Ґамма, Річард Гелм, Ральф Джонсон, Джон Влісідес. **Design Patterns: Елементи багаторазового об'єктно-орієнтованого програмного забезпечення**. Київ: Видавництво "Діалектика", 2009.
10. Петренко О.В. **Основи автоматизації тестування програмного забезпечення**. Львів: Видавництво "Новий Світ", 2020.

Фредерік Борисович Гуральник - аспірант кафедри Автоматизації та інтелектуальних інформаційних технологій, e-mail: frederikguralnik@gmail.com;
Квєтний Роман Наумович —доктор технічних наук, професор кафедри Автоматизації та інтелектуальних інформаційних технологій, e-mail: rkvetny@vntu.edu.ua .

F. B. Huralnyk[1]
R. N. Kvyetnyy[2]

Test Object Model: Concept, Application, and Role in Test Automation

[1]Vinnytsia National Technical University

This research explores the concept of the Test Object Model and its role in software test automation. The This paper examines the concept of a test object model (TOM) and its key role in automating software testing processes. The test object model is one of the most effective methodologies that simplifies the development and maintenance of tests through modularity and code reuse. The paper provides a detailed analysis of the fundamental principles of TOM construction, focusing on the structuring of tests via abstraction of user interface elements. The advantages of using this model are highlighted, including improved code readability, the ability to quickly adapt to interface changes, and the minimization of manual work when test scenarios change.

A separate section is dedicated to the practical aspects of implementing a test object model. In this context, methods for building a class catalog, creating class diagrams to describe test entities, and examples of deriving classes to ensure flexibility in testing are explored. The paper presents effective approaches to abstracting user interface elements, which help avoid code duplication and make tests more resistant to changes in the application interface.

Additionally, the possibilities of reusing existing test classes to increase the efficiency of the development and testing teams are discussed. The study also investigates the impact of the test object model on the overall project structure and its ability to make the project more scalable and transparent for all team members. The use of TOM allows for the optimization of testing processes by reducing the time spent on developing and maintaining test scenarios. This is confirmed by the study results, which show an increase in the stability and quality of automated tests, ultimately leading to an overall improvement in software quality.

Keywords: test automation, software testing, Page Object pattern, agile testing

Frederik Borisovich Huralnyk - Postgraduate at the Department of Automation and Intelligent Information Technologies, email: frederikguralnik@gmail.com;
Roman Naumovych Kvyetnyy — Professor at the Department of Automation and Intelligent Information Technologies, e-mail: rkvetny@vntu.edu.ua .