УДК 004.6

Тимчук О. Г., *здобувач*
1 курсу ОС «Магістр»
спеціальності 122 Комп'ютерні науки,
Потапова Н. А., *канд. екон. наук,*
доцент кафедри
інформаційних технологій

МОДЕЛІ ТЕСТУВАННЯ ПРОГРАМНИХ ПРОДУКТІВ

Донецький національний університет імені Василя Стуса, м. Вінниця

Програмне забезпечення є невід'ємною частиною сучасного суспільства і пронизує кожен аспект нашого життя, від охорони здоров'я до комерції. Однак зі зростанням його важливості зростають і вимоги до якості та надійності. У цьому контексті тестування програмного забезпечення є важливим етапом розробки для виявлення та виправлення дефектів, задоволення потреб користувачів та забезпечення надійності продукту. Тестування програмного забезпечення – це складний, багатогранний процес, що включає в себе різні методи і моделі. Вибравши правильний підхід до тестування, розробники можуть забезпечити якість і продуктивність продукту. За останні кілька десятиліть як програмне забезпечення, так і методології тестування значно еволюціонували. Ми живемо в епоху постійних змін та інновацій, і тестування програмного забезпечення йде в ногу з цими змінами. Постійно вдосконалюючи методи тестування, ми можемо забезпечити високу якість і надійність програмних продуктів, які відповідають сучасним вимогам користувачів. Тому обґрунтованість питань щодо тестування програмних продуктів залишається завжди актуальною.

З урахуванням швидкого темпу змін у сфері IT та постійного вдосконалення програмних продуктів методи та моделі тестування також постійно еволюціонують. Оновлені підходи до тестування дають змогу ефективно впроваджувати нові функціональності та забезпечувати стабільну роботу програм. Існує чимало моделей тестування, але серед усіх виділяються такі:

- Модель водоспаду (Waterfall model).
- Модель V-форми (V-model).
- Інкрементна модель (Incremental model).
- Модель спіралі (Spiral model).

Модель водоспаду (Waterfall model). Її також називають лінійною послідовною моделлю життєвого циклу. У моделі водоспаду кожна фаза повністю завершується лише перед початком наступної фази. Така модель розробки програмного забезпечення використовується для невеликих проєктів без чітких визначених вимог. У кінці кожної фази проводяться тести, щоб визначити, чи відповідає проєкт поставленим вимогам. Тестування починається тільки тоді, коли завершується розробка [1].

Переваги водоспаду: Waterfall є доволі простим для розуміння і використання. У цій моделі фази виконуються послідовно і завершуються також послідов-

но. Фази не можуть повторюватися. Модель водоспаду найкраще підходить для невеликих проєктів.

Недоліки водоспаду: коли додаток знаходиться на етапі тестування, дуже важко повернутися назад і змінити речі, які не були повністю продумані на етапі розробки концепції; ризики через невизначеність високі; не підходить для складних або об'єктно-орієнтованих проєктів; не підходить для проєктів, де вимоги змінюються від середнього ризику до високого; не підходить для довгострокових, безперервних проєктів.

V-модель (V model) розшифровується як модель верифікації та валідації. Подібно до моделі водоспаду, життєвий цикл V-моделі – це послідовний шлях процесів. Кожна фаза повинна бути завершена до того, як почнеться наступна фаза. У цій моделі план тестування системи готується до початку розробки. План тестування спрямований на виконання функцій, визначених під час збору вимог [2].

Переваги V-моделі: вона легка і проста у використанні; економиться багато часу, оскільки планування, розробка тестів та інші дії з тестування відбуваються задовго до написання коду.

Недоліки V-моделі: програмне забезпечення розробляється на етапі реалізації, тому перший прототип програмного забезпечення не створюється.

Інкрементна модель (Incremental model). Ця модель використовує розбиття вимог на різні збірки, через що відбувається багато циклів розробки. Цикли поділяються на менші керовані модулі. У цій моделі кожен модуль послідовно проходить через фази вимог, проєктування, реалізації та тестування. Робоча версія програмного забезпечення створюється в першому модулі. Отже, з програмним забезпеченням можна працювати на ранній стадії життєвого циклу. У кожній наступній версії модуля додається функціональність до попередньої версії. Цей процес триває до тих пір, поки не буде завершена вся система [3].

Переваги інкрементної моделі: модель є дуже гнучкою, а вартість зміни обсягу та вимог є низькою.

Недоліки інкрементної моделі: вона вимагає відповідного планування проєктування, і вся система повинна бути чітко і повністю визначена до того, як система буде декомпонована і побудована поетапно.

Спіральна модель (Spiral model) подібна до інкрементної моделі, але з великим акцентом на аналіз ризиків. Модель поділяється на чотири фази: планування, аналіз ризиків, розробка та оцінка. Програмні проєкти проходять через ці фази ітеративно (в цій моделі вони називаються спіралями). Базова спіраль починається з фази планування, де збираються вимоги та оцінюються ризики. Кожна наступна спіраль ґрунтується на базовій спіралі [4].

Переваги спіральної моделі: ризиків можна уникнути завдяки колективному аналізу ризиків; найкраще підходить для великих і важливих проєктів; функціональність може бути додана не одразу; програмне забезпечення може бути створене на ранній стадії життєвого циклу програмного забезпечення.

Недоліки спіральної моделі: може бути дорогою у використанні; успіх проєкту значною мірою залежить від етапу аналізу ризиків; не підходить для малих проєктів.

Отже, методи та моделі тестування, як-от модель водоспаду, V-модель, інкрементна модель та спіральна модель, надають розробникам різні стратегії для забезпечення якості та надійності продукту.

Список використаних джерел

1. Devaraj K. Waterfall Model in Software Testing. 2024. URL: https://testsigma.com/blog/waterfall-model-in-software-testing/ (дата звернення: 02.05.2024).

2. Swain D. V Model in Software Testing | What, Why, Advantages and Disadvantages. 2024. URL: https://testsigma.com/blog/v-model-in-software-development-life-cycle/ (дата звернення: 02.05.2024).

3. Kumari R. Incremental Testing in Software Testing. 2024. URL: https://testsigma.com/blog/incremental-testing/ (дата звернення: 02.05.2024).

4. Saxena A. Spiral Model in Software Testing. 2023. URL: https://testsigma.com/blog/spiral-model/ (дата звернення: 02.05.2024).

УДК 004.774.6:004.92]: 7.012(494)

Бурківський О. С., *здобувач 2 курсу спеціальності 122 Комп'ютерні науки,*
науковий керівник:
Зелінська О. В., *канд. техн. наук, доцент, доцент кафедри інформаційних технологій*

ШВЕЙЦАРСЬКИЙ СТИЛЬ У ВЕБДИЗАЙНІ

Донецький національний університет імені Василя Стуса, м. Вінниця

Більшість людей обізнані з чудовим швейцарським стандартом. Ця традиція дизайну, заснована майже 100 років тому, залишається незмінною та актуальною і сьогодні. Швейцарський дизайн відмінно справляється з викликами сучасного бізнес-світу, втілюючи в собі якість, простоту й інновації.

Швейцарський дизайн відомий своєю найвищою точністю та простотою серед усіх стилів. Він підкреслює функціональність і практичність, віддаючи перевагу призначенню перед декором. У цьому стилі філософія полягає в тому, що дизайнери повинні бути справжніми майстрами своєї справи, підкреслюючи мистецтво ремесла над мистецтвом [1].

Цей стиль дизайну відрізняється своєю стриманістю і чіткістю в передачі ідеї. Він прагне до простоти, але водночас зберігає високу якість і естетичний стиль. Швейцарський дизайн вчить нас тому, як використовувати мінімалізм та чіткість для створення сильних і лаконічних зображень.

Цей стиль має свої переваги у сучасному світі дизайну:

– дає змогу зосередитися на суттєвих аспектах без зайвих деталей;

– забезпечує консистентність і виразність іміджу бренду;

– підвищує ефективність комунікації, створюючи чітке й зрозуміле візуальне враження.

Швейцарський дизайн – це не лише стиль, але й філософія, яка навчає нас привносити ясність та елегантність у кожен аспект створення.