УДК 004.9 : 004.4 : 004.054

А. А. Яровий, Я. В. Іванчук, В. С. Озеранський, В. О. Василевський

ОСОБЛИВОСТІ МОДЕЛЮВАННЯ МУЛЬТИКОМПОНЕНТНОЇ ІНФОРМАЦІЙНОЇ ТЕХНОЛОГІЇ АВТОМАТИЗОВАНОГО ТЕСТУВАННЯ

Вінницький національний технічний університет, Вінниця

Анотація. У роботі розглянуто проблематику етапу тестування в процесі життєвого циклу програмного забезпечення. Після проведеного аналізу встановлено актуальність використання систем автоматизованого тестування в процесі реалізації програмних продуктів. На базі цього, описано структуру інформаційної технології автоматизованого тестування та особливості реалізації компонентів програмного інтерфейсу та динамічних аналізаторів. Як результат, змодельовано мультикомпонентну інформаційну технологію автоматизованого тестування, що в максимальній мірі задовольняє потреби користувачів у якісному проведенні процесу модульного тестування.

Ключові слова: **інформаційна технологія, моделювання, автоматизоване тестування, модульне тестування, програмне забезпечення.**

Аннотация. В работе рассмотрена проблематика этапа тестирования в процессе жизненного цикла программного обеспечения. После проведенного анализа установлена актуальность использования систем автоматизированного тестирования в процессе реализации программных продуктов. На базе этого описана структура информационной технологии автоматизированого тестирования и особенности реализации компонентов программного интерфейса и динамических анализаторов. Как результат смоделирована мультикомпонентная информационная технология автоматизированного тестирования, которая в максимальной степени удовлетворяет потребности пользователей в качественном проведении процесса модульного тестирования.

Ключевые слова*:* информационная технология, моделирование, автоматизированое тестирование, модульное тестирование, програмное обеспечение.

Abstract. This work considers the problems of the software life cycle testing stage. After the analysis, the relevance to use automated testing systems in the implementation of software products was determined. Based on this, the structure of information technology of automated testing and features of the implementation of software interface components and dynamic analyzers are described. As a result, the multicomponent information technology of automated testing is modeled, which satisfies the needs of users in the high-quality implementation of the modular testing process.

Keywords: **information technology, modelling, automated testing, modular testing, software.**
DOI: https://doi.org/10.31649/1999-9941-2021-52-3-53-59.

Вступ

Незважаючи на покращення засобів реалізації програмного забезпечення, складність та комплексність кінцевих продуктів має неуклінну тенденцію збільшуватись щорічно. Відповідно ускладнюється процес підтримки, можливості внесення змін та тестування систем. Якість готового програмного зебезпечення значно залежить від якості методів та систем автоматичного тестування, що застосовувались на попередніх етапах. Коректне моделювання, вибір компонентів та визначення особливостей їх розробки є основою якісної реалізації інформаційної технології автоматизованого тестувананя.

Актуальність

Інститут системних наук в IBM встановив, що вартість виправлення критичних проблем значно збільшується із кожним переходом на нову стадію розробки [1]. Вартість виправлення помилки, виявленої після випуску продукту, була в чотири-п'ять разів більшою, ніж та, що виявлена під час етапу розробки, і до 100 разів більше, ніж одна, виявлена на етапі проєктування програмного забезпечення.

Також визначено, що лише для економіки США загальні перевитрати пов'язані із недосконалостями програмного забезпечення склали близько 60 мільярдів доларів [2]. Розробники, маючи за мету мінімізувати такі витрати, намагаються застосовувати різні види тестування. Залежно від системи, що розробляється використовуються комбінації різних форм ручного та автоматизованого тестування.

Натепер, одним з найпоширеніших підходів у автоматизованому тестуванні є модульний, що базується на ідеї написання невеликих тестових методів, котрі реалізують перевірку невеликих ізольованих елементів системи (окремих функцій) на правильність виконання. Цей метод дає можливість визначати значну кількість помилок на ранніх стадіях розробки, і надалі значно прискорити процес регресійного тестування при внесенні змін в існуючий код, оскільки великі обсяги програмного коду тестуються автоматично [3].

Для реалізації модулів автоматизованого тестування розробникам необхідно використати програмну систему (фреймворк, бібліотеку) модульного тестування. Вони різняться наявністю різноманітного функціоналу, методом організації та проведення тестування, тому важливо правильно вибрати ту технологію, котра найповніше виконає усі вимоги до процесу автоматичного тестування.

Сучасні технології модульного тестування містять ряд допоміжних функцій, що дозволяють їх користувачам краще застосовувати засоби автоматизованого тестування [4]. Проте, як правило системи аналізу та виконання бібліотек є незадовільними і не розкривають всю повноту статичного та динамічного опрацювання тестових модулів, а тому застосовуються для більш загальних випадків. Саме тому мо-

делювання інформаційної технології автоматизованого тестування з використанням засобів динамічного аналізу є актуальною задачею.

Мета

Метою дослідження є моделювання мультикомпонентної інформаційної технології автоматизованого тестування, розробка структури та виявлення особливостей реалізації її компонентів.

Задачі

1. Виконати моделювання структури мультикомпонентної інформаційної технології автоматизованого тестування.
2. Дослідити особливості реалізації компонентів інформаційної технології автоматизованого тестування.

Розв'язання задач

Моделювання структури мультикомпонентної інформаційної технології автоматизованого тестування

Інформаційна технологія автоматизованого тестування (ІТАТ) – сукупність програмних компонентів, що мають на меті забезпечити потребу користувачів у організації та проведенні процесу модульного тестування, шляхом надання відповідних технічних засобів.

Залежно від складності, цілей та особливостей реалізації технологій автоматизованого тестування, набір компонентів, що вони містять, може значно різнитись.

Основою ІТАТ є програмна бібліотека модульного тестування, котра містить повний набір функцій, атрибутів та інших елементів, представлених у вигляді вихідного коду, що доступні користувачеві та надають вичерпні можливості користування технологією. Цей компонент представлений у вигляді чітко визначеного та зрозумілого програмного інтерфейсу, знаючи який, користувач може реалізувати усі тестові модулі свого проєкту, написати власні необхідні тестові методи, а також програмно запустити виконання процесу автоматизованого тестування. Іншим функціоналом, що може програмна бібліотека є можливість розширення або зміни особливостей проведення процесу тестування, написання підготовчих або завершаючих задач, що виконуються відповідно до та після завершення проведення тестування, спеціалізоване логування та інші різноманітні елементи, що можуть бути необхідні користувачеві для забезпечення успішного виконання своїх задач.

Формат програмного інтерфейсу визначається розробниками на основі аналізу та особистих уподобань реалізації в рамках можливостей, що надають обрані мова та платформа програмування.

Наступним необхідним елементом ІТАТ є програмний двигун системи. Двигуном називають компонент технології, що в її архітектурі займає найнижчий рівень, не доступний користувачеві. На основі технічних специфікацій та реалізації програмного двигуна, будуються решта програмних засобів технології. Для інформаційної технології автоматизованого тестування двигун повинен містити реалізацію засобів пошуку тестових методів, опрацювання вхідних даних, виконання, налаштування системи, роботи з програмними потоками, їх одночасним виконанням (мультипоточністю), роботи з файловою системою, низькорівневе спеціалізоване логування, отримання результатів виконання тестів та проведення їх динамічного та статичного аналізу. Повний перелік задач, що реалізуються двигуном залежить від потреб технології, специфіки та обмежень обраної платформи програмування.

Саме двигун визначає загальний рівень швидкодії технології, оскільки він містить операції, що мають найвищий час виконання. Тому, від якості реалізації двигуна залежить можливість якісної імплементації компонентів вищого рівня та загальний рівень технології.

Наступним компонентом ІТАТ є користувацькі інтерфейси (КІ). В більшості випадків вони поділяються на консольні та графічні. КІ є необов'язковим елементом, їх наявність визначається розробниками на етапі проєктування та залежить від обраного формату розповсюдження та надання доступу до функціоналу технології [5].

Одним із важливих компонентів ІТАТ є програмне розширення для обраного інтегрованого середовища програмування, з допомогою якого користувачі технології можуть запускати процес модульного тестування, отримувати результати роботи та аналізу в зручному форматі у вигляді стандартних засобів підтримки виконання тестових модулів самого середовища. Цей компонент є одним з найзручніших з точки зору користувача, тому для інформаційної технології, що має за мету максимально задовольнити потреби тих, хто її використовує, наявність такого розширення є критично важливим.

Серед інших компонентів, котрі можуть містити сучасні ІТАТ, можна виділити [6]:

– Компонент, що містить набір аналізаторів, котрі виконують функцію статичного аналізу тестових методів. Вони можуть бути як представлені окремим елементом, так і інтегровані в двигун системи автоматизованого тестування, або в програмне розширення.

– База даних, що зберігає необхідну інформацію про тестові методи та результати їх виконання.

– Компонент, що виконує інтеграцію із різноманітними сторонніми сервісами, наприклад з сервісами безперервної інтеграції та доставлення TeamCity або Jenkins, веб-сервісами, що виконують запуск

процесу тестування з допомогою хмарних технологій та багато інших бібліотек та застосунків, що застосовуються в проєкті.

– Інші компоненти.

В процесі моделювання інформаційної технології автоматизованого тестування, необхідно визначити усі компоненти, що вона буде містити, а також системні вимоги до них.

Основним елементом розроблюваної ІТАТ є двигун системи. В якісній технології до нього висуваються вимоги високої швидкодії та можливості розширення за потреб користувача. Він повинен виконувати усі головні функції, визначені в процесі аналізу.

Розроблювана технологія повинна мати зрозумілий та зручний програмний інтерфейс бібліотеки з високим різноманіттям функцій перевірки та атрибутів тестування.

Наявність хоча б одного користувацького інтерфейсу є великою перевагою ІТАТ, оскільки не усі розробники мають доступ до програмних засобів проведення тестування. Він повинен містити зручний метод запуску процесу тестування та зрозумілий користувачеві інтерфейс, де легко та швидко можна визначити результати проведення тестування.

ІТАТ, що має на меті якнайповніше задоволення потреб користувачів у проведенні якісного модульного тестування, повинна містити компонент програмного розширення для найбільш популярного інтегрованого середовища програмування. Таке розширення повинне бути максимально інтегрованим у засоби середовища для його зручного використання.

Останнім компонентом, що містить змодельована ІТАТ є набір аналізаторів, що виконуватимуть функцію статичного аналізу тестових функцій, та допомагатимуть користувачам коректно та повноцінно використовувати засоби програмної бібліотеки.

На етапі моделювання технології, необхідно чітко визначити структурні та архітектурні залежності між її компонентами. Існує пряма залежність між кількістю елементів, що необхідно реалізувати та кінцевою складністю проєкту, адже в більшості випадків додавання лише одного компоненту значно впливає на усі ті, що вже були створені, якщо між ними є взаємозв’язки. На рисунку 1 представлена структурна схема компонентів технології, що містить усі елементи ІТАТ, які були визначені вище.

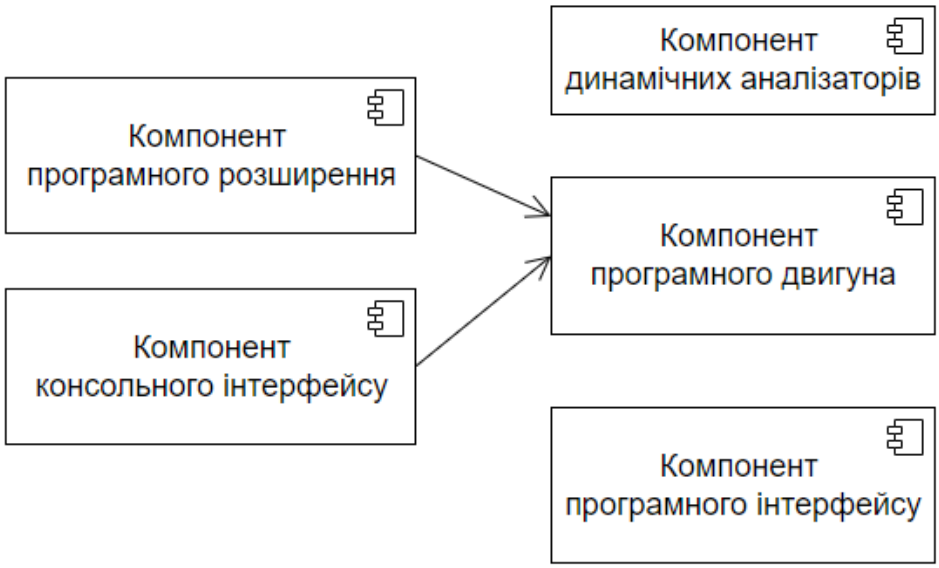

Рисунок 1 – Структурна схема компонентів змодельованої інформаційної технології автоматизованого тестування

У побудованій схемі кожен компонент є окремим структурним блоком, із вказанням його назви. Стрілками визначено залежності між модулями компонентів. Опишемо, які задачі виконує кожен окремий компонент технології:

1. Компонент програмного розширення містить програмні засоби необхідні розробнику для реалізації тестових методів згідно описаної технології.
2. Компонент консольного інтерфейсу містить вихідний код, результатом збірки якого є програмний продукт, котрий на основі вхідних даних виконує пошук, виконання та опрацювання результатів тестування та доступний користувачеві.
3. Компонент динамічних аналізаторів містить набір програмних аналізаторів, які після їх інтеграції з середовищем програмування виконують різноманітні перевірки реалізованих тестових методів на

відповідність їх коду до правил коректного використання програмного інтерфейсу, встановлених технологією.

4. Компонент програмного двигуна містить вихідний код, що відповідає за усі низькорівневі деталі реалізації пов’язані із пошуком, виконанням та опрацюванням результатів тестування.

5. Компонент програмного розширення містить реалізоване розширення, що при встановленні інтегрується із середовищем програмування та дозволяє використовувати стандартні засоби середовища для запуску та отримання результатів тестування технології.

Таким чином, використовуючи описану структуру компонентів, реалізована технологія, може найбільш ефективно задовільняти інформаційні потреби користувачів у проведенні якісного процесу автоматизованого тестування.

Дослідження особливостей реалізації компонентів інформаційної технології автоматизованого тестування.

В процесі моделювання важливо визначити стандартизований формат написання тестових методів. Формат написання тестових методів – узагальнений шаблон структури тестової функції, можливість реалізації якого повинен бути наданий програмними засобами технології.

Використання одного формату дозволяє розробникам легше розуміти структуру тестового методу, а відповідно полегшує імплементацію та внесення в нього змін. Саме тому, з метою покращення процесу проведення автоматизованого тестування програмний інтерфейс технології повинен надавати можливість використання оптимального та відомого підходу до написання тестових методів.

Натепер найвідомішим шаблоном написання модульних тестів є модель «AAA» («Arrange – Act – Assert») [7]. В її основі закладена ідея розділення тестового методу на три секції, кожна з яких виконує визначену функцію:

- В секції «Arrange» виконується підготовка об'єктів, що мають тестуватись. На цьому етапі необхідно привести інфраструктуру системи до бажаного стану та налаштувати залежності. Це виконується двома шляхами: прямим створенням екземпляру необхідного класу або створення його двійника, завчасно реалізованого для тестування.
- В секції «Act» відбуваються дії направлені на виклик відповідного методу системи, що тестується. Зазвичай, в цій частині шаблону викликається одна функція, котрій передаються налаштовані раніше залежності, та отримується результат її виконання для подальшої перевірки.
- Секція «Assert» містить перевірочні функції, що дають змогу визначати правильність виконання тестового методу на предмет відповідності очікуваному результату. На етапі може відбуватись перевірка результату виконання секції «Act», кінцевого стану системи, що підлягає тестуванню або інших елементів, що брали участь в процесі тестування.

На рисунку 2 наведено приклад застосування моделі «Arrange – Act – Assert» з використанням високорівневої мови програмування C#.

```
[Test]
0 references
public static void ContainsInt()
{
    // arrange
    var list = new List<int>(5);
    var expected = 3;

    // act
    list.Add(expected);

    // assert
    Assert.Contains(expected, list);
}
```

Рисунок 2 – Приклад використання моделі написання тестових методів «Arrange – Act – Assert»

Серед інших особливостей застосування моделі «AAA» варто виділити:

- Уникнення використання декількох однакових секцій в рамках одного тестового методу, що свідчить про те, що функція виконує перевірку декількох різних поведінкових випадків. В цьому випадку варто розділити цей метод на кілька таких, кожен з яких перевіряє окремий випадок. Приклад структури тестового методу, що відповідає цій неправильній особливості реалізації наведено на рисунку 3.

- Уникання використання умовних тверджень в тестових методах. Для правильної та однозначної перевірки очікуваного результату, тестовий метод повинен бути простою послідовністю чітко визначених кроків перевірки. Наявність розгалужень свідчить, що метод виконує занадто багато перевірок. Використання умовних тверджень не дає жодних переваг, а лише додаткові витрати на підтримку тестових модулів, оскільки такі функції важче розуміти та модифікувати.
- Секція «Act» зазвичай складається з одного рядка коду. Якщо етап складається з двох або більше рядків, це може свідчити про проблему з публічним інтерфейсом тестування системи, що тестується.
- Для кращого розуміння тестового методу, кожну секцію варто відділяти між собою пустою стрічкою, та на її початку використовувати коментарі з назвою секції, що послідує далі.

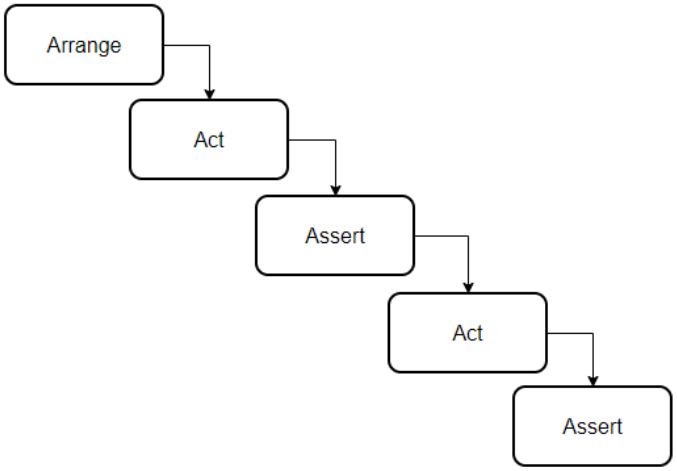

Рисунок 3 – Варіант використання декількох однакових секцій в одному тестовому методі

Визначено, що для покращення процесу проведення автоматизованого тестування компонент програмного інтерфейсу інформаційної технології повинен надавати можливість та встановлювати правило використання шаблону «Arrange – Act – Assert» при написанні тестових методів користувачем.

Компонент інформаційної технології, що містить аналізатори є одним з найголовніших для покращення процесу автоматизованого тестування, шляхом значного збільшення коректності використання програмної бібліотеки модульного тестування.

Динамічним аналізатором є програмний елемент, що в режимі реального часу аналізує новий, або щойно змінений програмний код на предмет невідповідності правилам описаним у ньому. Такий аналізатор зазвичай містить назву, повний опис знайденої проблеми та рекомендацію, щодо її усунення. В деяких випадках такий аналізатор містить варіанти виправлення, при виборі яких, код автоматично змінюється, відповідно запропонованій корекції.

Реалізація цього компоненту значно різниться залежно від обраного інтегрованого середовища, платформи та мови програмування. Ця залежність є вагомою, оскільки не всі інтегровані середовища програмування підтримують можливість відображення результатів роботи аналізаторів в реальному часі. Тому на це варто зважати на етапі реалізації розроблюваної технології.

Різні платформи та мови програмування надають різні можливості реалізації аналізаторів [8]. Наприклад компанія Microsoft, розробник платформи .NET та мов програмування C#, F# та інших, з допомогою спеціальних програмних бібліотек надає широкий доступ до потужностей своєї платформи, таким чином дозволяючи отримувати та аналізувати широкий спектр перевірок на глибокому рівні аналізу програмного коду. Значною перевагою також є реалізація прямої інтеграції інтерфейсу отримання результатів аналізаторів в найпопулярніше інтегроване середовище програмування платформи .NET – Visual Studio. Приклад такого відображення результатів показано на рис. 4. Компанія Microsoft не єдина, що пропонує власні програмні інтерфейси реалізації аналізаторів. Варто виділити популярні середовища програмування IntelijIDEA, WebStorm та PyCharm, від компанії JetBrains, що мають можливість впровадження аналізаторів для мов програмування Java, JavaScript та Python та ряд інших. Проте, рівень можливостей при реалізації цих аналізаторів та прямої інтеграції з середовищем не є оптимальним для їх широкого використання.

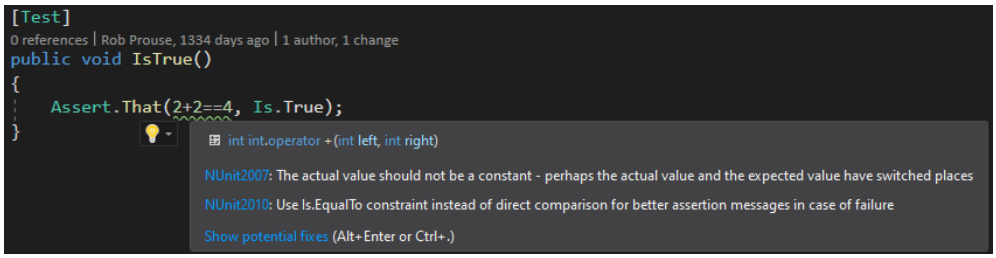

Рисунок 4 – Приклад візуального відображення результатів динамічного аналізу в середовищі програмування Visual Studio з використанням мови програмування C#

На етапі моделювання компоненту варто визначити типи аналізаторів, що будуть реалізовуватись. Це допоможе ширше покрити випадки некоректного використання бібліотеки, а отже збільшити якість перевірок.

Першим типом є структурні аналізатори. Їх правила відповідають за структурну цілісність тестових методів. Цей тип може містити перевірки використання асинхронних блоків коду, передачі коректних типів та кількості аргументів у тестовий метод, взаємодії з інтерфейсом підготовки та завершення виконання тестової функції, та інший аналіз.

Другим типом, що зазвичай являється найкількіснішим, є стверджувальні. Вони містять правила, які покращують використання тверджень в тестовому коді. До них відносяться правила перевірок неправильно використаних функцій ствердження, що надаються програмним інтерфейсом бібліотеки модульного тестування.

Третім, можливим типом аналізаторів є приховуючі. Вони містять ряд правил, котрі приховують помилки компілятора мови програмування на основі контексту, та відображають свої припущення та рекомендації, щодо виправлення цих помилок.

Для значного покращення процесу автоматизованого тестування, інформаційна технологія повинна містити компонент аналізаторів. Описані правила, що опрацьовуються аналізаторами мають містити максимально повну інформацію про проблему та, за можливості, надавати варіанти їх вирішення. Також, важливим фактором реалізації цього компоненту є його можливість прямої інтеграції із відомими інтегрованими середовищами програмування для полегшення отримання візуальних результатів аналізаторів.

Висновки

Встановлено, що розробка інформаційної технології автоматизованого тестування є доцільною та актуальною задачею.

Визначено оптимальну структуру компонентів інформаційної технології, що дозволяє найбільш ефективно виконувати задачі проведення та отримання результатів процесу автоматизованого тестування. Сформовано структурну схему інформаційної технології автоматизованого тестування.

Проаналізованого головні особливості реалізації компонентів інформаційної технології. Встановлено оптимальність застосування моделі «Arrange-Act-Assert» компонентом програмного інтерфейсу. Описані типи правил, що опрацьовуються аналізаторами та деталі їх інформаційного наповнення.

В подальших дослідженнях планується дослідити особливості реалізації компонентів інформаційної технології автоматизованого тестування, а також визначити оптимальний набір програмних засобів і технологій для розробки.

Список літератури

[1] The True Cost of a Software Bug: Part One. [Online]. Available: https://www.celerity.com/the-truecost-of-a-software-bug Accessed on: Oct. 30, 2021.

[2] Software Bugs Cost U.S. Economy $59.6 Billion Annually, RTI Study Finds, 2021. [Online]. Available: http://www.nist.gov/director/prog-ofc/report02-3.pdf. Accessed on: Oct. 30, 2021.

[3] В. О. Василевський, А. А. Яровий, «Створення інтелектуальної системи автоматизованого тестування на основі фреймворку юніт-тестування», *Матеріали L науково-технічної конференції факультету інформаційних технологій та комп'ютерної інженерії*, с. 627 – 629. 2021. [Електронний ресурс]. Режим доступу: https://conferences.vntu.edu.ua/public/files/1/vntu_2021_netpub.pdf. Дата звернення: Жовт. 30, 2021.

[4] V. O. Vasylevskyi, A. A. Yarovyi, «Features of providing access to the software tools of the automated testing information system», *Матеріали V міжнародної накового-практичної конференції «Modern directions of scientific research development»*, с. 230 – 234. 2021. [Електронний ресурс]. Режим доступу: https://sci-conf.com.ua/wp-content/uploads/2021/11/MODERN-DIRECTIONS-OF-SCIENTIFIC-RESEARCH-DEVELOPMENT-28-30.10.21.pdf. Дата звернення: Жовт. 30, 2021.

[5] Ошеров Р. The art of unit testing with examples in .NET – Гринвіч: Manning Publication Co, 2009,

с. 20.
[6] С. Фрімен, Growing Object-Oriented Software Guided by Tests – Crawfordsville, Indiana: Donnelley, 2012, с. 28-29.
[7] А. С. Авраменко, В. С. Авраменко, Г. В. Косенюк, *Тестування програмного забезпечення. Навчальний посібник*. Черкаси: ЧНУ імені Богдана Хмельницького, 2017, 284 с.
[8] Г. Майерс, Т. Баджетт, К. Сандлер, *Искусство тестирования программ, 3-е издание*. [Електронний ресурс]. Режим доступу: http://www.dialektika.com/books/978-5-8459-1796-6.html. Дата звернення: Жовт. 30, 2021.

Стаття надійшла: 12.11.2021.

References

[1] The True Cost of a Software Bug: Part One. [Online]. Available: https://www.celerity.com/the-truecost-of-a-software-bug Accessed on: Oct. 30, 2021.
[2] Software Bugs Cost U.S. Economy $59.6 Billion Annually, RTI Study Finds, 2021. [Online]. Available: http://www.nist.gov/director/prog-ofc/report02-3.pdf. Accessed on: Oct. 30, 2021.
[3] V. O. Vasylevskyi, A. A. Yarovyi, «Stvorenya intelektualnoi systemy avtomatyzovanoho testuvannya na osnovi framework unit-testuvannya», *Materialy L naukovo-tekhnichnoi konferentsii fakultetu informatsiinykh tekhnolohii ta kompiuternoi inzhenerii*, s. 627 – 629. 2021. [Elektronnyi resurs]. Rezhym dostupu: https://conferences.vntu.edu.ua/public/files/1/vntu_2021_netpub.pdf. Data zvernennia: Zhovt. 30, 2021 [in Ukrainian].
[4] V. O. Vasylevskyi, A. A. Yarovyi, «Features of providing access to the software tools of the automated testing information system», *Materialy V mizhnarodnoi naukovo-praktychnoi konferentsii «Modern directions of scientific research development»*, s. 230–234. 2021. [Elektronnyi resurs]. Rezhym dostupu: https://sci-conf.com.ua/wp-content/uploads/2021/11/MODERN-DIRECTIONS-OF-SCIENTIFIC-RESEARCH-DEVELOPMENT-28-30.10.21.pdf. Data zvernennia: Zhovt. 30, 2021 [in Ukrainian].
[5] Osherov R. The art of unit testing with examples in .NET – Grinvich: Manning Publication Co, 2009, s. 20.
[6] S. Freeman, Growing Object-Oriented Software Guided by Tests – Crawfordsville, Indiana: Donnelley, 2012, s. 28-29.
[7] S. Avramenko, V. S. Avramenko, G. V. Kosenyuk, *Testuvannya programnogo zabespechennia. Navchalnyi posibnyk*. Chernkasy: CNU imeni Bogdana Khmelnytskogo, 2017, s. 284 [in Ukrainian].
[8] G. Mayers, T. Badgett, C. Sandler, The art of software testing, 3-rd edition. [Elektronnyi resurs]. Rezhym dostupu: http://www.dialektika.com/books/978-5-8459-1796-6.html. Data zvernennia: Zhovt. 30, 2021.

Відомості про авторів

Яровий Андрій Анатолійович − доктор технічних наук, професор, завідувач кафедри комп'ютерних наук.

Іванчук Ярослав Володимирович − доктор технічних наук, доцент, професор кафедри комп'ютерних наук.

Озеранський Володимир Сергійович – кандидат технічних наук, старший викладач кафедри комп'ютерних наук.

Василевський Володимир Олегович − студент групи 2КН-20м, факультет інформаційних технологій та комп'ютерної інженерії.

А. А. Яровой, Я. В. Иванчук, В. С. Озеранский, В. О. Василевский

ОСОБЕННОСТИ МОДЕЛИРОВАНИЯ МУЛЬТИКОМПОНЕНТНОЙ ИНФОРМАЦИОННОЙ ТЕХНОЛОГИИ АВТОМАТИЗИРОВАНОГО ТЕСТИРОВАНИЯ

Винницкий национальный технический университет, Винница

A.Yarovyi, Ya. Ivanchuk, V. Ozeranskyi, V. Vasylevskyi

FEATURES OF MODELING OF MULTICOMPONENT INFORMATION TECHNOLOGY OF AUTOMATED TESTING

Vinnytsia National Technical University, Vinnytsia