МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЗАХІДНОУКРАЇНСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

БАЗАКА ЮРІЙ АНАТОЛІЙОВИЧ

УДК 004.05

МОДЕЛІ, МЕТОДИ ТА ІНФОРМАЦІЙНА ТЕХНОЛОГІЯ ПІДВИЩЕННЯ ЕФЕКТИВНОСТІ ТЕСТУВАННЯ РОЗПОДІЛЕНИХ СИСТЕМ

Спеціальність 05.13.06 «Інформаційні технології»

АВТОРЕФЕРАТ
дисертації на здобуття наукового ступеня
кандидата технічних наук

Тернопіль – 2021

Дисертацією є рукопис.

Робота виконана в Національному технічному університеті України «Київський політехнічний інститут імені Ігоря Сікорського» Міністерства освіти і науки України.

Науковий керівник: доктор технічних наук, доцент
Корнага Ярослав Ігорович
Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського», доцент кафедри технічної кібернетики.

Офіційні опоненти: доктор технічних наук, професор
Бармак Олександр Володимирович
Хмельницький національний університет, завідувач кафедри комп'ютерних наук та інформаційних технологій;

доктор технічних наук, професор
Шаховська Наталія Богданівна
Національний університет «Львівська політехніка», завідувач кафедри систем штучного інтелекту.

Захист відбудеться «05» травня 2021 року о 15 годині на засіданні спеціалізованої вченої ради К 58.082.02 у Західноукраїнському національному університеті за адресою: 46009, м. Тернопіль, вул. Львівська, 11а, зал засідань.

З дисертацією можна ознайомитись у бібліотеці Західноукраїнського національного університету за адресою: 46009, м. Тернопіль, вул. Бережанська, 4.

Автореферат розісланий «02» квітня 2021 року.

Учений секретар
спеціалізованої вченої ради
кандидат технічних наук, доцент М.П. Комар

ЗАГАЛЬНА ХАРАКТЕРИСТИКА РОБОТИ

Актуальність теми. Сучасний рівень організації та управління розподіленими інформаційними системами передбачає використання різних механізмів оброблення та зберігання даних. Одним із них є забезпечення користувачів розподілених систем можливістю необмеженого доступу до масивів даних за умови високої швидкості оброблення даних у системах з постійним масштабуванням обсягу оброблюваної інформації. Проведенням перевірки на працездатність таких систем займаються багато програмістів.

У наш час розподілені системи розвиваються надзвичайно швидко завдяки використанню мікросервісних архітектур та необмеженого доступу до ресурсів мережі інтернет. За останні три роки об'єми інформації у глобальних мережах збільшилися в кільканадцять разів та становлять понад 40 трильйонів гігабайт. Перші дослідження у сфері тестування розподілених систем обробки інформації розпочалися у кінці 1990-х рр., після того як глобальна мережа стала доступною більшості людей. Основним завданням досліджень стали можливості автоматизованого тестування та перевірка розподілених систем, що характеризуються обмеженням часу доступу та кількістю ресурсів. У свою чергу, це стало передумовою появи великої кількості можливостей застосування нових засобів та механізмів у практичному використанні.

На сьогодні тестування розподілених систем обробки інформації вважають одним із найбільш важливих завдань під час розроблення будь-якого програмного забезпечення, а з появою різних технологій побудови розподілених систем методи тестування дозволяють перевіряти великі масиви інформації за відповідних заданих параметрів на вузли системи. Ці параметри є важливими для впровадження автоматизованого тестування й досягнення мети підвищення швидкості та якості роботи програмних продуктів.

Питанням побудови механізмів тестування розподілених систем обробки інформації присвячено велику кількість наукових робіт українських та зарубіжних вчених В.Є. Мухіна, Ю.В. Кравченка, Г.А. Кучука, І.Ю. Субача, а також F. Eliassen, A. Polini, J. Wildstrom, P. Stone, E. Witchel, R. Mooney, M. Dahlin та ін. Питання забезпечення надійності роботи розподілених систем досліджено у роботах О.Г. Додонова, Д.В. Ланде, Ю.В. Журавського, І.В. Рубана.

Незважаючи на значні успіхи у вирішенні проблем тестування розподілених систем обробки інформації, далеко не всі завдання у цій сфері можна вважати вирішеними. Актуальними дослідженнями є системи з динамічною структурою середовища зберігання даних в умовах впливу внутрішніх і зовнішніх дестабілізуючих факторів. Виходячи з цього, недосконаліть та обмеженість відомих наукових методів тестування розподілених систем обробки інформації, зокрема у бездротових мобільних мережах, не дозволяє забезпечити повноцінне виявлення несправностей та своєчасне їх усунення.

Таким чином, дослідження моделей, методів та інформаційних технологій підтримки ефективного тестування програмного забезпечення на сьогодні є актуальним науковим завданням.

Зв'язок роботи з науковими програмами, планами та темами. Тематика дисертаційної роботи й отримані результати безпосередньо відповідають пріоритетності розвитку інформаційних та комунікаційних технологій в Україні до 2020 р. згідно із Законом України «Про пріоритетні напрями розвитку науки і техніки» від 11.07.2001 р., № 2623-III, зі змінами, внесеними згідно із Законом України «Про наукову та науково-технічну діяльність» від 26.11.2015 р., № 848-VIII.

Дисертаційна робота виконана відповідно до планів наукової і науково-технічної діяльності Національного технічного університету України «Київський політехнічний інститут імені Ігоря Сікорського» і є частиною досліджень у межах науково-дослідних робіт:

– «Антропоморфний роботизований транспортний засіб для розвантаження людини в умовах підвищеного ризику та невизначеності рельєфу місцевості» (Державний реєстраційний № 0117U001179, КПІ ім. Ігоря Сікорського, м. Київ), яку виконував Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського» у 2017–2018 рр.;

– «Оптимізація роботи веб-орієнтованих систем з великим набором даних» (державний реєстраційний № 0117U004913, КПІ ім. Ігоря Сікорського, м. Київ), яку виконує кафедра технічної кібернетики Національного технічного університету України «Київський політехнічний інститут імені Ігоря Сікорського» з 2018 р.

Особисто автором у першій науково-дослідній роботі запропоновано механізми тестування вузлів зв'язку з антропоморфним роботизованим транспортним засобом, у другій – методи автоматизованого тестування інформації великого обсягу на основі модифікації механізмів обміну даними.

Мета і задачі дослідження. Метою дисертаційної роботи є підвищення ефективності функціонування розподілених систем обробки інформації на основі розроблення моделей і методів тестування компонентів таких систем.

Для досягнення поставленої мети в роботі вирішуються такі **основні завдання**:

1. Аналіз проблеми використання наявних підходів до тестування в архітектурі розподілених систем.
2. Розроблення модифікованої «піраміди» тестування програмного забезпечення в архітектурі систем розподіленої обробки інформації.
3. Розроблення моделі тестування сервісів розподіленої системи обробки інформації.
4. Удосконалення методу тестування програмного забезпечення розподілених систем з використання імітаторів.
5. Удосконалення методу тестування інтерфейсів користувача розподіленої системи з використанням симуляційних тестів.

6. Розроблення інформаційної технології для підвищення ефективності тестування програмного забезпечення розподілених систем.

7. Розроблення спеціалізованого середовища для моделювання та порівняння результатів експериментальних досліджень запропонованих методів тестування розподілених систем.

Об'єкт дослідження – процеси тестування систем розподіленої обробки інформації.

Предмет дослідження – моделі, методи та інформаційна технологія для ефективного тестування розподіленого програмного забезпечення.

Методи дослідження. Дисертаційне дослідження ґрунтується на системному аналізі результатів сучасних теоретичних і прикладних розробок вітчизняних і зарубіжних вчених у сфері аналізу програмного коду. Під час виконання дослідження було використано логіко-імовірнісні та статистичні методи, методи теорії обробки спостережень для аналізу експериментальних даних, методи формалізації даних, методи аналітичного моделювання.

Наукова новизна отриманих результатів:

1. Уперше розроблено модель тестування програмного забезпечення розподілених систем на основі автономного розгортання програмних компонентів, яка ґрунтується на застосуванні механізмів на основі модифікованої піраміди Майка Кона, що дозволяє підвищити ефективність тестування та зменшити кількість об'ємних тестів.

2. Удосконалено метод тестування програмного забезпечення вузлів розподіленої системи, який різниться від наявних застосуванням контрактних тестів та аналізом прогнозованого результату поведінки сервісів, що дозволяє зменшити час розгортання компонентів тестового середовища та час проходження тестів.

3. Удосконалено метод тестування інтерфейсу користувача програмного забезпечення вузлів розподіленої системи, який різниться від наявних підходом до симуляції його роботи, що дозволяє проводити тестування окремих компонентів інтерфейсу системи.

4. Набула подальшого розвитку інформаційна технологія, яка ґрунтується на комплексному застосуванні запропонованих моделі та методів, що дозволяє підвищити ефективність тестування за рахунок прискорення процесу тестування програмного забезпечення розподілених систем.

Практичне значення одержаних результатів. Створена за запропонованою інформаційною технологією інформаційна система, що реалізує представлені в роботі моделі та методи, може бути використана для вирішення завдань забезпечення тестування розподілених систем обробки інформації. Проведені експериментальні дослідження тестування інформаційної системи підтверджують можливість її застосування для розподілених систем обробки інформації. Результати виконаних розрахунків та оцінок можуть стати основою для підготовки рішень з управління розробкою програмного забезпечення.

Практичне значення результатів роботи підтверджено актами впровадження інформаційної системи ТОВ «Нафтогазбудінформатика» та у конструкторському бюро інформаційних систем КПІ ім. Ігоря Сікорського, довідкою про застосування результатів роботи у навчальному процесі кафедри технічної кібернетики Національного технічного університету України «Київський політехнічний інститут імені Ігоря Сікорського».

Особистий внесок здобувача. Усі наукові результати дисертації одержано автором самостійно. У друкованих працях, опублікованих у співавторстві, йому належить таке: аналіз методів моніторингу роботи гетерогенних розподілених баз даних [1]; обґрунтування вимог до вибору параметрів, за якими тестують комп'ютерні компоненти [2]; метод тестування з використанням контрактів [3]; метод тестування інтерфейсів з використанням симуляторів [4]; дослідження питань побудови моделі захищеного коду з використанням віртуальних машин [5]; тестування обфускації програмного коду [6]; дослідження використання в мовах програмування високоточних обчислень [7]; модель тестування розподілених систем на основі служб управління [8]; підхід до методу тестування програмного коду після обфускації вставкою інструкцій [9]; розроблення методу тестування програмного забезпечення розподілених комп'ютерних систем [10]; побудова спрощеної математичної моделі тестування ресурсів розподіленої системи [11]; аналіз використання мережоцентричного підходу в розподіленій комп'ютерній системі [12]; тестування бездротової комп'ютерної системи з мережоцентричним управлінням [13].

Апробація результатів дисертації. Основні результати дисертаційних досліджень доповідалися й обговорювалися на таких конференціях і семінарах:

– XXIV Міжнародна конференція з Автоматичного Управління «АВТОМАТИКА 2017» (Київ, 2017).

– Міжнародна наукова конференція «Інтелектуальні системи прийняття рішень та проблеми обчислювального інтелекту» (ISDMCI) (Залізний порт, 2018).

– 2018 IEEE First International Conference on System Analysis & Intelligent Computing (SAIC) (Kyiv, 2018).

– 6th International Conference on Control and Optimization with Industrial Applications (COIA) (Baku, 2018).

– 10th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS) (Metz, 2019).

– 5th IEEE International Symposium on Smart and Wireless Systems within the International Conferences on Intelligent Data Acquisition and Advanced Computing Systems (IDAACS-SWS 2020) (Dortmund, Germany).

Публікації. Результати дисертації опубліковано в 13 друкованих працях. Статей – 7, з яких 1 стаття – у науковому фаховому виданні України з Переліку, затвердженого МОН України; 4 статті в наукових журналах, включених до міжнародних наукометричних баз; 2 статті в наукових журналах, включених до

Scopus; 6 публікацій у працях і тезах доповідей міжнародних та всеукраїнських наукових конференцій, з яких 3 публікації в наукових конференціях, включених до Scopus та/або Web of Science.

Структура та обсяг роботи. Дисертаційна робота складається зі вступу, 4 розділів, висновків, списку використаних джерел (153 найменування на 18 сторінках), 3 додатків (на 13 сторінках). Основний текст роботи викладено на 129 сторінках. Загальний обсяг роботи становить 176 сторінок.

ОСНОВНИЙ ЗМІСТ ДИСЕРТАЦІЇ

У **вступі** обґрунтовано актуальність теми дисертації, сформульовано мету, об’єкт, предмет, завдання дослідження, наукову новизну одержаних результатів, практичне значення результатів, зв’язок роботи з науковими програмами, планами, темами досліджень Національного технічного університету України «Київський політехнічний інститут імені Ігоря Сікорського». Визначено особистий внесок здобувача, наведено відомості про апробацію результатів роботи, публікації.

У першому розділі дисертації здійснено порівняльний аналіз основних механізмів тестування програмного забезпечення систем. Показано, що під час тестування великих систем, які містять клієнтські та серверні частини, важливо перевіряти не лише роботу системи, якою користується невелика кількість користувачів, а і за максимальних навантажень, які можуть спричинити непроходження тестів та збоїв у роботі системи. Важливою характеристикою також стає час реагування системи на запити, відправлені до неї.

Розглянуто піраміду тестування Майка Кона. Згідно з механізмами тестування піраміди, тестувальнику потрібно писати багато тестів невеликого об’єму та швидких юніт-тестів. Для написання загальних тестів слід використовувати зовсім мало високорівневих наскрізних тестів, які перевіряють програмне забезпечення від початку до кінця. При цьому потрібно слідкувати за тим, щоб у результаті механізми використання піраміди не вимагали великих затрат часу.

Досліджено можливості застосування фреймворків тестування для тестування розподілених баз даних, кешів, API (application programming interface) та інтерфейсів. Показано, що за допомогою Jepsen та QuickCheck тестують розподілені бази даних та кеші, а для тестування розподілених API та інтерфейсів краще використовувати Catcher. Фреймворк Jespen дозволяє легко діагностовувати результати проходження тестів, а фреймворк Catcher найзручніший для написання тестів.

Показано, що розглянуті фреймворки для тестування розподілених систем не повною мірою задовольняють роботу тестувальника програмного забезпечення і спричиняють вивчення нових технологій або написання коду в закритих системах.

Другий розділ присвячено розробленню основних теоретичних положень щодо тестування розподілених систем обробки інформації. Для ефективного тестування розподілених систем виконано модифікацію піраміди Майка Кона

таким чином, щоб вона повною мірою використовувала архітектуру систем розподіленої обробки даних. Для цього розглянуто різні види тестів і визначено ефективність їх реалізації.

У модифікованій піраміді Майка Кона нижній рівень залишається без змін (рис. 1), при цьому тести сервісу і наскрізні тести замінюються з урахуванням архітектури розподілених систем.

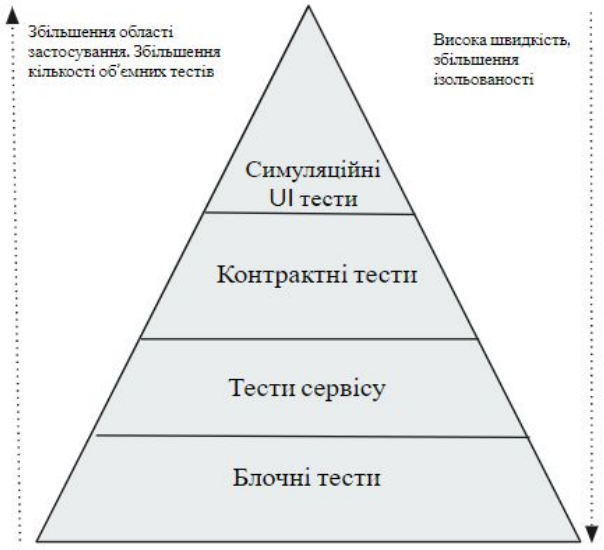

Рис. 1. Модифікована піраміда Майка Кона для тестування розподілених систем обробки інформації

Використання наскрізних тестів в архітектурі розподілених систем є неефективним підходом, а одним зі способів реалізувати наскрізні тести без використання реальних підсистем є використання «контрактів», складених на основі запитів від підсистеми. Під контрактом розуміють код тесту, який запускається в режимі постачальника.

Під час використання контрактів слід визначити очікування споживача від певного сервісу (постачальника). Ці тести мають бути виконані відносно окремого постачальника, який перебуває в ізоляції, тому такі тести будуть значно швидші та надійніші, ніж звичайні наскрізні тести для тестування сервісів API.

Обираючи метод тестування, потрібно визначити спосіб тестування сервісів розподіленої системи обробки інформації, для цього обирають схему, коли тестування проводиться на цілій системі, що складається з двох сервісів. Приймемо, що у першій схемі тестувальнику відомі всі параметри сервісів, які потрібно протестувати, а у другій – схема тестування буде відповідати системі, в якій запит від тестувальника буде проходити через два сервіси одночасно.

Для порівняння схем доцільно обчислити P^* справного стану другого сервісу, де P^* – апостеріорна імовірність справного стану сервісу 2 за умови, що результат тестування $r_2 = 0$.

Характеристика P^* буде враховуватиме припущення:

1) справність сервісів 1, 2;

2) надійність тестувальника;

3) імовірність правильного оцінювання відповіді на тест несправним тестувальником (якщо тестувальник несправний, то він видає результат перевірки навмання 0 або 1. Вважатимемо, що з імовірністю $P_r = 0,5$ видає 0 і з імовірністю $P_r = 0,5$ видає 1);

4) систему оцінювання:

$$r = \begin{cases} 1, \text{ якщо } \textit{тестувальник та модуль, що перевіряється справні}; \\ 0, \textit{ якщо тестувальник справний, а модуль, що перевіряється несправний}; \\ 0 \vee 1, \textit{ якщо тестувальник несправний}. \end{cases}$$

Розглянемо першу схему тестування двох сервісів, які тестуються паралельно, при цьому тестувальнику відомі запити та відповіді, які відправляються та отримуються (рис. 2).

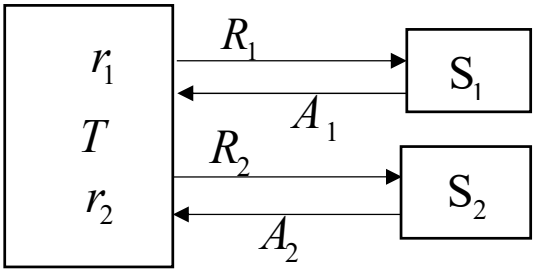

Рис. 2. Схема 1 тестування двох сервісів паралельно

У схемі 1 тестувальник: відправляє тест R_1 на сервіс S_1; якщо сервіс S_1 справний, то він передає відповідь A_1 тестувальнику; тестувальник приймає відповідь A_1 та видає результат r_1 за формулою:

$$r_1 = \begin{cases} 0, \text{ якщо } A_1 = A_{etal}, \\ 1, \text{ якщо } A_1 \neq A_{etal}, \end{cases} \quad (1)$$

де A_{etal} – еталонна відповідь для перевірки запиту до сервісу. Якщо результат $r_1 = 0$, то тестувальник вважає сервіс S_1 справним, і навпаки, якщо результат $r_1 = 1$, то тестувальник вважає сервіс S_1 несправним.

Аналогічним чином тестувальник проводить тестування другого сервісу першої схеми. Звідси за першою схемою тестувальник T та другий сервіс мають такі показники надійності:

1) апріорна до випробування імовірність справного стану тестувальника P_T, тоді імовірність відмови вираховують як $q_T = 1 - P_T$;

2) апріорна імовірність P_2 справного сервісу 2, тоді імовірність відмови вираховують як $q_2 = 1 - P_2$.

Визначають можливі чотири гіпотези H_i справного і несправного станів тестувальника та другого сервісу, для яких описано такі твердження:

$1 - \text{P}(H_1) = P_T * P_2,\ \text{P}(\text{A}/\ H_1) = 1.$ $\quad 2 - \text{P}(H_2) = P_T * \text{q}_2,\ \text{P}(\text{A}/\ H_2) = 0.$

$3 - \text{P}(H_3) = q_T * P_2,\ \text{P}(\text{A}/\ H_3) = 0,5.$ $\quad 4 - \text{P}(H_4) = q_T * q_2,\ \text{P}(\text{A}/\ H_4) = 0,5.$

При цьому $\sum_{i=1}^{4} P(H_i) = 1.$

Як подію A беремо подію отримання результату $r_2 = 0$. Тому очевидно, що для справного стану тестувальника та сервісу 2, то результат $r_2 = 0$ буде отримано з імовірністю $\mathrm{P}(\mathrm{A}/\,H_1) = 1$. Якщо тестувальник справний, а стан сервісу 2 несправний, то результат $r_2 = 0$ буде отримано з імовірністю $\mathrm{P}(\mathrm{A}/\,H_2) = 0$. В інших двох варіантах стан тестувальника є несправним, тому він «вгадує» результат з імовірністю $P_r = 0{,}5$, й, відповідно, $\mathrm{P}(\mathrm{A}/\,H_3) = \mathrm{P}(\mathrm{A}/\,H_4) = 0{,}5$.

Повну імовірність

$$P(A) = \sum_{i=1}^{4} P(H_i) \cdot P(A/H_i) = P_T P_2 + P_r q_T P_2 + P_r q_T q_2 = P_T P_2 + P_r q_T. \quad (2)$$

Тоді за теоремою Байєса можна обчислити апостеріорні ймовірності гіпотез і визначити апостеріорну імовірність достовірності тестування другого сервісу першої схеми

$$P_1^* = P(\mathrm{H}_1/\,\mathrm{A}) + P(\mathrm{H}_3/\,\mathrm{A}) = \frac{P_T P_2 + q_T P_2 P_r}{P_T P_2 + q_T P_r}. \quad (3)$$

Розглянемо другу схему тестування двох сервісів, які працюють послідовно (рис. 3).

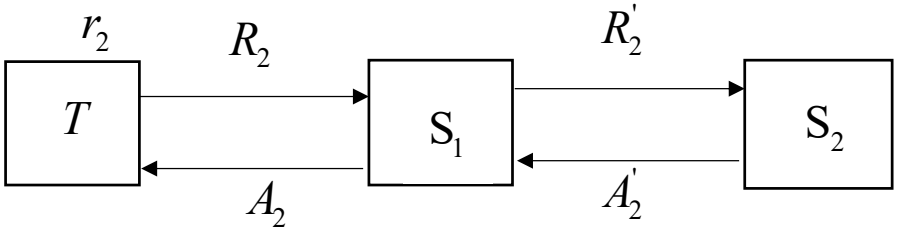

Рис. 3. Схема тестування двох сервісів, які працюють послідовно

У схемі на рис. 3 тестувальник: відправляє тест R_2 через сервіс S_1 на сервіс S_2; якщо сервіс S_1 справний, то він передає тест R_2 без спотворень; відповідь A_2 на тест R_2 передається також через сервіс S_1, який у разі справності передає її без спотворень; тестувальник приймає відповідь A_2 та видає результат r_2 за формулою (1).

В результаті тестувальник T та сервіси мають такі показники надійності:

1) апріорна до випробування імовірність справного стану тестувальника P_T, тоді імовірність відмови вираховують як $q_T = 1 - P_T$;

2) апріорна імовірність P_1 справного сервісу 1, тоді імовірність відмови вираховують як $q_1 = 1 - P_1$;

3) апріорна імовірність P_2 справного сервісу 2, тоді імовірність відмови вираховують як $q_2 = 1 - P_2$.

Можливі вісім гіпотез H_i справного і несправного станів тестувальника та сервісу 2. За подію A беруть подію отримання результату $r_2 = 0$. Тому очевидно, що коли стан тестувальник, сервісу 1 та сервісу 2 є справним, то

результат $r_2 = 0$ буде отримано з імовірністю $\mathrm{P}(\mathrm{A}/\, H_1) = 1$. Якщо стан тестувальник справний, а сервіс 2 несправний, то результат $r_2 = 0$ буде отримано з імовірністю $\mathrm{P}(\mathrm{A}/\, H_2) = 0$. В інших шести варіантах несправним є стан тестувальника або сервісу 1, тому тестувальник «вгадує» результат з імовірністю $P_r = 0{,}5$.

В такому разі повна імовірність

$$\mathrm{P}(\mathrm{A}) = \sum_{i=1}^{8} (\mathrm{P}(H_i)\,\mathrm{P}(\mathrm{A}/\, H_i)) =$$

$$P_T P_1 P_2 + P_r P_T q_1 P_2 + P_r P_T q_1 q_2 + P_r q_T P_1 P_2 + P_r q_T P_1 q_2 + P_r q_T q_1 P_2 + P_r q_T q_1 q_2 = \quad (4)$$

$$P_T P_1 P_2 + P_r P_T q_1 + P_r q_T P_1 + P_r q_T q_1.$$

Звідси апостеріорну імовірність достовірності тестування другого сервісу за другою схемою розраховують за формулою

$$P_2^* = P(\mathrm{H}_1/\,\mathrm{A}) + P(\mathrm{H}_3/\,\mathrm{A}) + P(\mathrm{H}_5/\,\mathrm{A}) + P(\mathrm{H}_7/\,\mathrm{A}) =$$

$$\frac{P_T P_1 P_2 + P_r P_T q_1 P_2 + P_r q_T P_1 P_2 + P_r q_T q_1 P_2}{P_T P_1 P_2 + P_r P_T q_1 + P_r q_T P_1 + P_r q_T q_1} \quad . \quad (5)$$

Виведемо формулу для оцінки апостеріорної імовірності для схеми з n вузлів, які розміщені послідовно та тестується останній вузол (рис. 4).

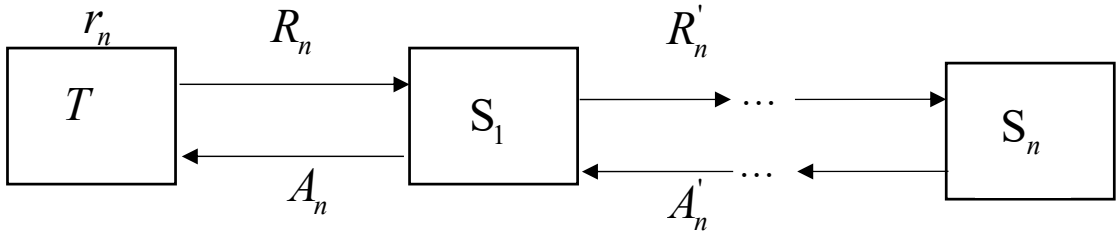

Рис. 4. Схема тестування n сервісу

При цьому за розширеною схемою тестувальник T та сервіси мають такі показники надійності:

1) апріорна до випробування імовірність справного стану тестувальника P_T, тоді імовірність відмови вираховують як $q_T = 1 - P_T$;

2) апріорна імовірність P_1 справного сервісу 1, тоді імовірність відмови вираховують як $q_1 = 1 - P_1$;

…

n) апріорна імовірність P_n справного сервісу n, тоді імовірність відмови вираховують як $q_n = 1 - P_n$.

Можливі 2^n гіпотез H_i справного і несправного станів тестувальника та сервісу n:

$1 - \mathrm{P}(H_1) = P_T * P_1 * ... * P_n,\ \mathrm{P}(\mathrm{A}/\, H_1) = 1.$

$2 - \mathrm{P}(H_2) = P_T * P_1 * ... * P_{n-1} * \mathrm{q}_n,\ \mathrm{P}(\mathrm{A}/\, H_2) = 0.$

…

$2^n - \mathrm{P}(H_{2^n}) = q_T * q_1 * ... * q_n,\ \mathrm{P}(\mathrm{A}/\, H_{2^n}) = 0{,}5.$

За подію A беруть подію отримання результату $r_2 = 0$, тому очевидно, що коли стан тестувальника та сервісів є справним, то результат $r_2 = 0$ буде отримано з імовірністю $\mathrm{P}(\mathrm{A}/\ H_1) = 1$. Якщо стан тестувальника справний, а стан сервіс n несправний, то результат $r_2 = 0$ буде отримано з імовірністю $\mathrm{P}(\mathrm{A}/\ H_2) = 0$. В інших $2^n - 2$ варіантах несправним є стан тестувальника або сервісу n, тому тестувальник «вгадує» результат з імовірністю $P_r = 0{,}5$, і, відповідно, імовірність

$$\mathrm{P}(\mathrm{A}/\ H_3) = \mathrm{P}(\mathrm{A}/\ H_4) = \ldots = \mathrm{P}(\mathrm{A}/\ H_{2^n}) = 0{,}5,$$

а повна ймовірність

$$\mathrm{P}(\mathrm{A}) = \sum_{i=1}^{2^n} (\mathrm{P}(H_i)\,\mathrm{P}(\mathrm{A}/\ H_i)). \qquad (6)$$

Звідси апостеріорна імовірність достовірності тестування сервісу n за схемою з n сервісами

$$P_n^* = P(\mathrm{H}_1/\ \mathrm{A}) + P(\mathrm{H}_3/\ \mathrm{A}) + \ldots + P(\mathrm{H}_{2^n-3}/\ \mathrm{A}) + P(\mathrm{H}_{2^n-1}/\ \mathrm{A}) = \sum_{i=1}^{2^n-1} P(\mathrm{H}_i/\ \mathrm{A}). \qquad (7)$$

Отже, за показником ефективності тестування, за який взято апостеріорну імовірність справного стану сервісу n, було запропоновано формули для розрахунку апостеріорної імовірності та розроблено модель тестування програмного забезпечення.

Аналітичні оцінки застосування експерименту для чотирьох різних апріорних імовірностей показали, що зі зменшенням апріорної імовірності графік залежностей спадає з меншим відхиленням (рис. 5).

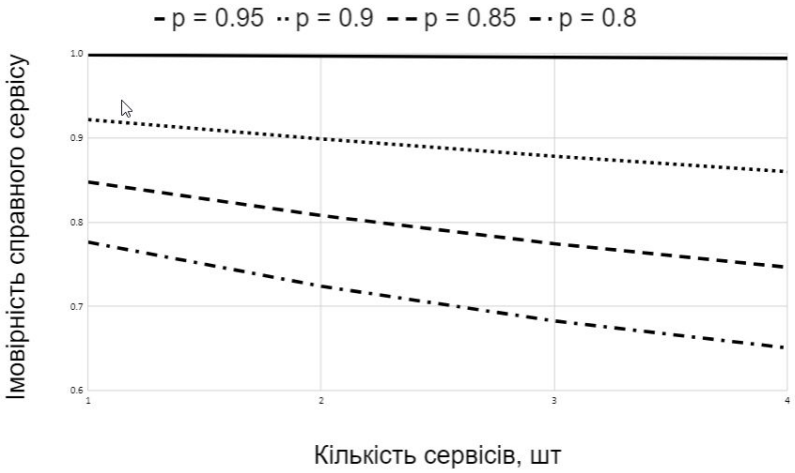

Рис. 5. Визначення апостеріорних імовірностей справного стану сервісу

Таким чином, розроблена модель тестування програмного забезпечення розподілених систем на основі автономного розгортання програмних компонентів, яка ґрунтується на застосуванні механізмів на основі модифікованої піраміди Майка Кона, дозволяє підвищити ефективність тестування та зменшити кількість об’ємних тестів.

Третій розділ присвячено розробленню удосконалених методів тестування програмного забезпечення вузлів розподіленої системи та

тестуванню інтерфейсу користувача програмного забезпечення вузлів розподіленої системи.

Для удосконалення роботи традиційного методу тестування сервісів введено поняття сервіс-імітатор (контракт) – це окремий сервіс, який має власну адресу, протокол та порт передавання даних, які користувач може налаштувати власноруч. Тоді процес тестування сервісом-імітатором розподіленої системи відбувається у двох режимах – споживача та постачальника. Режим споживача дозволяє обробити ті запити, які приходять з сервісу, а режим постачальника дозволяє відправляти заготовки запитів на сервіс та порівнювати відповідь з еталонним значенням. Ці два режими дозволяють модифікувати традиційне тестування в послідовній схемі на тестування у паралельній схемі (рис. 6).

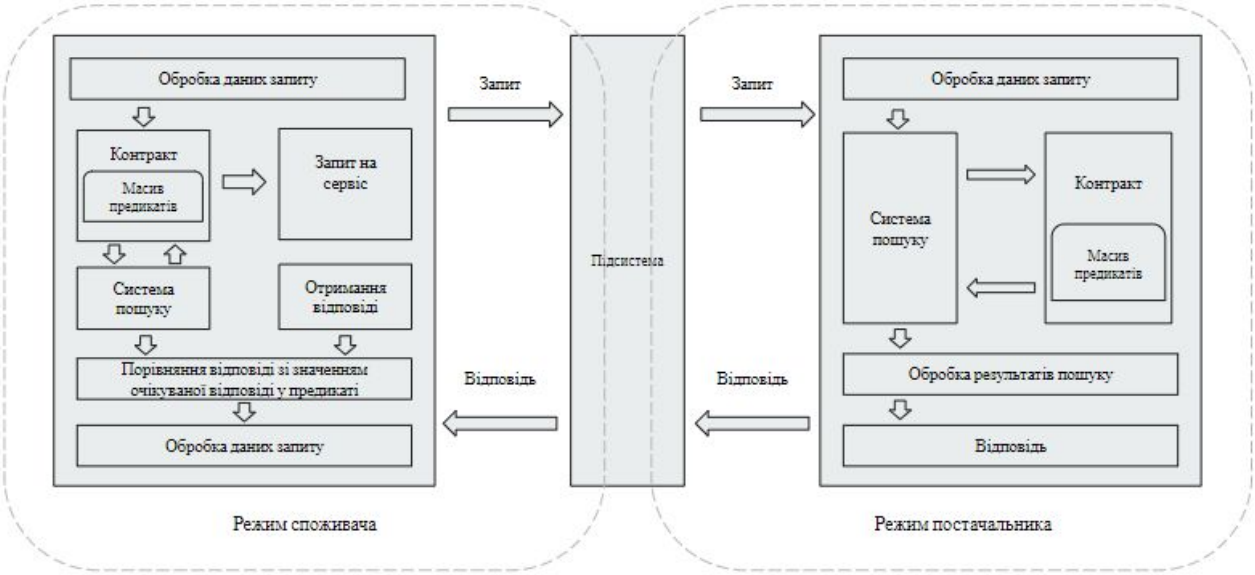

Рис. 6. Схема роботи режимів споживача та постачальника

У режимі споживача система створює контракти на основі запитів, які проходили через сервіс-імітатор. У режимі постачальника ці контракти використано як документацію. Масив вхідних даних відправляють на підсистему й тести успішно проходять, якщо отриманий результат збігається з очікуваним результатом у контракті.

Під час тестування сервісу за допомогою модифікованого механізму потрібно визначити: $tt(\mathrm{i})$ – час проходження тесту до сервісу, де $i = 1,2,...,N$ – кількість методів, $to(\mathrm{i})$ – час розгортання сервісу, $tk(\mathrm{i})$ – час розгортання контракту сервісу, $t(s)$ – час розгортання тестового середовища, – та врахувати час розгортання контракту tk сервісу для тестування.

Відповідно загальний час T, потрібний для тестування усіх компонентів розподіленої системи, буде рівний сумі часу розгортання середовища, часу розгортання сервісу, часу завантаження методів сервісів, часу тестування сервісів, а також часу згортання сервісу, тобто

$$T = \sum_{j=1}^{5}\left(\sum_{i=1}^{N}\left(t_j(\mathrm{i}) + to_j(\mathrm{i}) + tt_j(\mathrm{i})\right) + tk\right). \qquad (8)$$

Якщо механізм тестування розподіленої системи з невеликою кількістю вузлів застосовано одноразово, час тестування при класичному варіанті буде менший. Це пояснюється тим, що сума часу, витраченого на розгортання та згортання кожного сервісу, буде більшою за розгортання всієї системи. Якщо ж потрібно багато разів тестувати всю систему, то тестування за допомогою контрактів є більш вигідним, так як зміни відбуваються лише в кількох сервісах і наступного разу уже не потрібно розгортати всю систему, а тільки її окремі сервіси.

Сформовано кроки удосконаленого методу тестування програмного забезпечення вузлів розподіленої системи, який різниться від існуючих застосуванням контрактних тестів та аналізом прогнозованого результату поведінки сервісів, що дозволяє зменшити час розгортання компонентів тестового середовища та час проходження тестів, зокрема:

Крок 1. Встановлюють середовище для тестування.

Крок 2. Розгортають сервіс, який підлягатиме тестуванню.

Крок 3. Проводять тестування у режимі постачальника, при цьому система тестування моделює сервіс, який відправляє запити та отримує відповіді.

Крок 4. Проводять тестування у режимі споживача, при цьому система тестування приймає запит від сервісу та моделює відповідь на нього.

Крок 5. Записують результат проведення тесту в базу знань.

Крок 6. Згортають сервіс та очищують пам’ять у тестовому середовищі.

Крок 7. Кроки 2–6 повторюються доти, поки не будуть протестовані всі сервіси розподіленої системи обробки інформації.

Крок 8. Згортають тестове середовище.

Крок 9. Оцінюють результати тестування та готують рекомендацій розробникам.

Розроблено новий механізм на основі симуляторів, де симулятор – це сервіс (його встановлюють разом із сервісом інтерфейсу), який містить імітатори входу та виходу і дозволяє проводити його тестування.

Імітатор входу надсилає на сервіс, що піддається тестуванню, одразу весь масив запитів. Відповідно, цей сервіс, ще до початку тестування, має можливість коректно завантажити всі елементи інтерфейсу користувача, для повноцінної роботи яких потрібна інформація з інших сервісів.

Імітатор виходу містить масив очікуваних запитів та відповідей на них. Коли сервіс, який піддається тестуванню, відправляє запит на імітатор, останній проводить пошук аналогічного запиту в масиві очікуваних запитів. Коли аналогічний запит знайдено, імітатор надсилає заготовлену відповідь на сервіс. Якщо відповідь не знайдено, імітатор відправляє відповідь із кодом 404.

Отже, час тестування одного сервісу інтерфейсу модифікованим механізмом з використанням симуляторів

$$T = \sum_{i=1}^{N} (to(i) + tl(i) + tt(i) + tsi(i) + tso(i)) + t(s), \qquad (9)$$

де $to(\text{i})$ – час розгортання сервісу інтерфейсу, $i = 1,2,\ldots,N$ кількість компонентів, $tl(i)$ – час завантаження компонентів інтерфейсу сервісу, $tt(i)$ – час проходження тестування сервісу інтерфейсу, $tsi(i)$ – час розгортання стимулятора входу від сервісу інтерфейсу, $tso(i)$ – час розгортання стимулятора виходу від сервісу інтерфейсу.

Сформуємо кроки удосконаленого методу тестування інтерфейсу користувача програмного забезпечення вузлів розподіленої системи, який різниться від існуючих механізмом симуляції його роботи, що дозволяє проводити тестування окремих компонентів інтерфейсу системи, зокрема:

Крок 1. Встановлюють середовище для тестування інтерфейсів користувачів.

Крок 2. Розгортають інтерфейс користувача, який буде піддаватиметься тестуванню.

Крок 3. Завантажують симулятор входу.

Крок 4. Завантажують сторінки інтерфейсу користувача.

Крок 5. Завантажують симулятор виходу.

Крок 6. Тестують елементи інтерфейсу користувача.

Крок 7. Результати тестування записують у базу знань.

Крок 8. Згортають інтерфейс користувача.

Крок 9. Кроки 2–8 повторюють доти, поки не буде протестовано всі інтерфейси користувачів.

Крок 10. Згортають тестове середовище.

Крок 11. Оцінюють результати тестування та розробляють рекомендації розробникам програмного забезпечення.

Четвертий розділ присвячено проведенню експериментальних досліджень по перевірці достовірності моделі, яка запропонована у другому розділі, та двох удосконалених методів, запропонованих у третьому розділі. Розроблено інформаційну технологію (рис. 7) та запропоновано інформаційну систему (рис. 8) для підвищення ефективності тестування програмного забезпечення розподілених систем.

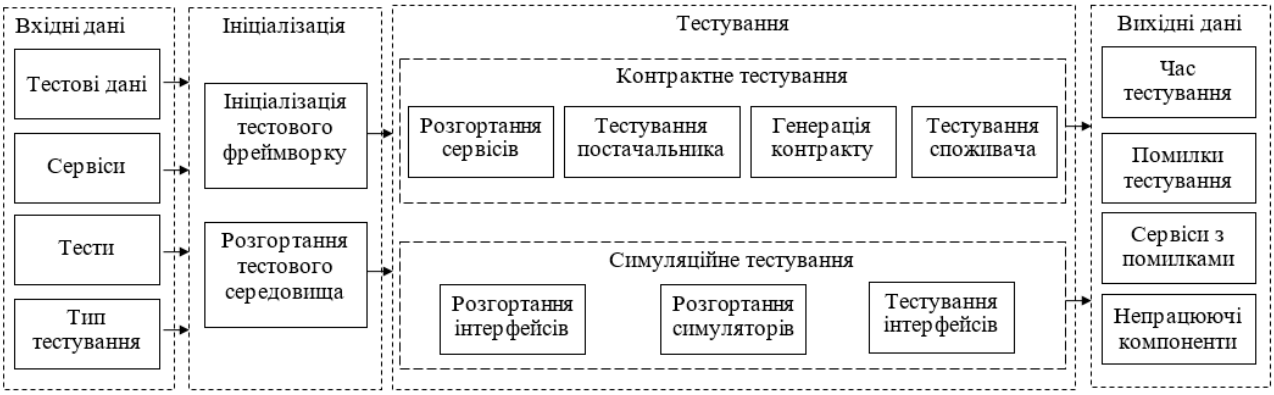

Рис. 7. Схема інформаційної технології тестування розподілених API та інтерфейсів

Для проведення експериментальних досліджень тестування розподілених систем обробки інформації було розроблено фреймворк на основі удосконалених методів та визначено час тестування (табл. 1).

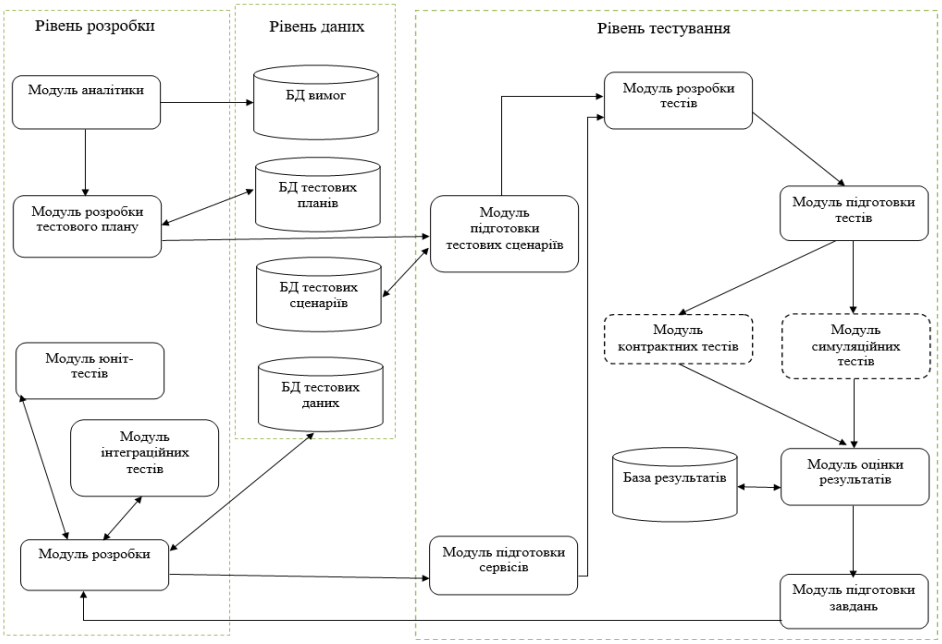

Рис 8. Функціональна схема інформаційної системи підвищення ефективності тестування розподілених систем

Таблиця 1

Результати тестування розподілених систем

Тип системи (кількість вузлів = 5)	Jepsen, с	QuickCheck, с	Catcher, с	Удосконалений метод тестування, с	Порівняння часу тестування системи, %
Розподілена система баз даних	160	416	341	162	–1.3
Розподілена система кешів	62	252	238	63	–1.6
Розподілена система API	–	105	77	72	6.5
Розподілена система інтерфейсів	–	–	286	265	7.3

Для тестування розподіленої системи API за допомогою фреймворку Catcher та удосконаленим методом з використанням контрактів було проведено експериментальні дослідження на розподілених системах з 2, 5, 10, 20, 40 та 50 вузлами (рис. 9).

Для кожного методу виміряно час проходження тестів, який для п’яти вузлів розподіленої системи був менший на 6,5 % у разі застосування механізму контрактів. Показано, що за невеликої кількості вузлів наскрізне тестування сервісів проходить швидше, ніж контрактне тестування цих же сервісів. Зі збільшенням кількості вузлів час, потрібний на тестування сервісів розподіленої системи контрактими тестами, стає меншим за час, потрібний для тестування цієї ж системи наскрізним методом.

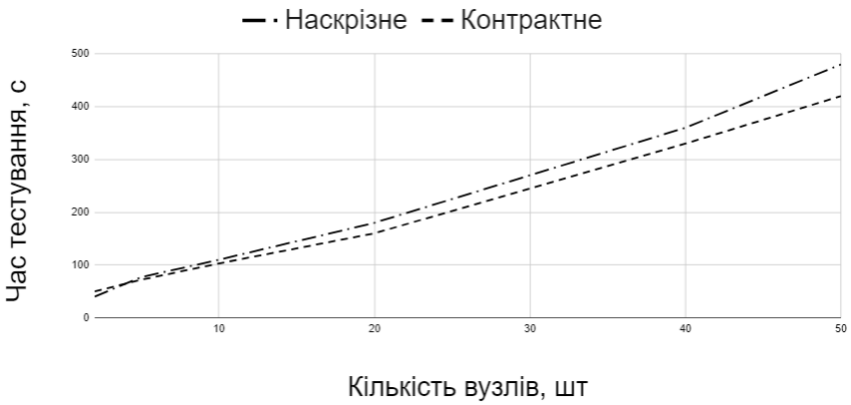

Рис. 9. Порівняння середнього часу тестування різними методами

Для тестування розподіленої системи інтерфейсів за допомогою фреймворку Catcher та удосконаленого методу з використанням симуляційних тестів було проведено експериментальні дослідження на розподілених системах з 2, 5, 10, 15 та 20 вузлами. Для кожного методу виміряно середній час проходження тестів (рис. 10). Останній, при п’яти вузлах, показав, що удосконалений метод симуляційного тестування виявився швидшим за наявний фреймворк Catcher на 7,3 %.

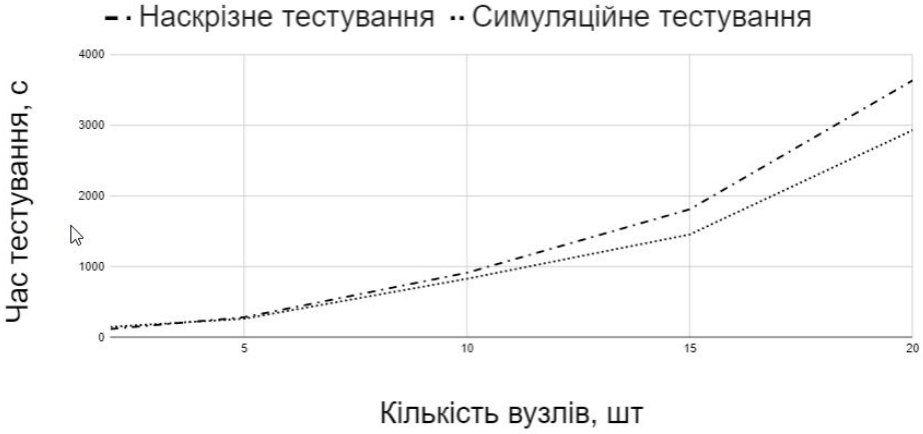

Рис. 10. Порівняння середнього часу тестування інтерфейсів різними методами

Якщо використано два сервіси інтерфейсу, наскрізне тестування проходить швидше за стимуляційне. Зі збільшенням кількості вузлів час, потрібний на тестування сервісів розподіленої системи інтерфейсів симуляційними тестами, стає меншим за час, потрібний для тестування цієї ж системи наскрізним методом.

Очевидно, що комплекс заходів щодо підвищення ефективності тестування розподілених систем обробки інформації у підсумку приведе до необхідності вирішувати нове наукове завдання вибору раціонального варіанту удосконалення розміщення вузлів системи на віртуальних серверах мережі з урахуванням не тільки топології, а й нових видів програмних продуктів. Зазначене завдання є важливим науковим напрямом наступних досліджень у цій сфері.

ОСНОВНІ РЕЗУЛЬТАТИ ТА ВИСНОВКИ

У дисертаційній роботі подано результати, які відповідно до поставленої мети є рішенням актуального науково-практичного завдання розроблення моделей, методів та інформаційної технології підтримки ефективного тестування програмного забезпечення.

Одержано такі нові теоретичні та практичні результати:

1. Проведено аналіз механізмів тестування програмного забезпечення систем. Показано, що під час тестування великих систем, які містять клієнтські та серверні частини, важливо перевіряти не лише роботу системи за невеликої кількості користувачів, а й за максимальних навантажень, що може спричинити непроходження тестів та збої у роботі системи. Важливою характеристикою також стає час реагування системи на запити, які до неї відправляються.

2. Модифіковано піраміду тестування Майка Кона. Показано, що для написання загальних тестів потрібно використовувати зовсім мало високорівневих наскрізних тестів, які перевіряють програмне забезпечення від початку до кінця. При цьому потрібно пильнувати, щоб механізми використання піраміди не призвели до великих затрат часу.

Результати порівняння фреймворків тестування показали, що для тестування розподілених баз даних та кешів підходять Jepsen та QuickCheck, а для тестування розподілених API та інтерфейсів краще використати Catcher. Проте Jespen – єдиний фреймворк, який дозволяє легко діагностовувати результати проходження тестів. Фреймворк Catcher є найзручнішим для написання тестів, причому не потрібно писати код, а лише сценарії.

3. Уперше розроблено модель тестування програмного забезпечення розподілених систем на основі автономного розгортання програмних компонентів, яка ґрунтується на застосуванні механізмів на основі модифікованої піраміди Майка Кона, що дозволяє підвищити ефективність тестування та зменшити кількість об’ємних тестів.

4. Удосконалено метод тестування програмного забезпечення вузлів розподіленої системи, який різниться від наявних застосуванням контрактних тестів та аналізом прогнозованого результату поведінки сервісів, що дозволяє зменшити час розгортання компонентів тестового середовища та час проходження тестів. Проведено експериментальне дослідження застосування методу тестування програмного забезпечення вузлів розподіленої системи з визначенням часу тестування. Для трьох та менше вузлів системи тестування контрактними тестами неефективне порівняно з механізмом наскрізного тестування, а в іншому випадку час тестування зменшується. Для п’яти вузлів розподіленої системи час тестування API зменшився на 6,5 %.

5. Удосконалено метод тестування інтерфейсу користувача програмного забезпечення вузлів розподіленої системи, який різниться від наявних підходом до симуляції його роботи, що дозволяє проводити тестування окремих компонентів інтерфейсу системи. Проведено експериментальне дослідження застосування методу тестування інтерфейсу користувача програмного

забезпечення вузлів розподіленої системи, який показав, що зі збільшенням кількості вузлів час, потрібний на тестування сервісів розподіленої системи інтерфейсів симуляційними тестами, стає меншим за час, потрібний для тестування цієї ж системи наскрізним методом. Для п’яти сервісів інтерфейсів розподіленої системи час тестування зменшився порівняно з фреймворком Catcher на 7,3 %.

6. Розроблено інформаційну технологію, що реалізує подані в роботі моделі та методи. Інформаційну систему, створену за такою інформаційною технологією, було використано для реалізації експериментальних досліджень – вона підвищує ефективність тестування програмного забезпечення розподілених систем обробки інформації.

7. Розроблені моделі, методи та інформаційна технологія підвищення ефективності тестування програмного забезпечення розподілених систем обробки інформації апробовані, застосовані та впроваджені під час розробки інформаційних систем у ТОВ «Нафтогазбудінформатика» та у конструкторському бюро інформаційних систем КПІ ім. Ігоря Сікорського для вирішення завдання зменшення часу тестування програмного забезпечення, що підтверджено відповідними актами впровадження та довідкою про застосування результатів роботи у навчальному процесі кафедри технічної кібернетики Національного технічного університету України «Київський політехнічний інститут імені Ігоря Сікорського».

СПИСОК ОПУБЛІКОВАНИХ ПРАЦЬ ЗА ТЕМОЮ ДИСЕРТАЦІЇ

Статті у періодичних зарубіжних виданнях, індексованих у наукометричній базі Scopus

1. Mukhin V., Kornaga Ya., Mostovyi Y., Bazaka Y. A Model For Events Monitoring Heterogeneous Distributed Databases Based on Vector-matrix Operations. The Far East Journal of Electronics and Communications, 2016. Vol. 16, Issue 3. P. 645 – 656. (This journal is indexed in Elsevier Scopus).

2. Zhenbing H., Mukhin V., Kornaga Ya., Herasymenko O., Bazaka Y. The scheduler for the grid system based on the parameters monitoring of the computer components. Eastern European Journal of Enterprise Technologies, 2017. № 1 (2-85). P. 31 – 39. (This journal is indexed in Elsevier Scopus).

Статті у міжнародних виданнях

3. Корнага Я.І., Герасименко О.Ю., Базака Ю.А., Базалій М.Ю., Мухін О.В. Метод автоматизації тестування розподіленої системи з використанням контрактів. Sciences of Europe, 2020. № 55. С. 45 – 49.

4. Корнага Я.І., Герасименко О.Ю., Базака Ю.А., Базалій М.Ю., Мухін О.В. Метод тестування інтерфейсу користувача з використанням імітаторів. The scientific heritage, 2020. № 51. С. 54 – 57.

5. Корнага Я.І., Герасименко О.Ю., Базака Ю.А., Базалій М.Ю., Мухін О.В. Method of Protecting Software Code from Reengineering Using Virtual Machines. Sciences of Europe, 2020. № 58. С. 59 – 62.

6. Корнага Я.І., Базака Ю.А., Базалій М.Ю. Захист програмного

забезпечення за допомогою заплутуючих перетворень. The scientific heritage, 2020. № 54. С. 72 – 75.

Статті у наукових фахових виданнях України

7. Корнага Я.І., Герасименко О.Ю., Базака Ю.А., Базалій М.Ю., Мухін О.В. Використання баз даних та мов програмування у високоточних обчисленнях чисел із плаваючою точкою великої розрядності. Вчені записки Таврійськ. нац. ун-ту ім. В.І. Вернадського, 2020. Т. 31 (70). №5. С. 82 – 88.

Публікації у працях і тезах доповідей міжнародних та всеукраїнських наукових конференцій

8. Mukhin V., Kornaga Ya., Yakovleva A., Herasymenko O., Bazaca Yu., Bazaliy M. The model for distributed systems testing based on the control services. XXIV Міжнар. конф. з Автоматичного Управління “АВТОМАТИКА – 2017”. 13 – 15 верес. 2017 р., м. Київ. 2017. С. 45 – 46.

9. Мухін В.Є., Корнага Я.І., Яковлєва А.П., Базалій М.М., Базака Ю.А. Модифікований метод обфускації програмного коду за допомогою вставки інструкцій. Міжнар. наук. конф. «Інтелектуальні системи прийняття рішень та проблеми обчислювального інтелекту» (ISDMCI’2018), 21 – 27 трав. 2018 р., с. Залізний порт. – Херсон: Вид-во ПП Вишемирський, 2018. С. 67 – 68.

10. Mukhin V., Kornaga Ya., Bazaka Yu., Bazaliy M., Yakovleva A. Modified Method of Software Testing for Distributed Computer System. 2018 IEEE First International Conference on System Analysis & Intelligent Computing (SAIC). 8 – 12 October 2018, Kyiv, Ukraine. 2018. P. 1 – 4. (This conference is indexed in Elsevier Scopus).

11. Mukhin V.Ye., Kornaga Ya.I., Abdullayev S.H., Gerasymenko O.Yu., Bazaka Yu.A. Method of the reserve resources determination for the distributed computer systems with the network-centric resource control. 6th International Conference on Control and Optimization with Industrial Applications (COIA). 11 – 13 July 2018. Baku, Azerbaijan. 2018. P. 226 – 231. (This conference is indexed in Web of Science).

12. Mukhin V., Vyshnivskyi V., Kornaga Ya., Herasymenko O., Bazaka Yu., Bazaliy M. Study of the functioning of the distributed computer system with a resource control mechanism based on a network-centric approach. 10th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications, IDAACS 2019. 18 – 21 September 2019. Metz, France. 2019. P. 100 – 105. (This conference is indexed in Elsevier Scopus, Web of Science).

13. Mukhin V., Kornaga Y., Tkach M., Herasymenko O., Bazaka Y., Mukhin O. Subtask Prioritization on Workflow Execution in Distributed Wireless Computer System with Network-Centric Approach to Resource Control. 5th IEEE International Symposium on Smart and Wireless Systems within the International Conferences on Intelligent Data Acquisition and Advanced Computing Systems, IDAACS-SWS 2020, 17 September 2020. Dortmund, Germany. 2020. P. 1 – 5. (This conference is indexed in Elsevier Scopus, Web of Science).

АНОТАЦІЯ

Базака Ю.А. Моделі, методи та інформаційна технологія підвищення ефективності тестування розподілених систем. – Кваліфікаційна наукова праця на правах рукопису.

Дисертація на здобуття наукового ступеня кандидата технічних наук за спеціальністю 05.13.06 «Інформаційні технології». – Західноукраїнський національний університет, Тернопіль, 2021.

Дисертація присвячена вирішенню актуальної задачі розроблення моделей і методів тестування компонентів програмного забезпечення шляхом створення інформаційної технології, що дозволяє підвищити ефективність функціонування розподілених систем обробки інформації.

Запропоновано модель тестування програмного забезпечення розподілених систем на основі автономного розгортання програмних компонентів. Удосконалено піраміду тестування Майка Кона для застосування її під час тестування розподілених систем.

Модифіковано метод тестування програмного забезпечення вузлів розподіленої системи. Розроблено механізм застосуванням контрактних тестів з аналізом прогнозованого результату поведінки сервісів. Модифіковано метод тестування інтерфейсу користувача програмного забезпечення вузлів розподіленої системи.

Ключові слова: тестування програмного забезпечення, розподілена система обробки інформації, інформаційна система.

АННОТАЦИЯ

Базака Ю.А. Модели, методы и информационная технология повышения эффективности тестирования распределенных систем. – Квалификационный научный труд на правах рукописи.

Диссертация на соискание ученой степени кандидата технических наук по специальности 05.13.06 «Информационные технологии». – Западноукраинский национальный университет, Тернополь, 2021.

Диссертация посвящена решению актуальной задачи разработки моделей и методов тестирования компонентов программного обеспечения путем создания информационной технологии, позволяющей повысить эффективность функционирования распределенных систем обработки информации.

Предложена модель тестирования программного обеспечения распределенных систем на основе автономного развертывания программных компонентов. Усовершенствована пирамида тестирования Майка Кона для применения ее при тестирования распределенных систем.

Модифицирован метод тестирования программного обеспечения узлов распределенной системы. Разработан механизм применения контрактных тестов с анализом прогнозируемого результата поведения сервисов. Модифицирован метод тестирования интерфейса программного обеспечения узлов распределенной системы.

Ключевые слова: тестирование программного обеспечения, распределенная система обработки информации, информационная система.

ABSTRACT

Bazaka Yu.A. Models, methods and information technology to increase the efficiency of testing distributed systems. – Qualifying scientific work on the rights of manuscript.

The dissertation on competition of a scientific degree of the candidate of technical sciences on a specialty 05.13.06 “Information technologies”. – West Ukrainian National University, Ternopil, 2021.

The dissertation is devoted to the decision of actual problem of software testing models and methods development of components by the creation of information technology that allows increasing the efficiency of distributed systems functioning.

The analysis of software testing systems is carried out. The considered frameworks for testing distributed systems do not fully satisfy the work of a software tester who has to learn new technologies or write code in closed systems.

A model of software testing of distributed systems based on software components autonomous deployment is proposed. Modified the Mike Kohn testing pyramid for adaptation when testing in distributed information processing systems, which allowed to expand the testing capabilities and apply the features of distributed systems. The comparison of schemes with different numbers of services on the indicator of testing efficiency, as which the a posteriori probability of good condition are accepted on condition of receiving a positive test result, is made.

The method of distributed system software testing nodes has been modified. The mechanism of application of contract tests with the analysis of the forecasted result of the service’s behavior is developed. The software testing method for the distributed system user interface of nodes has been modified.

It is shown that in comparison with end-to-end testing of user interfaces, the advantages of using simulators for testing user interfaces allow reducing the time spent on testing any service of user interface in comparison with end-to-end testing. The time is reduced by reducing the number of simultaneous user interface services.

An experimental study of the application of the distributed system software testing method of nodes in determining the average testing time. For three or fewer nodes, the contract testing system is ineffective compared to the end-to-end testing mechanism, otherwise, the average testing time is reduced. For 5 nodes of the distributed system, the API testing time decreased by 6.5%.

An experimental study of the application of the user interface software testing method of the distributed system nodes, which showed that with the increasing number of nodes, the time required to test distributed system services by simulation tests becomes less than the time required to test the same system through. For 5 interfaces of the distributed system, the test time decreased compared to Selenium WebDriver by 7.3%.

Keywords: software testing, distributed information processing system, information system.

Підписано до друку 30.03.2021 р.
Формат 60х90/16. Гарнітура Times.
Папір офсетний. Друк на дублікаторі.
Умов. друк. арк. 0,9. Обл.-вид. арк. 1,0.
Зам. № А015-21. Тираж 100 прим.

Видавець та виготовлювач
Західноукраїнський національний університет
вул. Львівська, 11, м. Тернопіль 46009

Свідоцтво про внесення суб'єкта видавничої справи
до Державного реєстру видавців ДК № 3467 від 23.04.2009 р.

Видавничо-поліграфічний центр «Університетська думка»
вул. Бережанська, 2, м. Тернопіль 46009
тел. (0352) 47-58-72
E-mail: edition@wunu.edu.ua