 ISSN 2071-1077(print)
ISSN 2663-2411(online

doi: 10.32620/oikit.2026.107.11

УДК 004.05

І. В. Шостак, О. В. Череватенко

Аналіз проблеми тестування при розробленні програмних проєктів в умовах аутсорсингу для непрофільних замовників

Національний аерокосмічний університет «Харківський авіаційний інститут»

Робота присвячена проблемі підвищення ефективності **процесу тестування** при розробленні програмних проєктів в Agile-середовищі аутсорсингових IT- компаній**. Актуальність** даного дослідження обумовлена кон'юнктурою сучасного ринку IT-послуг і жорсткими умовами щодо конкурентоспроможності аутсорсингових компаній. **Об'єкт дослідження** – процеси забезпечення якості (Quality Assurance) програмних проєктів, які реалізуються в Agile-середовищі аутсорсингових компаній. На даний час Agile-маніфест і гнучкі методології (Scrum, Kanban, Lean тощо) є природним середовищем (Agile-середовищем) для розроблення аутсорсингових програмних проєктів. Питання організації виробничих процесів у такому середовищі для різних за тематикою проєктів, (стартапи, SaaS, фінансові технології, телемедицина, логістика тощо) добре досліджені і підтримуються розвинутим інструментарієм (Jira, Azure та інші).

Разом з цим, досвід і проведені ретроспективні дослідження вказують на присутність системних факторів-тригерів істотного падіння ефективності роботи команд. Загалом результати цих досліджень інтерпретуються, як існування специфічних проблем, що, як правило, виникають в проєктах з розподіленими командами і непрофільними замовниками. Отже, питання забезпечення ефективності процесів розроблення в указаному аспекті є відкритим. **Предмет дослідження** – проблеми, що виникають у процесі тестування при розробленні аутсорсингових програмних проєктів і методи їх вирішення. **Мета:** На підставі аналізу практичного досвіду сформулювати специфічні проблеми тестування аутсорсингових проєктів, що розробляються для непрофільних замовників, та які можуть бути причиною зниження ефективності команд і, з використанням відомих практик, розробити рекомендації щодо усунення або зменшення впливу цих проблем.

Для досягнення зазначеної мети, у ході дослідження були поставлені і вирішені такі **задачі:** конкретизовані і систематизовані основні проблеми процесу тестування, проаналізовані і класифіковані за ступенем значущості причини їх виникнення, запропоновані рекомендації щодо усунення цих проблем. **Отримані результати:** у рамках якісного дослідження, основаного на систематизації досвіду аутсорсингових компаній і аналізу досвіду типових сценаріїв, були виділені ключові проблеми, з якими зустрічаються команди тестувальників. Серед них – недостатнє залучення тестувальників на ранніх етапах процесів розроблення, складність у комунікації і зборі вимог, проблеми з оцінкою «готовності» функціоналу, актуалізація тестової документації, управління дефектами, обмеження часу і ресурсів на тестування, труднощі з автоматизацією, обсяги регресійного тестування, приймальне тестування і звітність, формування реалістичних очікувань замовника, передача знань і мотивація команди. Запропоновані рекомендації щодо вирішення означених проблем. Вони включають активне запровадження у виробничі процеси підходу "Shift-Left" і сумісне формування вимог, побудова прозорих і структурованих комунікацій стейкхолдери–девелопери, розроблення гнучких, але чітких критеріїв завершення завдань ("Definition of Done") та ітеративних процесів приймального тестування, а також, застосування адаптивних підходів до тестової документації і автоматизації. Розроблені рекомендації передбачають удосконалення процесу управління дефектами, ефективності планування тестового циклу, автоматизацію регресійного тестування, розроблення адаптивної системи метрик якості і звітності, а також проактивне управління очікуваннями замовника, систематизацію і обмін досвідом команд.

***Ключові слова*:** Agile, аутсорсинг, менеджмент програмних проєктів, тестування, забезпечення якості (QA), Shift-Left, DoD.

Вступ

В умовах динамічного розвитку інформаційних технологій і зростаючої складності програмних продуктів, забезпечення якості (Quality Assurance, QA) є важливим фактором успіху проєкту. Особливого значення ця задача набуває при аутсорсингу програмних проєктів, де замовники, не дивлячись на глибоке розуміння свого бізнесу, можуть не володіти компетенціями IT-галузі, достатніми для чіткого і однозначного формулювання технічного завдання командам виконавців. Більш того, типовими є випадки, коли робота починається з формулювання ідеї проєкту з подальшими уточненнями і деталізацією безпосередньо в процесі розроблення. Додатковою сучасною специфікою таких проєктів є труднощі взаємодії між замовником і виконавцем в умовах значної географічної, часової і культурної роз'єднаності.

Актуальність даної роботи обумовлена кон'юнктурою сучасного ринку IT-послуг і жорсткими умовами щодо конкурентоспроможності аутсорсингових компаній.

Об'єкт дослідження – процеси забезпечення якості (Quality Assurance) в програмних проєктах, які реалізуються в Agile-середовищі аутсорсингової компанії. На даний час Agile-маніфест і гнучкі методології (Scrum, Kanban, Lean тощо) є природним середовищем (Agile-середовище) для розроблення аутсорсингових програмних проєктів. Питання побудови виробничих процесів у такому середовищі для проєктів, найрізноманітніших за тематикою (сфера обслуговування, SaaS, фінансові технології, телемедицина, управління логістикою автоперевезень тощо) добре досліджені і підтримуються розвинутим інструментарієм (Jira, Azure, Trello тощо).

Разом з цим, досвід і проведені ретроспективні дослідження вказують на присутність системних факторів-тригерів істотного падіння ефективності роботи команд. Загалом, результати цих досліджень інтерпретуються, як існування специфічних проблем, що, як правило, виникають в проєктах з розподіленими командами і непрофільними замовниками. Отже, питання ефективності і якості процесів розроблення в указаному аспекті є відкритим.

Предмет дослідження – проблеми, що виникають у процесі тестування при розроблені аутсорсингових програмних проєктів і методи їх вирішення.

Мета: На підставі аналізу практичного досвіду сформулювати специфічні проблеми тестування аутсорсингових проєктів для непрофільних замовників, які можуть бути причиною зниження ефективності команд і, з використанням відомих практик, розробити рекомендації щодо усунення або зменшення впливу цих проблем.

Для досягнення поставленої мети були сформульовані такі задачі:

1. **Виявити і систематизувати** основні проблеми, з якими стикаються команди тестувальників в процесі розроблення і тестування аутсорсингових проєктів.
2. **Проаналізувати,** з урахуванням специфіки аутсорсингу, причини виникнення цих проблем.
3. **Класифікувати** виявлені проблеми за ступенем їх значимості (висока, середня, помірна) для процесу тестування та успішності проєкту.
4. **Розробити конкретні та практичні рекомендації** щодо підвищення якості процесу тестування для кожної категорії проблем.
5. **Обґрунтувати** для аутсорсингових компаній та їх замовників можливість підвищення якості процесу тестування при впровадженні запропонованих

рекомендацій.

Наукова новизна дослідження полягає у:

- систематизації та приорітизації проблем, характерних для процесу тестування саме в умовах аутсорсингу і непрофільного замовника;
- розробленні набору рекомендацій, які враховують аспекти саме такої взаємодії;
- виділенні задачі проактивного управління очікуваннями і міжкультурною комунікацією в процесі тестування проєкту.

1. Огляд літератури і теоретичні основи

Шляхи підвищення ефективності тестування в процесі розроблення програмних проєктів широко висвітлені в науковій та технічній літературі. Проте, з розвитком Agile-маніфесту та гнучких методологій XP, Scrum, Kanban та ін. відбувся істотний зсув від традиційних "водоспадних" моделей процесу розроблення до ітеративних, які забезпечували в умовах аутсорсингу обернений зв’язок, неперервну інтеграцію і швидкий вихід продукту на ринок [1, 15].

Концепція "Shift-Left Testing", згідно до якої тестувальники залучаються до процесу розроблення, починаючи з самих ранніх етапів формування вимог до продукту [8], є однією з основних в Agile QA. Цей підхід спрямований на виявлення та усунення дефектів на ранніх стадіях проєкту, що істотно зменшує загальну вартість тестування і підвищує якість продукту [13]. Наприклад, у рамках "Shift-Left" активно використовуються техніки Behavior-Driven Development (BDD) та Test-Driven Development (TDD), які дозволяють формувати зрозумілі для замовника специфікації та водночас слугують основою для автоматизованих тестів.

Разом з цим, аутсорсингові компанії широко використовують розподілені команди для виконання проєктів. Це є цілком природним, але створює в Agile-середовищі компанії додаткові труднощі в **комунікації, синхронізації і координації**, які обумовлені географічними і часовими бар’єрами, а також культурними відмінностями [5]. Ефективне управління цими аспектами вимагає впровадження додаткових заходів і практик, спрямованих на підвищення прозорості і довіри у стосунках.

Саме поняття аутсорсингу вказує і на друге джерело зростання труднощів розроблення проєктів в Agile-середовищі. Це, так звані, **непрофільні замовники.** Як правило, для таких компаній програмні проєкти, що замовляються, є засобами автоматизації у процесах досягнення бізнес-цілей. Відсутність у клієнта глибокого технічного розуміння реалізації проєкту, може призводити до:

- Нечітких, неповних або постійно змінюваних вимог [21].
- Нереалістичним очікуванням щодо термінів і вартості робіт [16].
- Труднощам у приймальному тестуванні та оцінюванні якості, оскільки бізнес-користувачі можуть фокусуватися на поверхневих аспектах, упускаючи технічні, можливо критично важливі, деталі.

На даний момент теоретичні і практичні аспекти забезпечення якості програмного забезпечення (Software Quality Assurance) досліджені достатньо повно і полягають у комплексному підході, який включає не тільки тестування, а й превентивні міри, управління ризиками і постійне удосконалення процесів [13]. В Agile-маніфесті цей підхід реалізований у вигляді критеріїв **"Definition of Done" (DoD)**, безперервного тестування і автоматизації регресивних тестів [3, 7].

Слід зазначити, що в останні роки автоматизація тестування, яка зменшує обсяг рутинного ручного тестування і прискорює цикли оберненого зв’язку, стала одним із ключових напрямків підтримки якості при частих змінах і стиснутих термінах проєктів в умовах аутсорсингу [6, 10].

Проте, не дивлячись на велику кількість досліджень в області Agile та аутсорсингу, недостатньо комплексних робіт, які б систематизували проблеми тестування саме у зв’язці аутсорсинг – Agile – розподілені команди – взаємодія із непрофільними замовниками. Більшість досліджень фокусується на технічних аспектах Agile-тестування або загальних проблемах аутсорсингу, не враховуючи специфіку бізнес-орієнтованих клієнтів без професійного IT-бекграунда. Дана робота має за мету заповнити цю прогалину, оскільки саме тут, на наш погляд, прихований значний потенціал підвищення ефективності і якості процесу розроблення програмних проєктів при аутсорсингу.

2. Методологія досліджень

Повне дослідження означеної проблеми підвищення ефективності та якості передбачає вирішення питання щодо метрик процесів розроблення проєктів у Agile-середовищі. Загалом це питання вирішено цілком достатньо [28], але у контексті означеної вище проблеми необхідні серйозні окремі дослідження. Отже, дана робота базується на якісному підході, який полягає в **огляді літератури, узагальненні експертного досвіду** в області управління і розроблення проєктів, а також забезпечення якості при аутсорсингу. Методологія досліджень полягає у послідовному виконанні таких етапів:

1. Ідентифікація та агрегація проблем: На початковому етапі проведений аналіз типових труднощів, з якими стикаються аутсорсингові команди. Проведено огляд кейсів із реальної практики та узагальнені проблемні сценарії, що виникають і повторюються при роботі на Agile-проєктах, які не мають глибокої IT-експертизи з боку замовника. Основна увага приділялася труднощам, пов’язаним з комунікацією, управлінням вимогами, процесом тестування і доступом до інфраструктури проєкту.

2. Причинно-наслідковий аналіз: Для кожної з виявлених проблем був проведений аналіз першопричин. Особливу увагу приділялося факторам, специфічним для аутсорсингової моделі (наприклад, географічна і культурна відмінність) та особливостей непрофільних замовників (наприклад, невизначеність бізнес-вимог, нереалістичні очікування, відсутність розуміння IT-процесів).

3. Класифікація і пріоритизація проблем: Виявлені проблеми були систематизовані і класифіковані за ступенем їх потенційного негативного впливу на успішність проєкту, якість продукту і задоволеність замовника. Використовувалася трирівнева шкала значимості: "Висока значимість" (критичні проблеми), "Середня значимість" (істотні проблеми зниження ефективності), і "Помірна значимість" (проблеми, які створюють відчуття дискомфорту і знижують мотивацію). Така класифікація дала змогу сфокусувати зусилля на найбільш критичних аспектах.

4. Розробка рекомендацій: На основі аналізу проблем і причин були сформульовані конкретні, практичні та зі здатністю до масштабування рекомендації. При розробленні рекомендацій застосовувався синтетичний підхід, який інтегрує загальновизнані методології і найкращі практики (Best practices) (такі, як "Shift-Left Testing", Behavior-Driven Development (BDD), Test-Driven

Development (TDD), стратегії автоматизації тестування, ефективні моделі комунікації для розподілених команд) з адаптацією під специфіку взаємодії з непрофільними аутсорсинговими замовниками.

5. Обґрунтування та оцінка потенційної ефективності: Для кожної запропонованої рекомендації було проведено обґрунтування її потенційної ефективності та переваг для обох сторін – аутсорсингової компанії (через підвищення ефективності, зменшення ризиків) і замовника (через підвищення якості продукту, скорочення термінів і покращення взаємодії).

Дана методика дозволила не тільки виявити і структурувати існуючі проблеми, але й запропонувати комплексний набір рішень, який може бути практичним посібником для аутсорсингових компаній (бестпрактика), які бажають оптимізувати свої процеси тестування в Agile-середовищі при роботі з непрофільними замовниками.

3. Результати досліджень та обговорення

У даному розділі представлені основні результати проведеного дослідження, які включають виявлені проблеми процесу тестування Agile-проєктів в умовах аутсорсингу з непрофільними замовниками, класифікацію цих проблем за ступенем їх значимості й рекомендації щодо їх усунення та докладне їх обговорення.

На основі аналізу та систематизації даних були виявлені такі **ключові**, на наш погляд, **проблеми**, оскільки вони істотно впливають на ефективність і якість QA-процесів. Ці проблеми класифіковані на три категорії за ступенем їх впливу.

3.1. Проблеми високої значимості та рекомендації щодо їх усунення

Проблеми високої значимості являють собою критичні бар’єри, здатні привести до системних збоїв, істотним затримкам, значному збільшенню дефектів у продукті та, як наслідок, до високого невдоволення замовника. Усунення цих проблем є першочерговою задачею управління якістю.

3.1.1. Проблема: Недостатнє залучення тестувальників аутсорсингової команди на ранніх етапах розробки зі сторони замовника

Вплив: Пізнє залучення тестувальників призводить до неправильного розуміння вимог, упущення критичних сценаріїв, експоненціальному зростанню вартості виправлення дефектів на пізніх стадіях розробки, а також ризику поставки продукту, якій не відповідає початковим або еволюціонуючим очікуванням замовника [8, 13].

Рекомендація: Впровадження підходу "Shift-Left" з акцентом на сумісне формування вимог. Принцип "Shift-Left" передбачає максимально раннє залучення тестувальників у життєвий цикл проєкту [8]. Для аутсорсингових команд це означає проактивну участь вже на стадії формування бізнес-ідеї проєкту і початкових вимог. Тестувальники, наприклад, можуть виступати у ролі майбутнього користувача, на основі свого досвіду задавати питання, спрямовані на виявлення неоднозначності, межових умов і потенційних сценаріїв помилок ще до початку фази розроблення. Це особливо важливо при роботі з непрофільними ІТ-замовниками, чиї вимоги часто бувають неповними, розпливчастими, двозначними та суперечливими. Застосування таких практик, як **"Три Аміго" (Three Amigos)**, де Product Owner (далі – продукт-овнер), (розробник і

тестувальник сумісно обговорюють користувацькі історії (User-Stories) і критерії приймання роботи), сприяє єдиному розумінню задачі і значно знижує кількість дефектів на пізніх етапах [16]. Аутсорсингова компанія має обґрунтовувати замовнику економічну доцільність ранніх інвестицій у якість, пояснюючи, що це дозволяє скоротити терміни та загальну вартість проєкту завдяки зменшенню кількості дефектів у майбутньому.

3.1.2. Проблема: Труднощі в ефективній комунікації та збиранні вимог до тестування через географічну, часову і культурну відмінності, посилені невизначеністю початкових бізнес-вимог замовника

Вплив: Неефективна комунікація призводить до неповних або невизначених вимог, що породжує помилки у реалізації, необхідність постійних переробок, затримок у проєкті та високі витрати. Це також знижує прозорість процесу та веде до недовіри між командами [5, 6].

Рекомендація: Побудова прозорого і структурованого комунікаційного мосту з фокусом на управління очікуваннями та ітераційне уточнення вимог. Необхідно встановити чіткі канали і графік комунікації, використовуючи спільні платформи для управління проєктами і вимогами (наприклад, Jira, Confluence) для фіксації всіх рішень. Тестувальники повинні проактивно відбирати вимоги, використовуючи візуальні артефакти (діаграми процесів, карти користувацьких історій [18], прототипи) для спрощення розуміння непрофільним замовником. Встановлення спільного глосарія термінів, проведення тренінгів із міжкультурної комунікації та застосування підходу ScrumOntoBDD, який поєднує Scrum, поведінково-орієнтовану розробку (BDD) та онтології для формалізації сценаріїв і приймальних критеріїв, сприяє усуненню непорозумінь та зменшенню неоднозначності у специфікаціях [5, 23]. В умовах постійної зміни вимог критично важлива **формалізація процесу управління змінами**, за якої кожна зміна документується, оцінюється її вплив і узгоджується з продакт-овнером [21].

3.1.3. Проблема: Проблеми оцінки готовності функціоналу до тестування та проведення його приймального тестування замовником за умов постійних змін вимог

Вплив: Ця проблема ускладнює своєчасне завершення ітерацій, призводить до «зависання» функціоналу у статусі «у процесі тестування/приймання», відтягує вихід продукту на ринок (або його нову версію) і знижує загальну прозорість процесів для замовника.

Рекомендація: Сумісно розробити та підтримувати гнучку практику "Definition of Done" та ітеративного процесу приймального тестування.

Сумісне розроблення і постійна актуалізація критеріїв "готовності" (DoD) для кожної користувацької історії з активною участю продакт-овнера – життєво важливі [19]. DoD повинно бути "живим" артефактом, що переглядається по мірі змінення вимог. Замість однієї демонстрації у кінці спринту, команда мусить прагнути до частіших, навіть неформальних, демонстрацій готового функціоналу, щоб отримати обернений зв’язок. Використання спільних дошок задач і діаграм згорання (Burndown chart) [24] підвищує прозорість прогресу, допомагає висвітити «блокери», рівномірність розподілу навантаження та ефективність планування [3]. Важливо заздалегідь узгодити процес "відкату" або зміни пріоритетів при

наявності динаміки вимог, а також навчати замовника принципам Agile для кращого розуміння його ролі в ітеративному прийманні [3].

3.1.4. Проблема: Проблема актуалізації тест-кейсів і тестової документації та їх підтримка у динамічному середовищі, що посилюється ймовірними відмінностями у стандартах документування та еволюції вимог замовника

Вплив: Призводить до застарілої документації, втраті тестового покриття, неможливості ефективного регресійного тестування, зниження швидкості і якості тестування, особливо із появою нових членів команди [3]. З практики: проблемою також може послужити недостатня якість самої тестової документації. Якщо тест-кейси написані недостатньо детально або мають певний «поріг входу» (в даному контексті саме знання самого продукту для ефективного проходження самого тесту, наприклад, під час регресійного тестування). Це, безпосередньо, тягне за собою зниження швидкості та якості тестування.

Рекомендація: Застосування адаптивних підходів до тестової документації й автоматизації, орієнтованих на "живу" актуальність і сумісне володіння. Замість об’ємних, швидко старіючих тест-кейсів, слід фокусуватися на більш полегшені формати: **BDD-сценарії**, які водночас є специфікацією і автоматизованим тестом [16], а також **тестові чартери** для дослідницького тестування і **ментальні карти** для високорівневого планування (дослідницького тестування в тому числі) [25]. Максимальне використання **автоматизації регресійних тестів** критично важливе, оскільки автоматизовані тести слугують "живою" документацією функціоналу [6, 10]. Необхідно узгоджувати із замовником необхідний рівень деталізації документації і використовувати системи управління тестуванням, інтегровані із системами управління проєктами для відслідковування і актуалізації [3]. Регулярні рефайнменти (англ. refinement) тестового беклогу також сприяють актуалізації тестових артефактів.

3.2. Проблеми середньої значимості та рекомендації щодо їх усунення

Ці проблеми можуть не блокувати проєкт повністю, але істотно знижують ефективність, збільшують операційні витрати і потенційно погіршують стосунки із замовником, якщо залишаються без уваги.

3.2.1. Робота з багами та їх пріоритизація у межах ітерації з урахуванням процесів узгодження із замовником

Вплив: Відсутність чіткого процесу призводить до хаотичного оброблення дефектів, невірної пріоритизації, затримок у виправленні критичних помилок, зниження загальної якості продукту та невдоволення замовника.

Рекомендації: Впровадження прозорого й автоматизованого процесу управління дефектами з чіткою матрицею пріоритетів і узгодженням зі стейкхолдерами. Необхідно використовувати єдину систему баг-трекінгу (наприклад, Jira) для обох сторін, а також сумісно розробити і узгодити чітку **матрицю пріоритетів і серйозності (Severity/Priority Matrix)** дефектів [13]. Регулярні короткі зустрічі (meeting) з продакт-овнером для рефайнменту дефектів і налаштування автоматичних повідомлень про зміну статусу дефектів підвищує прозорість. Важливо визначити чіткі критерії "готовності" (DoD) для багів, які гарантують їх повне усунення і підтвердження регресії [19].

3.2.2. Проблема: Недостатність часу для глибокого тестування у коротких спринтах при обмеженому доступі до систем або інформації замовника

Вплив: Призводить до поверхневого тестування, збільшення кількості дефектів у продукті, необхідності термінових виправлень і зниженню довіри замовника до якості кінцевого продукту та компетентності команди.

Рекомендація: Оптимізація тестового циклу, запит раннього доступу і проактивне управління тестовими оточеннями [26]. В умовах обмеженого часу необхідно застосовувати ризик-орієнтоване тестування, фокусуючись на найбільш критичному функціоналі та об’єктах. **Дослідницьке тестування (Exploratory Testing)** з документуванням у вигляді тестових чартерів ефективне для швидкого виявлення проблем у динамічному середовищі [9]. Аутсорсингова команда повинна проактивно запитувати ранній і повний доступ до тестових середовищ, баз даних, API-документації та експертам замовника. Важливо домовитися про створення репрезентативних тестових даних і, за можливістю, стандартизувати й автоматизувати розгортання тестових оточень для скорочення часу налаштування.

3.2.3. Проблема: Труднощі впровадження і підтримки автоматизації тестування (вибір інструментів, написання і підтримка тестів) через обмеження інфраструктури замовника або відсутність необхідних ліцензій/інструментів

Вплив: Збільшує витрати на ручне регресійне тестування, сповільнює цикли поставки версій, знижує швидкість отримання оберненого зв’язку і обмежує масштабованість QA-процесів.

Рекомендація: Стратегічне планування автоматизації, основане на загальній цінності для бізнесу і гнучкості інфраструктури. Критично важливо чітко обґрунтовувати замовнику **економічну вигоду автоматизації** у довготривалій перспективі, особливо для регресійного тестування (скорочення витрат, прискорення поставки, підвищення надійності) [6, 10]. На ранніх етапах проєкту необхідно узгоджувати стек інструментів для автоматизації, пропонуючи сумісні або альтернативні варіанти при наявності обмежень замовника. Впровадження автоматизації слід проводити поетапно, починаючи з найбільш критичної і стабільної функціональності (core flow) [6]. При інфраструктурних обмеженнях замовника слід пропонувати гнучкі рішення, наприклад, використання хмарних тестових середовищ або розроблення фреймворку, який мінімізує залежності від специфічних інструментів замовника. План передачі знань, необхідних для обґрунтування доцільності впровадження автоматизації також повинен бути частиною стратегії [3].

3.2.4. Проблема: Регресійне тестування та його обсяги

Вплив: Без ефективної стратегії регресії може проводитися надлишковий об’єм тестування, що тягне за собою затримки у релізах. Ручне виконання великого обсягу регресійних тестів потребує багато часу і ресурсів. Але, при наявності автоматизації, «прогін» надлишкового об’єму тестів займе кількість часу у межах допустимого відхилення. З практики: проходження, умовно, 50 тест-кейсів вручну може зайняти від 2 до 5 днів, в залежності від самих тестів, а саме їх складності чи комплексності. Прогін автоматизованих тестів такого ж об’єму – 30-60 хв.

Рекомендація: Оптимізація стратегії регресійного тестування через автоматизацію і розумну пріоритизацію. Пріоритетна **автоматизація регресійного тестування є** найбільш ефективним рішенням для Agile-середовища [6, 10]. Автоматизовані набори повинні запускатися регулярно як частина CI/CD конвеєра [7], якщо це є можливим. На додаток до повного автоматизованого набору, необхідно проводити **селективне регресійне тестування**, сфокусоване на областях, потенційно порушених новими змінами. Застосування практик **Test Impact Analysis** може допомогти визначити, які тести найбільш релевантні для запуска. Регресійне тестування повинно проводитися ітеративно, раз у певний проміжок часу, згідно з графіком релізів. З практики: доволі зручно, коли це реліз раз на спринт-два (в залежності від довжини спринта, згідно з Agile – 2-4 тижні). Нехай невеликий, але реліз. Це допомагає тримати певний темп (англ. cadence). В такому випадку, регресійне тестування буде проводитись, наприклад, раз на три тижні. Водночас, прогін автоматизованого смоук-с'юту (англ. smoke-suite) під час кожного мерджу коду дозволить виявити потенційні помилки одразу.

3.2.5. Проблема: Труднощі з приймальним тестуванням (Acceptance Testing) і звітністю перед замовником щодо якості

Вплив: Неузгодженість у критеріях приймання, відсутність чіткого розуміння замовником поточного рівня якості, труднощі прийняття рішень щодо релізу і потенційні конфлікти через нерозуміння реального стану проєкту.

Рекомендація: Розробка адаптивної системи метрик якості і прозорості звітності, орієнтованої на цінність для замовника. Необхідно сумісно із замовником визначити **ключові показники якості (KPI)**, які дійсно важливі для його бізнесу (наприклад, відсоток критичних дефектів у продакшені, задоволеність користувача (User Satisfaction / USAT), швидкість доставки цінності) [13]. Слід узгоджувати регулярні формати звітності про якість і прогрес, зрозумілі непрофільному замовнику (наприклад, дашборди з визуалізованими графіками). Регулярні короткі огляди метрик якості з продакт-овнером та іншими стейкхолдерами дає змогу обговорювати тенденції та ризики. Звітність повинна бути орієнтована на **бізнес-ризики**, а не просто на кількість дефектів. Використання **Service Level Agreements (SLA)** з вимірними метриками якості може формалізувати очікування.

3.3. Проблеми помірної значимості і рекомендації щодо їх усунення

Хоча ці проблеми менш критичні для безпосереднього функціонування продукту, їх вирішення сприяє створенню більш екологічного, гармонійного, продуктивного і стійкого середовища, підвищуючи мотивацію команди і задоволеність замовника у довготривалій перспективі.

3.3.1. Проблема: Формування реалістичних очікувань замовника відносно термінів і можливостей тестування, особливо при наявності жорстких дедлайнів і динаміці пріоритетів

Вплив: Нереалістичні очікування приводять до стійкого стану психічної незадоволеності один одним, як у замовника, так і у команди, потенційному вигоранню [27], зниження якості і можуть призвести до конфліктів у стосунках.

Рекомендація: Проактивне управління очікуваннями через

неперервну прозорість і раннє інформування про ризики. Аутсорсингова команда повинна постійно інформувати замовника про реальний стан справ, ризики, прогрес у тестуванні, та знайдені дефекти, що виникають, використовуючи дашборди з ключовими метриками. При виявленні потенційних проблем, що впливають на терміни або якість (наприклад, нестабільне оточення, нестача інформації, різка зміна вимог), необхідно терміново інформувати продакт-овнера для своєчасного прийняття рішень. Регулярні зустрічі з планування спринтів повинні містити обговорення реалістичності обсягу робіт із тестування [16]. Менеджери аутсорсингової компанії повинні делікатно, але чітко поясняти замовнику компроміси та ризики, пов’язані з жорсткими дедлайнами при високих вимогах до якості, навчаючи його концепції «швидко, дешево, якісно – вибери два» [12].

3.3.2. Проблема: Передача знань від замовника щодо продукту і бізнес-логіки всередині аутсорсингової команди

Вплив: Сповільнює онбордінг нових співробітників, призводить до втрати інституціональних знань, збільшує час на з’ясування інформації і може викликати помилки через недостатнє розуміння контексту.

Рекомендація: Систематизований підхід до підвищення кваліфікації співробітників і менторства. Необхідно створити централізовану базу знань (зокрема Wiki-like систему, як-от Confluence), яка буде містити всю важливу інформацію про продукт, бізнес-логіку, архітектуру, тестове оточення та інші дані [5]. Важливо проводити регулярні внутрішні воркшопи, демонстрації і семінари для нових членів команди. Організація менторства і парного тестування прискорює занурення у проєкт і сприяє більш глибокому розумінню функціоналу [6]. Також слід наполягати на можливості регулярної взаємодії з експертами із боку замовника (ключовими користувачами, бізнес-аналітиками та ін.) для уточнення деталей бізнес-логіки і отримання відповідей на питання [18].

3.3.3. Проблема: Мотивація та утримання тестувальників в умовах віддаленої роботи і потенційної відсутності прямого контакту з кінцевим користувачем

Вплив: Може призвести до зниження залученості, якості роботи, плинності кадрів, що, у свою чергу, веде до збільшення витрат на найм і навчання, а також сповільнює розвиток команди.

Рекомендація: Створення залучаючого середовища і підкреслення внеску тестувальників. Вкрай важливо регулярно демонструвати тестувальникам, як їх робота впливає на кінцевого користувача та успішність продукту, ділячись позитивними відгуками від клієнтів. Якщо є можливість, слід організовувати для тестувальників пряме спілкування з кінцевими користувачами або можливість бачити, як використовується продукт [6]. Інвестиції у професійний розвиток (тренінги, сертифікація, участь у конференціях тощо) показують цінність співробітників [10]. Регулярні тімбілдинги, онлайн-заходи і створення культури взаємодопомоги сприяє побудові сильних команд у середині аутсорсингової компанії. [6]. Також важливо чітко донести до тестувальників можливості їх професійного і кар’єрного зростання у компанії [3].

4. Обмеження дослідження і перспектива подальших досліджень

Дана праця, як комплексна і практико-орієнтована, має ряд обмежень. По-перше, висновки основані на якісному аналізі та узагальненні експертного (практичного) досвіду. а не на кількісних емпіричних даних або контрольованих експериментах. По-друге, хоч рекомендації універсальні, їх застосовність може залежати від специфіки конкретного проєкту, культурного складу і розміру команд.

У зв’язку з цим, перспективою подальших досліджень може бути:

1. **Емпірична валідація:** Проведення кількісних досліджень за участі аутсорсингових команд для підтвердження ефективності запропонованих рекомендацій на основі вимірних метрик (наприклад, зменшення кількості дефектів, скорочення часу циклу, підвищення NPS (Net Promoter Score) замовника тощо).
2. **Розроблення метрик для оцінки комунікації:** Створення та апробація метрик для оцінки якості та ефективності комунікації між розподіленими командами і непрофільними замовниками.
3. **Дослідження можливостей специфічних інструментів:** Більш глибоке дослідження впливу конкретних інструментів і платформ (наприклад, AI-інструментів для аналізу вимог або автоматизації тестування) на вирішення виявлених проблем.
4. **Культурологічний аспект:** Детальний аналіз впливу культурних відмінностей на процеси і тестування зокрема, в аутсорсингу і розроблення стратегій для їх подолання.
5. **Аналіз економічної ефективності:** Розробка моделей для кількісної точної оцінки економічної вигоди від впровадження кожної із запропонованих рекомендацій.

Висновки

Проведене дослідження було спрямоване на оптимізацію процесу тестування в Agile-середовищі для аутсорсингових компаній, працюючих з непрофільними замовниками, з метою підвищення якості, скорочення термінів виходу продукту на ринок і покращення взаємодії зі стейкхолдерами. В результаті всі поставлені задачі були вирішені і мета досягнута.

В ході дослідження були означені та упорядковані за рівнем їх значимості основні проблеми у процесі тестування аутсорсингових проєктів для непрофільних замовників. Ці проблеми виникають у таких областях, як комунікація, управління вимогами, оптимізація процесу тестування і побудова довірчих стосунків між командами розробників і стейкхолдерами.

Наукова новизна роботи полягає у комплексному дослідженні проблем тестування проєктів, які розробляються у взаємозв’язку Agile-методології, аутсорсингової компанії і непрофільними замовниками. Для усунення цих проблем запропонований набір рекомендацій, який являє собою синтез відомих практик (таких як "Shift-Left", BDD та/або TDD, автоматизація тестування та ін.), адаптований до використання в умовах розподілених команд і нетехнічного замовника. Зокрема, особлива увага приділена питанням проактивного управління очікуваннями і міжкультурної комунікації у процесі тестування, адаптивної документації і побудові прозорої системи комунікації, що критично важливо в умовах віддаленої роботи.

Практична значимість даної праці полягає у розробленні конкретних і

застосовуваних рекомендацій щодо підвищення ефективності QA-процесів аутсорсингових компаній. Впровадження запропонованих рекомендацій дає змогу:

– **Зменшити ризики** непорозуміння (у взаємодії замовник-виконавець) з вимогами і пізнього виявлення дефектів.

– **Підвищити якість** кінцевих програмних продуктів за рахунок більш раннього і всебічного тестування.

– **Скоротити терміни** виходу продукту на ринок завдяки оптимізації тестового циклу та ефективній роботі зі змінами.

– **Покращити взаємодію** із замовниками, забезпечивши прозорість, передбачуваність та їх задоволеність.

Реалізація цих рекомендацій вимагає сумісних зусиль як зі сторони аутсорсингової компанії, так і зі сторони замовника**.**

Список літератури

1. Beck K., Beedle M., van Bennekum A., Cockburn A., Cunningham W., Fowler, M. et al. Manifesto for Agile Software Development. AgileAlliance. – 2001. Retrieved from https://agilemanifesto.org/iso/en/manifesto.html

2. Brooks Jr., F. P. The Mythical Man-Month: Essays on Software Engineering (Anniversary ed.). Addison-Wesley Professional. – 1995. – 212 h. Retrieved from: https://web.eecs.umich.edu/~weimerw/2018-481/readings/mythical-man-month.pdf

3. Cohn M. Succeeding with Agile: Software Development Using Scrum. Addison-Wesley Professional. – 2009. – 51 p. Retrieved from: https://api.pageplace.de/preview/DT0400.9780321660510_A23552372/preview-9780321660510_A23552372.pdf

4. Crispin L., Gregory J. Agile Testing: A Practical Guide for Testers and Agile Teams. Addison-Wesley Professional. – 2009. – 112 p. Retrieved from: https://api.pageplace.de/preview/DT0400.9780134190648_A25762397/preview-9780134190648_A25762397.pdf

5. Crispin L., Gregory J. More Agile Testing: Learning Journeys for the Whole Team. Addison-Wesley Professional. – 2014. – 93 p. Retrieved from: https://ptgmedia.pearsoncmg.com/images/9780321967053/samplepages/9780321967053.pdf

6. DeMarco T., Lister T. Peopleware: Productive Projects and Teams. Dorset House Publishing. – 1987. – 261 p. Retrieved from: https://orion2020.org/archivo/articulos/00_peopleware.pdf

7. Eckstein J. Agile Software Development with Distributed Teams. Dorset House Publishing. – 2004. – 53 p. Retrieved from: https://ptgmedia.pearsoncmg.com/images/9780133491982/samplepages/0133491986.pdf

8. Fowler M. Patterns of Enterprise Application Architecture. Addison-Wesley Professional. – 2004. – 389 p. Retrieved from: https://ptgmedia.pearsoncmg.com/images/9780321127426/samplepages/9780321127426.pdf

9. Graham D., Fewster M. Experiences of Test Automation: Case Studies of Software Test Automation. Addison-Wesley Professional. – 2012. – 114 p. Retrieved from:
https://ptgmedia.pearsoncmg.com/images/9780321754066/samplepages/032175406

9.pdf

10. Humble J., Farley D. Continuous Delivery: Reliable Software Releases through Build, Test, and Deployment Automation. Addison-Wesley Professional. – 2010. – 497 p. Retrieved from: https://proweb.md/ftp/carti/Continuous-Delivery-Jez%20Humble-David-Farley.pdf

11. HubSpot. (n.d.). The Complete Guide to Shift–Left Testing [Whitepaper]. Retrieved from https://cdn2.hubspot.net/hubfs/3937956/whitepapers/The%20Complete%20Guide%20to%20Shift%20Left%20Testing.pdf

12. Kaner C., Bach J., Pettichord B. Lessons Learned in Software Testing: A Survival Guide for the Real World. Wiley. – 2002. – 30 p. Retrieved from: https://content.e-bookshelf.de/media/reading/L-592829-e29d79a396.pdf

13. Leffingwell D. Scaling Software Agility: Best Practices for Large Enterprises. Addison-Wesley Professional. – 2007. – 79 p. Retrieved from: https://pdfs.semanticscholar.org/ae8f/7fb9801a7d73ed5499ca9f5b579df4711bc4.pdf

14. Martin, R. C. Clean Code: A Handbook of Agile Software Craftsmanship. Prentice Hall. – 2008. – 464 p. Retrieved from: https://ptgmedia.pearsoncmg.com/images/9780132350884/samplepages/9780132350884.pdf

15. McConnell S. Code Complete. Microsoft Press. – 1993. – 952 p. Retrieved from: http://aroma.vn/web/wp-content/uploads/2016/11/code-complete-2nd-edition-v413hav.pdf

16. McConnell S. Rapid Development: Taming Wild Software Schedules. Microsoft Press. – 1996. – 672 p. Retrieved from: https://ptgmedia.pearsoncmg.com/images/9781556159008/samplepages/9781556159008.pdf

17. Naik K., Tripathy P. Software Testing and Quality Assurance: Theory and Practice. Wiley. – 2015. – 648 p. Retrieved from: https://www.softwaretestinggenius.com/download/staqtpsn.pdf

18. Patton J. User Story Mapping: Discover the Whole Story, Build the Right Product. O'Reilly Media. – 2014. – 324 p. Retrieved from: https://www.agileleanhouse.com/lib/lib/People/JeffPatton/StoryMapping_sampler.pdf

19. Schwaber K., Sutherland J. The Scrum Guide. Scrum.org & ScrumInc. – 2020. Retrieved from: https://scrumguides.org/

20. Smart J. F. BDD in Action: Behaviour-Driven Development for the Whole Team. Manning Publications. – 2015. – 384 p. Retrieved from https://kupichitay.com.ua/product/bdd-in-action-behavior-driven-development-for-the-whole-software-lifecycle-john-ferguson-smart/

21. Wiegers K., Beatty J. Software Requirements (3rd ed.). Microsoft Press. – 2013. – 673 p. Retrieved from https://olivroqueaprende.com/WDK/Software_Requirements_3rd_Edition.pdf

22. Sheikh S. Integrating Continuous Testing and Shift Left Practices in the Software Development Life Cycle for Enhanced Code Reliability. International Journal of Advanced Research in Management (IJARM). – 2024. – 15 (3). – P. 1-10 Retrieved from

23. Шостак, І., Череватенко, О, https://iaeme.com/MasterAdmin/Journal_uploads/IJARM/VOLUME_15_ISSUE_3/IJARM_15_03_009.pdf

24. Lopes de Souza P., Lopes de Souza W., Ferreira P. ScrumOntoBDD: Agile

software development based on scrum, ontologies and behaviour-driven development. Journal of the Brazilian Computer Society. – 2021. – v.27. – Art. 10. Retrieved from https://doi.org/10.1186/s13173-021-00114-w

25. Pham, K. P., & Neumann, M. How to Measure Performance in Agile Software Development? A Mixed-Method Study. 50th Euromicro Conference on Software Engineering and Advanced Applications (SEAA). IEEE. 2024. - Retrieved from DOI: 10.1109/SEAA64295.2024.00074

26. Government Digital Service. Exploratory testing — Service Manual. GOV.UK. – 2016. Retrieved from https://www.gov.uk/service-manual/technology/exploratory-testing

27. MoldStud. Best practices for test environment management in software testing. MoldStud. – 2024. Retrieved from https://moldstud.com/articles/p-best-practices-for-test-environment-management-in-software-testing

28. Trinkenreich, B. A Model for Understanding and Reducing Developer Burnout. SEIP. – 2023. Retrieved from https://biancatrink.github.io/files/papers/SEIP2023.pdf. Метод порівняльного аналізу попередніх оцінок і фактичних зусиль команди при розробленні програмних проектів в Agile-середовищі – Відкриті інформаційні та комп'ютерні інтегровані технології. 2025 – № 105. – С. 171-183. Retrieved from http://nti.khai.edu/ojs/index.php/oikit/article/view/oikit.2025.105.14

References

1. Beck K., Beedle M., van Bennekum A., Cockburn A., Cunningham W., Fowler, M. et al. Manifesto for Agile Software Development. AgileAlliance. – 2001. Retrieved from https://agilemanifesto.org/iso/en/manifesto.html

2. Brooks Jr., F. P. The Mythical Man-Month: Essays on Software Engineering (Anniversary ed.). Addison-Wesley Professional. – 1995. – 212 h. Retrieved from: https://web.eecs.umich.edu/~weimerw/2018-481/readings/mythical-man-month.pdf

3. Cohn M. Succeeding with Agile: Software Development Using Scrum. Addison-Wesley Professional. – 2009. – 51 p. Retrieved from: https://api.pageplace.de/preview/DT0400.9780321660510_A23552372/preview-9780321660510_A23552372.pdf

4. Crispin L., Gregory J. Agile Testing: A Practical Guide for Testers and Agile Teams. Addison-Wesley Professional. – 2009. – 112 p. Retrieved from: https://api.pageplace.de/preview/DT0400.9780134190648_A25762397/preview-9780134190648_A25762397.pdf

5. Crispin L., Gregory J. More Agile Testing: Learning Journeys for the Whole Team. Addison-Wesley Professional. – 2014. – 93 p. Retrieved from: https://ptgmedia.pearsoncmg.com/images/9780321967053/samplepages/9780321967053.pdf

6. DeMarco T., Lister T. Peopleware: Productive Projects and Teams. Dorset House Publishing. – 1987. – 261 p. Retrieved from: https://orion2020.org/archivo/articulos/00_peopleware.pdf

7. Eckstein J. Agile Software Development with Distributed Teams. Dorset House Publishing. – 2004. – 53 p. Retrieved from: https://ptgmedia.pearsoncmg.com/images/9780133491982/samplepages/0133491986.pdf

8. Fowler M. Patterns of Enterprise Application Architecture. Addison-Wesley Professional. – 2004. – 389 p. Retrieved from:

https://ptgmedia.pearsoncmg.com/images/9780321127426/samplepages/9780321127426.pdf

9. Graham D., Fewster M. Experiences of Test Automation: Case Studies of Software Test Automation. Addison-Wesley Professional. – 2012. – 114 p. Retrieved from: https://ptgmedia.pearsoncmg.com/images/9780321754066/samplepages/0321754069.pdf

10. Humble J., Farley D. Continuous Delivery: Reliable Software Releases through Build, Test, and Deployment Automation. Addison-Wesley Professional. – 2010. – 497 p. Retrieved from: https://proweb.md/ftp/carti/Continuous-Delivery-Jez%20Humble-David-Farley.pdf

11. HubSpot. (n.d.). The Complete Guide to Shift–Left Testing [Whitepaper]. Retrieved from https://cdn2.hubspot.net/hubfs/3937956/whitepapers/The%20Complete%20Guide%20to%20Shift%20Left%20Testing.pdf

12. Kaner C., Bach J., Pettichord B. Lessons Learned in Software Testing: A Survival Guide for the Real World. Wiley. – 2002. – 30 p. Retrieved from: https://content.e-bookshelf.de/media/reading/L-592829-e29d79a396.pdf

13. Leffingwell D. Scaling Software Agility: Best Practices for Large Enterprises. Addison-Wesley Professional. – 2007. – 79 p. Retrieved from: https://pdfs.semanticscholar.org/ae8f/7fb9801a7d73ed5499ca9f5b579df4711bc4.pdf

14. Martin, R. C. Clean Code: A Handbook of Agile Software Craftsmanship. Prentice Hall. – 2008. – 464 p. Retrieved from: https://ptgmedia.pearsoncmg.com/images/9780132350884/samplepages/9780132350884.pdf

15. McConnell S. Code Complete. Microsoft Press. – 1993. – 952 p. Retrieved from: http://aroma.vn/web/wp-content/uploads/2016/11/code-complete-2nd-edition-v413hav.pdf

16. McConnell S. Rapid Development: Taming Wild Software Schedules. Microsoft Press. – 1996. – 672 p. Retrieved from: https://ptgmedia.pearsoncmg.com/images/9781556159008/samplepages/9781556159008.pdf

17. Naik K., Tripathy P. Software Testing and Quality Assurance: Theory and Practice. Wiley. – 2015. – 648 p. Retrieved from: https://www.softwaretestinggenius.com/download/staqtpsn.pdf

18. Patton J. User Story Mapping: Discover the Whole Story, Build the Right Product. O'Reilly Media. – 2014. – 324 p. Retrieved from: https://www.agileleanhouse.com/lib/lib/People/JeffPatton/StoryMapping_sampler.pdf

19. Schwaber K., Sutherland J. The Scrum Guide. Scrum.org & ScrumInc. – 2020. Retrieved from: https://scrumguides.org/

20. Smart J. F. BDD in Action: Behaviour-Driven Development for the Whole Team. Manning Publications. – 2015. – 384 p. Retrieved from https://kupichitay.com.ua/product/bdd-in-action-behavior-driven-development-for-the-whole-software-lifecycle-john-ferguson-smart/

21. Wiegers K., Beatty J. Software Requirements (3rd ed.). Microsoft Press. – 2013. – 673 p. Retrieved from https://olivroqueaprende.com/WDK/Software_Requirements_3rd_Edition.pdf

22. Sheikh S. Integrating Continuous Testing and Shift Left Practices in the Software Development Life Cycle for Enhanced Code Reliability. International Journal

of Advanced Research in Management (IJARM). – 2024. – 15 (3). – P. 1-10 Retrieved from https://iaeme.com/MasterAdmin/Journal_uploads/IJARM/VOLUME_15_ISSUE_3/IJARM_15_03_009.pdf

23. Lopes de Souza P., Lopes de Souza W., Ferreira P. ScrumOntoBDD: Agile software development based on scrum, ontologies and behaviour-driven development. Journal of the Brazilian Computer Society. – 2021. – v.27. – Art. 10. Retrieved from https://doi.org/10.1186/s13173-021-00114-w

24. Pham, K. P., & Neumann, M. How to Measure Performance in Agile Software Development? A Mixed-Method Study. 50th Euromicro Conference on Software Engineering and Advanced Applications (SEAA). IEEE. 2024. - Retrieved from DOI: 10.1109/SEAA64295.2024.00074

25. Government Digital Service. *Exploratory testing — Service Manual*. GOV.UK. – 2016. Retrieved from https://www.gov.uk/service-manual/technology/exploratory-testing

26. MoldStud. Best practices for test environment management in software testing. MoldStud. – 2024. Retrieved from https://moldstud.com/articles/p-best-practices-for-test-environment-management-in-software-testing

27. Trinkenreich, B. A Model for Understanding and Reducing Developer Burnout. SEIP. – 2023. Retrieved from https://biancatrink.github.io/files/papers/SEIP2023.pdf

28. Shostak I. Cherevatenko O. Metod porivnialnoho analizu poperednikh otsinok i faktychnykh zusyl komandy pry rozroblenni prohramnykh proektiv v Agile-seredovyshchi [Method of comparative analysis of previous estimates and actual efforts of the team during the development of software projects in the Agile environment] – Vidkryti informatsiini ta kompiuterni intehrovani tekhnolohii. 2025 – № 105. – С. 171-183. Retrieved from http://nti.khai.edu/ojs/index.php/oikit/article/view/oikit.2025.105.14

Надійшла до редакцію 7.01.2026, розглянута на редколегії 10.02.2026

Analysis of the Problem of Testing in Software Project Development under Outsourcing Conditions for Non-Core Clients

The article is devoted to the problem of improving the efficiency of the testing process in the development of software projects within an Agile environment of outsourcing IT companies. The relevance of the study is determined by the current conditions of the IT services market and the strict requirements for the competitiveness of outsourcing companies. The object of the research is quality assurance processes of software projects implemented in the Agile environment of outsourcing companies. At present, Agile methodology and flexible frameworks (Scrum, Kanban, Lean, etc.) constitute a natural environment for the development of outsourcing software projects, and issues related to organizing production processes for projects of various domains (startups, SaaS, financial technologies, telemedicine, logistics, etc.) are well studied and supported by mature tooling (Jira, Azure, and others). At the same time, practical experience and retrospective studies indicate the presence of systemic trigger factors that lead to a significant decline in team performance, which is generally interpreted as the existence of specific problems typical for projects with distributed teams and non-core customers. Therefore, ensuring the efficiency of development processes in this

context remains an open issue. **The subject of the study** is the problems that arise during the testing process in outsourcing software projects and the methods for addressing them. **The purpose of the study** is, based on the analysis of practical experience, to identify and formulate specific testing problems in outsourcing projects developed for non-core customers that may cause a decrease in team efficiency, and, using established practices, to develop recommendations to eliminate or reduce the impact of these problems. To achieve this purpose, the study identified and systematized the main testing process issues, analyzed and classified the causes of their occurrence according to their significance, and proposed recommendations for their resolution. **The results obtained** indicate that, within a qualitative study based on the systematization of outsourcing company experience and analysis of typical scenarios, key problems faced by testing teams were identified, including insufficient tester involvement at early development stages, difficulties in communication and requirements elicitation, challenges in assessing feature readiness, test documentation maintenance, defect management, limited time and resources for testing, automation difficulties, regression testing scope, acceptance testing and reporting, shaping realistic customer expectations, knowledge transfer, and team motivation. The proposed recommendations include the active adoption of the Shift-Left approach and collaborative requirements definition, the establishment of transparent and structured stakeholder–developer communication, the development of flexible yet clear Definition of Done criteria and iterative acceptance testing processes, as well as the application of adaptive approaches to test documentation and automation. The recommendations also aim to improve defect management processes, test cycle planning efficiency, regression test automation, the development of an adaptive system of quality metrics and reporting, proactive management of customer expectations, and systematic knowledge sharing within teams.

Keywords: Agile, outsourcing, software project management, testing, quality assurance (QA), Shift-Left, Definition of Done (DoD).

Відомості про авторів

Шостак Ігор Володимирович – доктор технічних наук, професор кафедри інженерії програмного забезпечення, м. Харків, Національний аерокосмічний університет, Україна, shostak@gmail.com, +380982809755, ORCID:0000-0002-3051-0488.

Череватенко Олександр Вячеславович – аспірант кафедри інженерії програмного забезпечення, м. Харків, Національний аерокосмічний університет, Україна, appliesoftdev@gmail.com, +380503002811, ORCID:0000-0002-4982-5914

About the Authors

Igor SHOSTAK – Doctor of Technical Sciences, Professor of the Department of Software Engineering, Kharkiv, National Aerospace University, Ukraine, shostak@gmail.com, +380982809755, ORCID:0000-0002-3051-0488.

Oleksandr CHEREVATENKO – PhD student of the Department of Software Engineering, Kharkiv, National Aerospace University, Ukraine, Ukraine, appliesoftdev@gmail.com, +380503002811, ORCID:0000-0002-4982-5914