УДК 004.415.3:004.41
DOI https://doi.org/10.32689/maup.it.2025.2.28

Данило ТРОЩИЙ
аспірант кафедри інженерії програмного забезпечення,
Національний аерокосмічний університет «Харківський авіаційний інститут»,
d.troshchyi@khai.edu
***ORCID:** 0009-0008-3160-6060*

Юлія КУЗНЕЦОВА
кандидат технічних наук, доцент кафедри інженерії програмного забезпечення,
Національний аерокосмічний університет «Харківський авіаційний інститут»,
y.kuznetsova@khai.edu
***ORCID:** 0000-0001-9416-6981*

ОГЛЯД ТА АНАЛІЗ АСПЕКТІВ ПОВТОРЮВАНОСТІ РУТИННИХ ЗАВДАНЬ У ПРОЦЕСІ РОЗРОБЛЕННЯ МОБІЛЬНИХ ДОДАТКІВ

***Анотація. Метою дослідження** є системний аналіз рутинних повторюваних завдань, що виникають у процесі розроблення мобільних застосунків, а також визначення їхнього впливу на ефективність роботи розробницьких команд. Особлива увага приділяється виявленню підходів до зниження обсягу рутинних дій за рахунок автоматизації, стандартизації та повторного використання рішень.*

***Методологія.** У роботі застосовано комплексний підхід, що поєднує аналіз наукових публікацій, огляд галузевих звітів, а також систематизацію типових рутинних завдань на різних етапах життєвого циклу мобільного застосунку. Проведено порівняльний аналіз існуючих технічних рішень: інструментів автоматизованого тестування, генерації користувацького інтерфейсу, застосування практик DevOps і інтеграції віртуальних помічників.*

***Наукова новизна.** Запропоновано узагальнену класифікацію рутинних завдань мобільної розробки з урахуванням специфіки Android та iOS-платформ. Вперше комплексно проаналізовано прогалини у застосуванні сучасних інструментів автоматизації саме у мобільній сфері. Визначено, що більшість існуючих підходів не враховують особливості мобільних платформ або мають вузький технічний фокус.*

***Висновки.** Результати дослідження підтверджують актуальність необхідності уніфікації та автоматизації повторюваних завдань для підвищення ефективності мобільної розробки. Запропоновано перспективні напрями подальших досліджень, зокрема розроблення спеціалізованих інструментів, шаблонів та практик, адаптованих до викликів мобільної інженерії, що дозволить скоротити витрати часу та ресурсів, зменшити кількість помилок і прискорити вихід продукту на ринок*

***Ключові слова:** мобільний застосунок, розробка додатків, рутина, повторюваність, програмне забезпечення, автоматизація, життєвий цикл програмного забезпечення.*

Danylo TROSHCHYI, Yuliya KUZNETSOVA. REVIEW AND ANALYSIS OF REPETITIVE ROUTINE TASKS IN THE MOBILE APPLICATION DEVELOPMENT PROCESS

***Abstract. The purpose of the study** is to provide a systematic analysis of repetitive routine tasks in the development of mobile applications and to determine their impact on the efficiency of development teams. Special attention is paid to identifying approaches to reducing routine workload through automation, standardization, and reuse of solutions.*

***Methodology.** The study applies a comprehensive approach that combines analysis of scientific publications, review of industry reports, and systematization of typical routine tasks at different stages of the mobile application lifecycle. A comparative analysis of existing technical solutions is conducted, including automated testing tools, user interface generation, DevOps practices, and the integration of virtual assistants.*

***Scientific novelty.** A generalized classification of routine tasks in mobile development is proposed, taking into account the specifics of Android and iOS platforms. For the first time, existing gaps in the application of modern automation tools in the mobile domain are comprehensively analyzed. It is identified that most existing approaches either overlook platform-specific requirements or have a narrowly focused technical application.*

***Conclusions.** The research results confirm the relevance of unifying and automating repetitive tasks to improve the efficiency of mobile development. Promising directions for further research are proposed, including the development of specialized tools, templates, and practices tailored to the challenges of mobile engineering, which will reduce time and resource consumption, minimize errors, and accelerate product delivery to the market.*

***Key words:** mobile application, app development, routine tasks, repetition, software development, automation, software lifecycle.*

Вступ. У сучасному світі мобільних технологій розробка додатків стає все більш популярною та необхідною. З кожним днем зростає попит на якісні мобільні рішення, що спонукає розробників працювати над новими проєктами та вдосконалювати існуючі. Особливої популярності набувають mobile-first підходи, коли саме мобільна версія продукту стає основною точкою взаємодії з користувачем. Приклади таких продуктів – український застосунок Diia, який кардинально змінив сприйняття державних послуг у цифровому просторі, або BetterMe, що зумів вийти на глобальний ринок з рішеннями для фізичного та ментального здоров'я. Також варто згадати стартапи на кшталт Reface та Liki24, які демонструють, як мобільний застосунок може стати ядром продукту та драйвером бізнесу.

Успішність таких проєктів залежить не лише від інноваційної ідеї, а й від ефективності реалізації – і саме тут виявляється важливість системного підходу до розробки. У практиці мобільного розробника значна частина часу витрачається на вирішення типових задач, які з проєкту в проєкт мають подібну структуру й логіку. Йдеться про такі етапи, як створення архітектури проєкту, підключення бібліотек, організація взаємодії з API, тестування, підтримка тощо.

Постановка проблеми. Незважаючи на те, що рутинні завдання часто сприймаються як другорядні, їх повторюваність і трудомісткість можуть істотно впливати на загальну продуктивність команди розробників. Кожен проєкт починається з набору типових дій, які вимагають витрат часу і уваги, але при цьому не додають унікальної цінності продукту. Така ситуація стимулює пошук способів автоматизації, стандартизації та оптимізації цих процесів.

В останні роки у сфері мобільної розробки з'явилися численні інструменти та підходи, що дозволяють зменшити навантаження на розробника, скоротити час виходу продукту на ринок і підвищити якість кінцевого продукту. Проте їх застосування часто фрагментарне, а відсутність системного підходу до повторюваних задач призводить до дублювання зусиль та ускладнень при масштабуванні командної роботи.

У цьому контексті постає потреба у більш глибокому аналізі повторюваних завдань, властивих мобільній розробці, з метою подальшого формування рекомендацій щодо їх ефективної організації, оптимізації або автоматизації. Зокрема, важливо враховувати специфіку життєвого циклу мобільного застосунку – від формування вимог до підтримки продукту в пострелізному періоді.

Аналіз досліджень і публікацій. Розробка мобільних застосунків охоплює цілий спектр технічних і організаційних процесів, що повторюються з проєкту в проєкт. Такі задачі, попри свою технічну простоту, становлять основу щоденної роботи розробника і мають суттєвий вплив на загальну продуктивність команди. Незважаючи на свою буденність, ці задачі впливають на ефективність команди, якість коду, швидкість релізів і загальну стабільність продукту.

Рутинні завдання в мобільній розробці включають (але не обмежуються): налаштування середовища, підключення бібліотек, створення типових компонентів інтерфейсу, конфігурацію API, написання юніт- та UI-тестів, запуск тестів, аналіз логів, оновлення залежностей, складання збірки, запуск симуляторів або емуляторів, конфігурацію CI/CD, перевірку пул реквестів тощо. Часто саме ці дії вимагають повторення, не додаючи новизни, але залишаючись критично важливими для успішного запуску і підтримки застосунку.

Важливо відзначити, що характер рутинних задач значною мірою залежить від масштабу проєкту, використовуваних технологій та особливостей команди розробників. Наприклад, у невеликих проєктах частина цих завдань може бути мінімізована завдяки простоті структури, тоді як у масштабних корпоративних рішеннях складність і кількість рутинних процесів зростає в рази.

За деякими оцінками, на виконання рутинних задач мобільні розробники витрачають до 60% свого часу. Такий обсяг роботи часто недооцінюється, хоча саме від ефективності їх виконання залежить швидкість виведення продукту на ринок, якість коду і подальша підтримка [6, 7].

Аналіз публікацій у сфері програмної інженерії свідчить про наявність значної уваги до рутинних задач, особливо в контексті автоматизації, повторного використання коду, застосування шаблонів проєктування тощо.

Однак переважна частина таких досліджень стосується загальної розробки програмного забезпечення і недостатньо фокусуються саме на особливостях мобільної розробки, де специфіка платформ, залежність від магазинів застосунків і взаємодія з мобільними пристроями часто залишається поза межами уваги дослідників.

На практиці команди створюють власні бібліотеки, використовують локальні шаблони, формалізують окремі етапи CI/CD, але без єдиної уніфікованої методології такі дії залишаються точковими й погано масштабуються на інші проєкти або команди.

Таким чином, наявність повторюваних рутинних задач у мобільній розробці є системною проблемою, що напряму впливає на ефективність роботи, підтримку якості та швидкість оновлення продукту.

ISSN 2786-5460 (Print), ISSN 2786-5479 (Online)

Її ігнорування – один із бар'єрів для росту та зрілості команд, тому подальше дослідження цієї теми є вкрай актуальним.

Упродовж останніх років дослідницький інтерес до оптимізації процесів розроблення програмного забезпечення постійно зростає, що зумовлено як технологічним прогресом, так і підвищеними вимогами до продуктивності команд розробників. Особливу увагу в цьому контексті привертають саме повторювані, рутинні дії, які супроводжують проєктування, реалізацію, тестування й обслуговування мобільних застосунків.

Відповідно до звіту GitLab DevSecOps Report 2024 [1], значна частина часу фахівців витрачається не на створення нового функціоналу, а на підтримку інфраструктури, усунення технічного боргу, рев'ю коду, налаштування середовища, оновлення бібліотек тощо. У тому ж звіті зазначено, що 27% розробників вважають збільшення рівня автоматизації ключовим чинником, який міг би суттєво підвищити їхню задоволеність роботою, що ще раз підкреслює важливість зменшення обсягу повторюваних ручних дій у щоденних процесах мобільної розробки.

Gitlab та Jetbrains у свої дослідницьких роботах [1, 3] акцентують увагу на обмеженнях, що виникають внаслідок великого обсягу рутинної праці. Зокрема, дослідники підкреслюють, що повторюваність дій у мобільній розробці негативно впливає на стабільність, швидкість постачання оновлень і ментальне здоров'я інженерів. Проблема поглиблюється в умовах обмежених ресурсів, типової для невеликих стартапів або інді-команд.

В межах статті [10] автори досліджують вплив віртуальних персональних асистентів на ефективність виконання рутинних завдань. Зокрема, йдеться про використання Siri, Google Assistant та інших аналогів для автоматизації базових дій. Хоча підхід є цікавим, більшість кейсів орієнтовані на побутові або загальні офісні сценарії, а не на процеси, специфічні для мобільної розробки. Крім того, практичні аспекти інтеграції таких рішень у робоче середовище розробника залишаються недостатньо розкритими.

Робота [5] присвячена темі автоматизації DevOps-задач у процесі розробки програмного забезпечення. Автор окреслює базові принципи CI/CD, тестування та управління релізами, що безперечно стосується рутинних задач. Водночас, ця праця має досить загальний характер – більшість висновків стосуються лише традиційної бекенд-розробки та не враховуються особливості мобільних платформ. Крім того, приклади мають обмежену практичну цінність для тих, хто працює у сфері Android або iOS.

У дослідженні [9] розглядаються особливості впровадження CI/CD у мобільній розробці в умовах суворих регуляторних вимог. Автори аналізують виклики, пов'язані з дотриманням стандартів безпеки та конфіденційності, та пропонують стратегії для успішного впровадження CI/CD у таких середовищах. Однак, дослідження обмежується специфікою високорегульованих галузей, що може зменшити його застосовність у ширшому контексті мобільної розробки.

Публікація [11] представляє підхід MOBICAT, спрямований на автоматичну генерацію GUI-коду для Android-додатків за допомогою методів інженерії, керованої моделями. Цей підхід дозволяє зменшити ручну працю при створенні інтерфейсів користувача. Проте, дослідження зосереджене на специфічному аспекті розробки – генерації GUI-коду, що обмежує його застосовність у вирішенні ширшого спектру рутинних завдань у мобільній розробці.

У статті [8] аналізується застосування інструментів автоматизованого тестування, таких як Selenium, Appium і TestComplete, у проєктуванні та підтримці якісних мобільних застосунків. Автори розглядають ключові характеристики інструментів, зокрема їхню гнучкість та підтримку різних платформ. Незважаючи на корисний огляд, дослідження побудовано лише в межах тестування та не містить інформації щодо покращення інших типових задач під час розробки мобільного застосунку.

Інша публікація [4] також присвячена автоматизованому тестуванню. У ній автори описують специфіку вибору інструментів залежно від типу застосунку та бюджету команди. Стаття корисна для загального ознайомлення з підходами до автоматизації, але не акцентує увагу на мобільних розробниках як цільовій аудиторії. Крім того, більша частина прикладів пов'язана з веб-тестуванням, що зменшує релевантність у контексті мобільної інженерії.

Ще одне дослідження [2] зосереджене на автоматизації рутинних дій фронтенд-розробників. У статті запропоновано концепцію «єдиної платформи» – середовища, де об'єднано типові інструменти, що найчастіше використовуються під час повсякденної розробки. Автор демонструє можливість створення кастомізованого робочого середовища з підтримкою повторного використання компонентів і збору фідбеку для їх адаптивного оновлення. Платформа реалізована з урахуванням Lean Startup-моделі, що дозволяє швидко адаптувати рішення під потреби користувача. Водночас, попри цінність запропонованої архітектури, робота не охоплює проблеми мобільної розробки, зосереджуючись переважно на веб-контексті. Крім того, аналіз обмежується лише розробкою інтерфейсів, без розгляду супутніх рутинних процесів, таких як CI/CD, тестування чи збір аналітики.

Таким чином, більшість доступних публікацій або не приділяють достатньо уваги мобільній розробці, або ж досліджують продуктивність у загальному контексті, зосереджуючись на високорівневих підходах – архітектурі, тестуванні або застосуванні DevOps-практик. Інша частина робіт має вкрай вузьку спрямованість (наприклад, тестування) і стосується окремих технічних рішень, що не завжди масштабуються або є релевантними в довгостроковій перспективі. Це створює потребу в більш глибокому аналізі повторюваних задач саме в контексті мобільної розробки – з урахуванням її особливостей на різних етапах життєвого циклу програмного забезпечення.

Аналіз рутинних завдань під час розробки застосунку. Огляд рутинних задач доцільно здійснювати у зв'язку з етапами життєвого циклу програмного забезпечення. Це дозволить точніше визначити вузькі місця, що найменше автоматизовані.

На етапі збирання та уточнення вимог розробник часто стикається з повторюваними комунікаціями з менеджером або замовником, необхідністю формалізувати неструктуровану інформацію, адаптувати її до технічних обмежень платформи. Незважаючи на наявність систем управління вимогами, велика частка роботи тут залишається ручною.

Конфігурація проєкту на початку розробки також передбачає багато рутинної роботи: налаштування структури модулів, інтеграція SDK, налаштування CI/CD, тестових середовищ, схем збірки, signing configs тощо. Ці задачі часто повторюються від проєкту до проєкту з мінімальними відмінностями, однак потребують значного часу та уваги.

Під час проєктування інтерфейсу рутинними завданнями стають перенесення дизайн-макетів до коду, адаптація їх до різних екранів, відсутність узгоджених UI-компонентів або стилів. Ці завдання потребують багаторазових дрібних змін, перевірок і уточнень, що уповільнює роботу.

На етапі архітектурного проєктування розробник постійно повторює схожі дії: створення шаблонних елементів, підключення бібліотек, реалізація типових взаємодій із бекендом, написання однакових структур даних. Більшість команд не мають власних генераторів коду або DSL-інструментів, тому багато з цих процесів виконуються вручну.

Протягом реалізації функціоналу, типовими стають завдання інтеграції з API, обробка помилок, створення UI тощо. Ці задачі рідко є творчими, але вимагають значного часу та зусиль.

На етапі тестування розробник стикається з низкою рутинних дій, пов'язаних із забезпеченням якості продукту. Насамперед ідеться про написання юніт-тестів для перевірки логіки окремих компонентів та UI-тестів, які автоматично відтворюють сценарії взаємодії користувача з інтерфейсом. Окрім самого написання, важливо регулярно запускати ці тести, аналізувати результати, усувати причини можливих збоїв, а також підтримувати тести в актуальному стані в міру зміни функціоналу. Також слід враховувати, що навіть за наявності автоматизованих тестів, частина перевірок виконується вручну – однак ця ручна перевірка, як правило, відбувається ще на етапі розробки, коли інженер перевіряє працездатність нового функціоналу локально.

Що ж до фази релізу, то основна частина налаштувань вже має бути виконана раніше, на етапі конфігурації збірки та налаштування CI/CD-процесів. Втім, тут залишається чимало повторюваних завдань, пов'язаних із доналаштуванням параметрів релізної збірки, перевіркою цільових середовищ (наприклад, конфігурації build flavors), підготовкою скріншотів, метаданих, списків змін, а також запуском альфа- та бета-версій для обмеженого кола користувачів.

На етапі супроводу й обслуговування розробник витрачає час на перевірку аналітики, crash-репортів, відстеження проблем, які проявилися тільки в користувачів. Окрім того, виникає потреба в періодичному оновленні бібліотек, зміні політик платформ (наприклад, Google Play), які потребують швидкого реагування і спричиняють додаткові рутинні навантаження.

Окремо варто виділити роботу з фідбеком, як із боку кінцевих користувачів, так і замовників. Збір, інтерпретація, пріоритезація побажань і проблем це постійна діяльність, яка нерідко не має чіткої структури і здійснюється вручну через кілька джерел одночасно (відгуки в маркеті, email, месенджери, внутрішні трекери тощо).

Таким чином, майже кожен етап життєвого циклу мобільного застосунку супроводжується значним обсягом повторюваних завдань. Частина з них є технічно тривіальною, але вимагає витрат часу й уваги, відволікає від креативної або аналітичної діяльності, а в довгостроковій перспективі – знижує загальну ефективність команди.

Висновки. У ході аналізу було виявлено, що значна частина завдань, які щодня виконують мобільні розробники, має рутинний характер і часто залишається поза увагою дослідників. На основі систематизації процесів, характерних для всіх етапів життєвого циклу мобільного застосунку – від ініціалізації до підтримки та супроводу – можна зробити висновок, що існує потреба у переосмисленні підходів до оптимізації саме повторюваних дій. Попри широке впровадження засобів автоматизації, окремі сфери,

як-от локалізація чи фідбек-менеджмент, залишаються недостатньо дослідженими з точки зору їх впливу на загальну ефективність.

Поточне дослідження має аналітичний характер і не претендує на повне охоплення теми. Натомість його метою було окреслити карту повторюваних задач та виявити ділянки, що потребують подальшого вивчення. З огляду на це, перспективним напрямком наступного дослідження є поглиблений розгляд складнощів, які виникають під час вирішення типових рутинних завдань, а також систематизація існуючих підходів, їх переваг, обмежень та доцільності використання у контексті мобільної розробки.

Список використаних джерел:

1. GitLab Global DevSecOps Survey. URL: https://about.gitlab.com/developer-survey

2. Gupta S. Design of Controlled and Customizable Platform for Frontend Developer's Repetitive Tasks. *International Research Journal of Modernization in Engineering Technology and Science*. 2021. Vol. 3, Issue 11. C. 245–249. DOI: 10.13140/RG.2.2.17892.09604

3. JetBrains Developer Ecosystem Survey 2023. URL: https://www.jetbrains.com/lp/devecosystem-2023/

4. Muhammed Suhail TS. Automation: Deep Dive Into Web, Mobile & API – Part 1; *International Journal of Scientific and Research Publications (IJSRP)* 12(12) (ISSN: 2250-3153). 2022 DOI: 10.29322/IJSRP.12.12.2022.p13233

5. Nguyen Ba Long. Enterprise-grade CI/CD pipeline for mobile development: Bachelor's Thesis. – Metropolia University of Applied Sciences. 2022. URL:https://www.theseus.fi/bitstream/handle/10024/753156/Nguyen_Ba_Long.pdf

6. Repetitive Tasks at Work: Research and Statistics 2024. URL:https://www.processmaker.com/blog/repetitive-tasks-at-work-research-and-statistics-2024/

7. Time Spent on Recurring Tasks. Clockify Blog. URL: https://clockify.me/time-spent-on-recurring-tasks

8. Venkata Amarnath Rayudu Amisetty. AI-driven mobile and web automation: The CI/CD integration. revolution. *World Journal of Advanced Research and Reviews*, 2025, 26(02), 3554–3562. DOI: 10.30574/wjarr.2025.26.2.2039.

9. Vijay Bhasker Reddy Bhimanapati, Om Goel. Implementing CI/CD for Mobile Application Development in Highly Regulated Industries. *International Journal of Novel Research and Development*. 2023. DOI: 10.1234/jis.2020.0301

10. Wali, A., Mahamad, S., Sulaiman, S. Task Automation Intelligent Agents: A Review. Future Internet 2023, 15, 196. 2023. DOI: 10.3390/fi15060196

11. Zafar H, Ur Rehman Khan S, Mashkoor A and Nisa HU MOBICAT: a model-driven engineering approach for automatic GUI code generation for Android applications. Front. Comput. Sci. 2024 6:1397805. DOI: 10.3389/fcomp.2024.1397805

Дата надходження статті: 26.06.2025
Дата прийняття статті: 02.07.2025
Опубліковано: 23.09.2025