УДК 004.415.2

В. В. Никитюк, канд. тех. наук, доц., М. В. Галюк, М. В. Тененський
(Тернопільський національний технічний університет імені Івана Пулюя, Україна)

АНАЛІЗ ВИКОРИСТАННЯ ТА ПОРІВНЯННЯ МІКРОСЕРВІСНОЇ І МОНОЛІТНОЇ АРХІТЕКТУР

V. V. Nykytyuk, Ph.D., Assoc. Prof., M. V. Haliuk, M. V. Tenenskyi

ANALYSIS OF THE USE AND COMPARASION OF MICROSERVICE AND MONOLITHIC ARCHITECTURE

Сучасна розробка застосунків та систем складається з рядом складнощів, серед яких не останнє місце посідають вимоги до гнучкості, масштабованості та швидкості функціонування кінцевого продукту тощо. Дані показники на пряму залежить від типу обраної архітектури ПЗ чи застосунку. Так, прийнято виділяти так звану монолітну та мікросервісну архітектури. Так, мікросервісний підхід набув широкого поширення за останні п’ять років [1]. Натомість використання монолітного підходу, хоч він це все ще і вважається стандартною моделлю створення ПЗ, поступово йде на спад. Розглянемо детальніші дані архітектурні підходи та визначимо їх ключові переваги та недоліки.

Так, відповідно до монолітної архітектури, застосунок будується як один великий самодостатній функціональних блок, не залежний від інших застосунків (див. рис. 1) [2]. Фактично це означає наявність великої обчислювальної мережі з єдиною кодовою базою, яка об’єднує виконання усіх бізнес задач. Серед переваг такого підходу можна виділити високу швидкість розробки завдяки простоті створення ПЗ на основі однієї кодової бази. Варто відмітити і простоту первинного розгортання монолітного застосунку. Проте для внесення змін до нього необхідно оновити увесь стек, отримавши перед цим доступ до бази коду, що робить оновлення певною мірою обмеженими та трудомісткими [3].

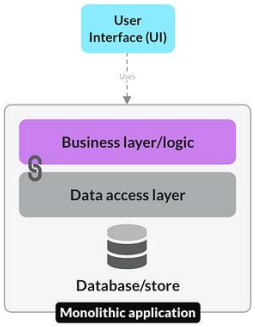

Рисунок 1. Схематичне зображення структури монолітного застосунку

Моноліти можуть бути дуже ефективними, допоки не посте питання щодо їх масштабування чи розширення функціональних можливостей. Так, внесення навіть найменшої зміни вимагатимете повне тестування усієї системи, що на пряму суперечить сучасному гнучкому підходу до розробки застосунків. Пов’язано це із високим рівнем взаємозв’язків між внутрішніми модулями системи, що часто створює багато технічних проблем. З цього випливає низька надійність монолітних систем та обмеження до впровадження використання нових технологічних засобів, через що ускладнюється їх підтримка. Ось чому для оновлення технологічного стеку чи додання комплексної бізнес логіки монолітні системи не рідко переписують ледь не з нуля [3].

Мікросервісна архітектура в певній мірі є альтернативою монолітній. В основі даного підходу лежить використання ряду незалежних один від одного сервісів, які розгортаються окремо. Також такі сервіси обов'язково виконують різні задачі та мають власні бази даних (див. рис. 2) [2]. Інакше кажучи, мікросервіси розділяють виконання бізнес логіки на окремі, незалежні бази коду. Такий підхід дещо збільшує складність розробки, проте значно полегшує вирішення різноманітних технічних проблем

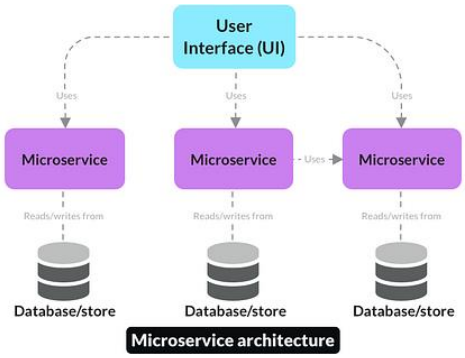

Рисунок 2. Схематичне зображення мікросервісного застосунку

Так, мікросервісний підхід дозволяє проводити оновлення, тестування, розгортання та масштабування для кожного сервісу окремо, що дозволяє швидко адаптуватися до вимог користувачів чи замовника. Окрім того, можна динамічно регулювати ресурси для конкретних мікросервісів в залежності від їх навантаження, що покращує загальну ефективність та продуктивність системи [1]. Проте з переваг мікросервісної архітектури випливають і її недоліки: довша та дорожча розробка ПЗ; складність до підтримки узгодженості і цілісності даних через велику кількість баз даних та комунікаціям між мікросервісами; високі витрати на розгортання сервісів; відсутність стандартизації або її недостатній рівень.

Підсумовуючи все вищезгадане в процесі дослідження та порівняння мікросервісної та монолітної архітектур можна зробити висновок, що монолітні застосунки можуть бути зручними на ранніх етапах свого життєвого циклу. Так, за відсутності потреби до масштабування, перехід до мікросервісної архітектури є просто зайвим та лише створить зайві витрати. І хоча даний підхід стає все менш популярним, він має свої сильні сторони. Так, якщо метою є створення простого та легкого застосунку, необхідність впровадження мікросервісів відпадає сама собою. В свою чергу, мікросервісний підхід буде більш кращим рішенням для застосунків чи систем, які охоплюють велику кількість користувачів та мають розгалужену і складну бізнес логіку. Такі системи легко підтримувати, оновлювати та масштабувати. Ось чому варто завжди подумати двічі щодо вибору між монолітною та мікросервісною архітектурами.

Література

1. Harris C. Microservices vs. monolithic architecture. URL: https://www.atlassian.com/microservices/microservices-architecture/microservices-vs-monolith
2. IcePanel. Monolithic vs. microservices architectures. *Medium*. URL: https://icepanel.medium.com/monolithic-vs-microservices-architectures-e71c75b252d1
3. From Monolithic Systems to Microservices: A Comparative Study of Performance / F. Tapia, M. Angel Mora, W. Fuertes, H. Aules et al. *Appl. Sci.* 2020, 10(17), 5797. DOI: https://doi.org/10.3390/app10175797