УДК 004.31

О.Д. Азаров

М.Г. Тарновський

Д. Т. Гріша

ЗАСОБИ МІКРОСЕРВІСНОЇ АРХІТЕКТУРИ В РОЗРОБЦІ СУЧАСНИХ ВЕБДОДАТКІВ

Вінницький національний технічний університет

Анотація

Огляд сучасних підходів до проєктування та впровадження вебдодатків на основі мікросервісної архітектури, що забезпечує високу масштабованість, надійність та гнучкість програмних систем. Дослідження принципів розподілу функціональності між незалежними сервісами, які взаємодіють за допомогою мережевих протоколів та інструментів оркестрації. Аналіз технологій і фреймворків, що сприяють ефективній реалізації мікросервісних рішень. Розглянуто методи забезпечення безпеки, моніторингу та безперервного розгортання мікросервісів у хмарних середовищах. Результати дослідження можуть бути використані при розробці масштабованих корпоративних вебдодатків, інформаційних порталів і сервісів електронної комерції.

Ключові слова: Мікросервісна архітектура, сервіси, сервер, вебдодатки, масштабованість, хмарні технології, розгортання сервісів, безпека, моніторинг, метрика, мікромодульність, екосистема.

Abstract

Review of modern approaches to the design and implementation of web applications based on microservice architecture, which provides high scalability, reliability and flexibility of software systems. Research on the principles of distributing functionality between independent services that interact using network protocols and orchestration tools. Analysis of technologies and frameworks that contribute to the effective implementation of microservice solutions. Methods for ensuring security, monitoring and continuous deployment of microservices in cloud environments are considered. The results of the study can be used in the development of scalable corporate web applications, information portals and e-commerce services.

Keywords: Microservice architecture, services, server, web applications, scalability, cloud technologies, service deployment, security, monitoring, metrics, micromodularity, ecosystem.

Вступ

У сучасному світі мережа Інтернет стала невід'ємною частиною більшості людей. За допомогою такої технології можливо зробити будь яку операцію – від купівлі товару до отримання коштів. Щоденно сотні мільйонів людей використовують Інтернет з тією чи іншою метою.

Але сама по собі мережа Інтернет не дає таких можливостей, вона є певним простором для цього. Для здійснення різних повсякденних дій, спеціалістами зі сфери інформаційних технологій були розроблені різного роду вебдодатки, які допомагають людям в повсякденних задачах.

Вебдодатки – це програмне забезпечення, яке працює у веб-браузері та доступне через Інтернет, надаючи користувачам інтерактивні функції для виконання завдань, а не лише відображаючи інформацію, як веб-сайт. Завдяки такій програмі, користувач має змогу, наприклад, замовляти товари з магазину, чи спілкуватися з колегами по роботі.

Оскільки вебдодатки є досить популярним інструментом для взаємодії між людиною та сервісами, вивчення та аналіз методів їх розробки є доволі важливим етапом в розвитку сфери інформаційних технологій. Для проєктування подібного програмного забезпечення існує достатньо різноманітний перелік технологій та методів, які дозволяють ефективно створити той чи інший програмний продукт. Серед таких методів є мікросервісний архітектурний стиль або мікросервісна архітектура. Цей метод розробки є доволі популярним та все більше набирає поширення серед розробників у сфері інформаційних технологій.

Результат дослідження

Більшість вебдодатків мають доволі широкий функціонал та можливості, що забезпечують зручність та багатозадачність для найрізноманітніших користувачів. Такі ознаки потребують багато зусиль зі сторони розробників програмного забезпечення, що виливається в доволі масштабну кодову базу. Щоб забезпечити правильний менеджмент, розроблювальний проєкт потрібно правильно розробити структуру програми. Саме тут використовуються архітектурні стилі.

Побудова архітектури додатку — це комплекс методів, який спрямований на чітке визначення побудови система. Це визначає як різні компоненти програмного забезпечення взаємодіятимуть один з одним, і відіграє ключову роль у його продуктивності, масштабованості та зручності обслуговування.

Архітектурні стилі вебдодатків включають три основні типи верхньорівневої архітектури: моноліт, мікросервіси та серверлес, що відрізняються за способом структурування та розгортання, а також різні архітектурні патерни, такі як MVC, MVP та MVVM, які пропонують готові рішення для організації компонентів застосунку та бізнес-логік [1].

Монолітний стиль архітектури передбачає створення цілісного застосунку, в якому усі функціональні компоненти взаємопов’язані та працюють у межах одного проєкту. У такій архітектурі бізнес-логіка, користувацький інтерфейс, модулі роботи з базою даних утворюють цілу структуру, що забезпечує узгодженість процесів і спрощує розробку. Завдяки своїй цілісності монолітні додатки легше створювати на початкових фазах розробки, оскільки вони не потребують складних механізмів взаємодії між компонентами.

Однією з ключових характеристик серверлес є модель автоматичного масштабування, за якої кількість ресурсів динамічно змінюється залежно від фактичного навантаження на застосунок. Це означає, що система здатна ефективно обробляти як невелику кількість запитів, так і пікові навантаження без потреби у попередньому плануванні чи ручному розширенні інфраструктури. Такий підхід позитивно впливає на оптимізацію витрат, оскільки замовник сплачує лише за фактичне використання ресурсів, що робить архітектуру економічно привабливою у порівнянні з традиційними підходами [2].

Мікросервісна архітектура є одним із найпоширеніших сучасних підходів до проєктування та розробки програмних систем, що ґрунтується на ідеї поділу застосунку на набір відносно невеликих, автономних сервісів, кожен з яких виконує чітко визначену бізнес-функцію. Кожен мікросервіс розробляється, тестується, розгортається та масштабується незалежно від інших, що забезпечує високу гнучкість системи та значно підвищує її стійкість до змін.

Завдяки своїй структурі мікросервісна архітектура значно підвищує надійність застосунку. Вихід з ладу одного сервісу, як правило, не призводить до зупинки всієї системи, оскільки інші її компоненти продовжують функціонувати. Це дозволяє створювати відмовостійкі рішення, що актуально для критично важливих сфер.

Разом з тим, впровадження мікросервісної архітектури супроводжується і певними недоліками. Розподілення системи призводить до зростання складності управління нею. Забезпечення надійності комунікацій, узгодженості даних та цілісності транзакцій потребує використання спеціалізованих технологій і ретельного планування. Ще одною проблемою є централізоване спостереження та моніторинг за сервісами, оскільки в такій архітектурі потрібно слідкувати за помилками та миттєво вирішувати їх. Важливою задачею є організація безпеки, адже збільшення кількості точок взаємодії підвищує ризик потенційних атак [3].

Важливим етапом розробки додатків, що використовують мікросервісну архітектуру є управління даними. На відміну від монолітного архітектурного стилю, де база даних зазвичай є частиною самої програми, у мікросервісній архітектурі застосовується принцип незалежності збереження даних. Кожен сервіс має власне сховище, яке відповідає його функціональним завданням і може реалізовуватися за допомогою різних типів баз даних відповідно до характеру даних і вимог до швидкості обробки. Важливим аспектом є узгодження транзакцій та забезпечення цілісності інформації в умовах децентралізованої структури [4].

Далі формується технічна основа майбутнього середовища розгортання. На цьому етапі розглядаються процес контейнеризації та оркестрації, що є невід'ємною частиною для мікросервісних систем. Використання контейнерів дозволяє забезпечити уніфіковане середовище виконання незалежно від інфраструктурних відмінностей. Тут також враховуються механізми спостереження, моніторингу та логування, які дозволяють оперативно реагувати на зміни стану системи та своєчасно усувати потенційні проблеми.

Наступним етапом проєктування мікросервісного додатку є створення механізму безпеки. Автентифікація, авторизація та захист каналів комунікації становлять ключові складові, які гарантують конфіденційність даних і запобігають несанкціонованому доступу. У мікросервісних додатках механізм безпеки реалізується на рівні окремих сервісів або на рівні шлюзу [5].

Висновки

Було проведено огляд сучасних підходів до проєктування та впровадження вебдодатків на основі мікросервісної архітектури. Досліджено принципи розподілу функціональності між незалежними сервісами. Здійснено аналіз технологій і фреймворків, що сприяють ефективній реалізації мікросервісних рішень. Розглянуто методи забезпечення безпеки, моніторингу та безперервного розгортання мікросервісів у хмарних середовищах.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУИ

1. Gamma E., Helm R., Johnson R., Vlissides J. *Design Patterns: Elements of Reusable Object-Oriented Software.* Reading, Massachusetts: Addison-Wesley, 1994. 395 p.

2. Buschmann F., Meunier R., Rohnert H., Sommerlad P., Stal M. *Pattern-Oriented Software Architecture. Volume 1: A System of Patterns.* Chichester: John Wiley & Sons, 1996. 476 p.

3. Мартін Р. Чиста архітектура. Мистецтво розробки програмного забезпечення. Харків: Віват, 2020. 384 с.

4. Cervantes H., Kazman R. *Designing Software Architectures: A Practical Approach.* Boston: Addison-Wesley, 2016. 368 p.

5. Kruchten P. *Architectural Blueprints: The 4 + 1 View Model of Software Architecture.* IEEE Software, Vol. 12, No. 6, 1995. pp. 42–50. URL: https://arxiv.org/abs/2006.04975

6. Tamzalit D., Mens T., et al. *Evolution Patterns: Designing and Reusing Architectural Evolution Knowledge to Introduce Architectural Styles.* arXiv preprint arXiv:1605.06289, 2016. URL: https://arxiv.org/abs/1605.06289

7. Гавриляк А. В., Ліщук К. І. Архітектурні шаблони та стилі програмного забезпечення. *Електронний архів НТУУ «КПІ ім. Ігоря Сікорського».* URL: https://ela.kpi.ua/items/a31e0e38-438a-42ad-83ad-34aae2ca40fd

8. Любченко В. В. Архітектура програмного забезпечення та її якісні атрибути. *Проблеми програмування.* Київ: Інститут програмних систем НАН України, 2020. URL: https://pp.isofts.kiev.ua/index.php/ojs1/article/download/625/677

Азаров Олексій Дмитрович – докт. техн. наук, професор, завідувач кафедри обчислювальної техніки, Вінницький національний технічний університет, м. Вінниця.

Тарновський Микола Генадійович – канд. техн. наук, дипломований доцент, Вінницький національний технічний університет, Вінниця.

Гріша Даніл Тарасович – студент групи 1КІ-24м, факультету інформаційних технологій та комп'ютерної інженерії, Вінницький національний технічний університет, Вінниця, e-mail: grishadanil0@gmail.com.

Azarov Oleksiy Dmytrovych – doctor tech sciences, professor, head of the Computer Techniques Chair, Vinnytsia National Technical University, Vinnytsia.

Tarnovsky Mykola Genadiyovych – candidate tech sciences, associate professor, Vinnytsia National Technical University,Vinnytsia.

Hrisha Danil Tarasovych – student of 1KI-24m group, Faculty of Information Technologies and Computer Engineering, Vinnytsia National Technical University, Vinnytsia, e-mail: grishadanil0@gmail.com