УДК 004.65

І. С. Зоря
Ю. Ю. Козюк

АНАЛІЗ ТА ОЦІНКА СТРАТЕГІЙ ВИБОРУ БАЗ ДАНИХ ДЛЯ МІКРОСЕРВІСНИХ АРХІТЕКТУР: ПОРІВНЯЛЬНА ХАРАКТЕРИСТИКА POLYGLOT PERSISTENCE ТА ЄДИНОГО СХОВИЩА

Вінницький національний технічний університет

Анотація. *Дана робота присвячена аналізу стратегій вибору та методів організації баз даних у мікросервісних архітектурах, зокрема порівнянню концепцій Polyglot Persistence та єдиного сховища для забезпечення ефективного управління інформаційними ресурсами.*
Ключові слова: *мікросервісна архітектура; база даних; Polyglot Persistence; PostgreSQL; MongoDB; Redis; CQRS; консистентність даних; масштабованість; цілісність інформації.*

Abstract. *This paper is devoted to the analysis of database selection strategies and organization methods in microservice architectures, specifically comparing the concepts of Polyglot Persistence and a single data store to ensure effective management of information resources.*
Keywords: *microservice architecture; database; Polyglot Persistence; PostgreSQL; MongoDB; Redis; CQRS; data consistency; scalability; data integrity.*

Вступ

Сучасні програмні системи характеризуються переходом від монолітних структур до розподілених мікросервісних архітектур, що дозволяє забезпечити високу масштабованість та гнучкість розробки. Кожен окремий компонент такої системи виконує специфічну бізнес-функцію, що висуває унікальні вимоги до методів обробки та зберігання інформації. Питання ефективної організації даних у розподілених середовищах набуває особливого значення. Зміни у парадигмі проектування, зокрема впровадження концепції автономності сервісів, сприяли переходу від використання єдиного сховища до стратегії Polyglot Persistence. Цей підхід передбачає використання різних типів баз даних (реляційних, документних, графових тощо) залежно від специфіки завдань конкретного мікросервісу [1]. Постійні вимоги до відмовостійкості, швидкодії та консистентності даних у високонавантажених системах призвели до необхідності впровадження нових архітектурних рішень для управління розподіленими даними та забезпечення їх цілісності.

Результати дослідження

Враховуючи мету даного дослідження, варто проаналізувати основні переваги та недоліки впровадження стратегії Polyglot Persistence у мікросервісних системах. Зокрема, до переваг можна віднести:

– оптимізацію продуктивності під конкретні типи запитів;
– гнучкість у виборі моделей даних, незалежне масштабування сховищ;
– відсутність жорсткої залежності від одного постачальника ПЗ.

До недоліків Polyglot Persistence варто віднести:

– складність забезпечення цілісності даних між різними базами;
– необхідність підтримки ширшого стеку технологій командою розробників;
– збільшення витрат на хмарну інфраструктуру.
– складність централізованого моніторингу.

Незважаючи на ці недоліки, концепція вибору спеціалізованих баз даних залишається ключовим інструментом для побудови високонавантажених та відмовостійких систем, особливо в умовах інтенсивного зростання обсягів неструктурованих даних [2].

Використання реляційних систем керування базами даних (СУБД), таких як PostgreSQL – це перевірений метод забезпечення надійності для сервісів, що потребують суворої консистентності та складних транзакційних зв’язків.

PostgreSQL є потужним, розширюваним та безпечним механізмом, що базується на принципах ACID та підтримці складних SQL-запитів. Ця СУБД є доцільним варіантом для використання в мікросервісах, відповідальних за фінансові операції або управління профілями користувачів, оскільки вона мінімізує ризики втрати цілісності даних. Водночас, для сервісів обміну повідомленнями або кешування результатів часто впроваджуються NoSQL-рішення, такі як MongoDB або Redis, що забезпечують горизонтальне масштабування та високу швидкість обробки великої кількості записів у не приватних розподілених середовищах [3].

Зберігання даних з використанням комбінації реляційних баз (PostgreSQL) та документоорієнтованих систем (MongoDB) – це спосіб оптимізації обробки інформації в мікросервісах, де необхідно поєднувати сувору структуру та гнучкість. PostgreSQL використовується для забезпечення цілісності транзакційних даних, тоді як MongoDB застосовується для пакування неструктурованих даних (наприклад, логів або медіа-метаданих) у гнучкі документи і швидкої їх видачі без необхідності складних перетворень. У випадку використання даного методу реляційна модель створює надійний фундамент для критично важливих операцій. В той час як документоорієнтована модель забезпечує горизонтальне масштабування та високу швидкість запису великих масивів даних. Такий підхід дає можливість сформувати ефективний баланс між конфіденційністю, цілісністю транзакцій та продуктивністю при роботі з динамічним контентом у розподілених вузлах системи.

Поєднання кешування в пам’яті (In-memory storage) на базі Redis та основного сховища дозволяє пришвидшити отримання доступу до даних у мікросервісній мережі через загальнодоступні або внутрішні канали зв'язку.

Redis використовується для створення тимчасових логічних з’єднань між часто запитуваними ресурсами та кінцевим користувачем. Використання підхід дозволяє передавати дані практично миттєво. Основна база даних використовується для довготривалого зберігання, а Redis – для шифрування сесій або тимчасового захисту тунельних з’єднань шляхом швидкої перевірки токенів доступу. Такий підхід широко використовується для створення захищених з’єднань у високонавантажених сервісах, забезпечуючи мінімальні затримки та безпечний доступ до мережевих ресурсів з будь-якої точки. Це ефективно захищає систему від перевантажень і забезпечує стабільну роботу інфраструктури при різких стрибках трафіку [4].

Дослідження також показало, що ефективним методом забезпечення цілісності даних у розподілених середовищах є впровадження архітектурного патерну CQRS (Command Query Responsibility Segregation) у поєднанні з подієво-орієнтованим підходом. У цій моделі одна база даних (наприклад, реляційна) використовується виключно для операцій запису та зберігання «джерела істини», тоді як інша (наприклад, NoSQL або ElasticSearch) – для швидкого читання та складного пошуку. Такий підхід забезпечує прозорість інфраструктури та дозволяє створювати безпечні з’єднання між сервісами, мінімізуючи ризики блокування таблиць при масових запитах. Використання цього методу дозволяє підтримувати високу доступність внутрішніх ресурсів мережі навіть за умов інтенсивної синхронізації великих масивів інформації [5].

Крім того, аналіз стратегій управління даними підкреслює важливість використання API Gateway для забезпечення єдиної точки входу до різних типів сховищ. API Gateway виконує роль логічного посередника, який маршрутизує запити до відповідних мікросервісів, забезпечуючи при цьому додатковий рівень автентифікації та шифрування трафіку. Це дозволяє приховати складність архітектури Polyglot Persistence від кінцевого користувача, надаючи йому прозорий доступ до даних через стандартні інтерфейси. Таке поєднання технологій ефективно захищає мережевий трафік від несанкціонованого доступу та забезпечує гнучке управління правами доступу до конкретних сегментів інформаційної системи, що є критично важливим для забезпечення загальної кібербезпеки корпоративної мережі [6].

Оцінка методів організації даних у роботі здійснювалася за допомогою рейтингової шкали показників. Результати подані у таблиці 1.

Таблиця 1 – Порівняльний аналіз сучасних стратегій та типів баз даних

Метод / Тип БД	Швидкодія (Latency)	Рівень консистентності	Складність підтримки	Гнучкість масштабування
PostgreSQL (Shared)	Середня	Дуже високий	Низька	Низька
Polyglot (Postgres+Mongo)	Висока	Середній	Висока	Висока
Redis (In-memory)	Надзвичайно висока	Низький (Eventual)	Середня	Висока
Event Sourcing / CQRS	Висока	Середній	Дуже висока	Дуже висока
MongoDB (Standalone)	Висока	Середній	Низька	Висока

Тож згідно із аналізу порівняльної таблиці, можна зробити висновок, що Polyglot Persistence (на прикладі зв'язки Postgres та MongoDB) виділяється як метод з високою гнучкістю масштабування та оптимізованою швидкодією, що робить його привабливим вибором для складних мікросервісних систем. Проте використання єдиного реляційного сховища (PostgreSQL) все ще вважається найвищим рівнем консистентності даних та низькою складністю підтримки, що робить його привабливим варіантом для проектів з критичними вимогами до цілісності транзакцій та обмеженими ресурсами на адміністрування інфраструктури.

Висновки

Сучасні мікросервісні архітектури вимагають надійних і гнучких методів управління даними. Ключовим фактором, який гарантує продуктивність та автономність окремих компонентів системи під час їх взаємодії в розподіленій мережі, є вибір правильної стратегії організації сховищ.

В даній роботі було проаналізовано сучасні підходи до організації баз даних. Зокрема, досліджено концепцію Polyglot Persistence (на прикладі зв'язки PostgreSQL, MongoDB та Redis), стратегію єдиного сховища (Shared Database) та патерни розділення відповідальності (CQRS). Кожен з цих методів підходить для різних сценаріїв розробки. Проте, найбільш збалансованим для високонавантажених систем виявився підхід Polyglot Persistence. Він дозволяє поєднати сувору консистентність реляційних моделей із високою швидкістю обробки неструктурованих даних, забезпечуючи при цьому необхідний рівень ізоляції та безпеки інформаційних ресурсів.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Newman S. Building Microservices: Designing Fine-Grained Systems. – O'Reilly Media, 2021. – 616 p. (Дата звернення 04.05.2026).
2. Fowler M. Patterns of Enterprise Application Architecture. – Addison-Wesley, 2002. – 533 p.
3. PostgreSQL Documentation. – [Електронний ресурс]. – Режим доступу: https://www.postgresql.org/docs/ (Дата звернення 04.05.2026).
4. Redis Documentation. – [Електронний ресурс]. – Режим доступу: https://redis.io/docs/ (Дата звернення 04.05.2026).
5. Young G. CQRS Documents. – [Електронний ресурс]. – Режим доступу: https://cqrs.files.wordpress.com/2010/11/cqrs_documents.pdf (Дата звернення 04.05.2026).
6. Richardson C. Microservices Patterns. – Manning Publications, 2018. – 520 p (Дата звернення 04.05.2026).

Козюк Юлія Юріївна – студент групи КІТС-25м, факультет менеджменту та інформаційної безпеки, Вінницький національний технічний університет, Вінниця, e-mail: juliakozyk999@gmail.com

Зоря Ірина Сергіївна – ас. каф. Менеджменту та безпеки інформаційних систем, Вінницький національний технічний університет, м. Вінниця, e-mail: ira.zoria@vntu.edu.ua

Koziuk Yuliia Y. – student of the CITS-25m group, Faculty of Management and Information Security, Vinnytsia National Technical University, Vinnytsia, e-mail: juliakozyk999@gmail.com

Zoria Iryna S. – assistant of the Department of Management and Security of Information Systems, Vinnytsia National Technical University, Vinnytsia, e-mail: ira.zoria@vntu.edu.ua