Єфремов Ю. М., к.т.н., доцент
Павлішин М.С., магістрант
Державний університет «Житомирська політехніка»

АНАЛІЗ ПРИЧИН ВИКОРИСТАННЯ МІКРОСЕРВІСНОЇ АРХІТЕКТУРИ В СУЧАСНИХ ПРОГРАМНИХ РІШЕННЯХ

У минулому, на початку 2000-х років, інженери компанії Amazon відзначалися повільним темпом розгортання нового коду, що було далеко від сучасної середньої швидкості розгортання, яка становить приблизно 11,7 секунди. Для виконання цього завдання Amazon використовував спеціальний інженерний підрозділ, який здійснював ручне розміщення нових версій програмного забезпечення в виробничому середовищі компанії.

Що стало причиною таких змін? Amazon перейшов до більш гнучких методів розробки програмного забезпечення і впровадив архітектуру мікросервісів. В наші дні Amazon – це сукупність автономних сервісів, які майже не мають між собою сильних зв'язків або прямих залежностей і керуються командами «двох піц». Кожна з таких команд складається з 6-8 інженерів, які мають повний контроль над одним або двома мікросервісами (окремими функціональними частинами програми). Гнучкі команди відповідають за усі аспекти життєвого циклу продукту, включаючи збір вимог, планування функціональності, проектування, розробку, тестування, розгортання та подальшу підтримку. Саме таким чином Amazon виконує до 50 мільйонів розгортань програмного забезпечення щорічно.

Архітектура мікросервісів, або ж просто мікросервіси, – це спосіб організації розробки програмного забезпечення, при якому програми структуруються як незалежні автономні сервіси. Інакше кажучи, великі та складні продукти розбиваються на невеликі окремі міні-додатки, які відповідають за конкретні бізнес-функції, такі як входи в соціальних мережах або корзина для електронної комерції. Мікросервіси можуть бути слабко зв'язаними і розгортатися незалежно один від одного, що дозволяє змінювати один сервіс, не впливаючи на інші.

Основна ідея архітектури мікросервісів (MSA) полягає у створенні незалежних програмних компонентів, які ефективно вирішують окремі завдання. Кожен мікросервіс має чітко визначену мету, таку як відстеження історії доставки або обробка платежів. При розробці мікросервісів слід керуватися бізнес-процесами, а не горизонтальними шарами, такими як обмін повідомленнями або доступ до даних.

Кожен мікросервіс має власну межу, яка визначається контекстом домену, де він функціонує. Проте визначення відповідних меж для кожного мікросервісу може бути викликом, і тут важливим стає дизайн, орієнтований на домен (DDD). Основною ідеєю DDD є розробка архітектури, яка гармонійно поєднує всі бізнес-процеси та варіанти використання з відповідними артефактами коду. Це дозволяє створювати оптимальну архітектуру програмного забезпечення:

- Аналіз домену;
- Визначення обмеженого контексту;
- Визначення сутностей, агрегатів та сервісів;
- Визначення конкретних мікросервісів.

Використання методології проектування, орієнтованої на домен, має багато переваг. Ваше програмне забезпечення буде оптимально пристосоване до потреб конкретного домену, а не обмежене лише вимогами, такими як дружність інтерфейсу користувача. Ви отримуєте продукт, який враховує потреби цього домену.

Зрозуміло, що архітектура мікросервісів надає великим компаніям гнучкість і можливість використовувати нові технології для задоволення зростаючих потреб клієнтів. Однак для успішного впровадження цієї архітектури потрібно величезне зусилля в рефакторингу та переформатуванні старих монолітних систем зі складними залежностями. Для підтримки нової архітектури також необхідна висококваліфікована команда DevOps, яка забезпечить правильну оркестрацію, автоматизацію процесів та впровадження моделі безперервної доставки.

Список використаних джерел

1. Важливість впровадження мікросервісної архітектури для побудови сучасних легко розширюваних програмних рішень. [Електронний ресурс]. Режим доступу: https://www.infopulse.com/blog/the-importance-of-microservices-architecture-for-modern-applications.
2. Архітектура мікросервісів: Початковий посібник. [Електронний ресурс]. Режим доступу: https://www.infopulse.com/blog/the-importance-of-microservices-architecture-for-modern-applications.