УДК 004.2

М.М. Динако, канд. техн. наук, доц. А.М. Луцків
Тернопільський національний технічний університет імені Івана Пулюя, Україна

ПЕРЕВАГИ ВИКОРИСТАННЯ МІКРОСЕРВІСІВ ПРИ РОЗРОБЦІ СИСТЕМ З ВИСОКИМ ОБЧИСЛЮВАЛЬНИМ НАВАНТАЖЕННЯМ

M.M. Dynako, Ph.D., Assoc. Prof. A.M. Lutskiv

ADVANTAGES OF USING MICROSERVICE IN DEVELOPING LOW COMPUTATIONAL LOAD

Сьогодні процес проектування, розробки та функціонування програмного забезпечення значно еволюціонував в порівнянні із попередніми десятиліттями і характеризується значним ускладненням вимог в зв'язку із зростанням складності бізнес-процесів та зростанням кількості даних в обчислювальних системах. Вимогами до сучасних програмних систем є не тільки вимоги функціональності, але й вимоги до продуктивності, надійності та ефективності рішень. Виходячи з цього, велику роль у задоволенні вимог до програмних систем відіграє вибір архітектури реалізації, оскільки вона значною мірою впливає на алгоритмічні та математичні рішення в процесі розробки. При створенні розподілених обчислювальних систем, зокрема грід та хмарних сервісів для розв'язання ресурсоємних задач щодо створення власних IT-рішень та керування бізнес-процесами, доцільно розробити власну архітектуру програмної системи, або ж обґрунтувати вибір однієї з наявних. Архітектурний стиль мікросервісів - це підхід, при якому програмний додаток будується як набір невеликих сервісів, кожен з яких працює у власному процесі і виконує комунікацію з рештою сервісів використовуючи мережні механізми, як правило HTTP. Ці сервіси побудовані навколо бізнес-потреб і розгортаються незалежно з використанням повністю автоматизованого середовища. Самі по собі ці сервіси можуть бути написані на різних мовах і використовувати різні технології зберігання даних [1].

Табл. 1 Порівняльна характеристика архітектурних підходів

Параметр порівняння	Монолітна	Клієнт-серверна	Багаторівнева	Мікросервісна
Навантаження	• прикладний додаток; • настільний додаток	• мережеві програми; • веб-сайти	• бізнес-системи; • SaaS	• бізнес-системи; • PaaS
Паралелізм	розпаралелення коду	розпаралелення обробки звернень клієнта	• розпаралелення обробки звернень клієнта • масштабування обчислювальних рівнів системи	масштабування бізнес-модулів системи
Чинник порогу ефективно-сті	ресурси обчислювальних машин	• ресурси сервера; • ресурси каналу; • ресурси сховища	• ресурси сервера; • ресурси каналу; • ресурси сховища	ресурси сховища
Домен поширення помилки	вся система	• сеанс клієнта; • вся система	• сеанс клієнта; • сеанс між рівнями	сеанс між модулями

Порівнюючи мікросервісний підхід із іншими існуючими (табл.1), можна виділити ряд визначальних переваг:

- Мікросервіси дозволяють збільшити продуктивність та ефективність систем завдяки зменшенню порогу обмежень відносно паралелізму та розподіленості реалізації окремих сервісів.
- Надійність мікросервісної архітектури та напрацювання на відмову значно вищі, оскільки домен поширення помилок обмежено окремим сервісом.
- Розподіл функціональності та масштабування рівнів виконується у відповідності до потреб бізнесу, виходячи із конкретних вимог до системи.

Мікросервіси описують способи дизайну додатків у вигляді набору незалежно-розгорнутих сервісів, якій характерні організація сервісів навколо бізнес-потреб, автоматичне розгортання, перенесення логіки від шини повідомлень до приймачів та децентралізований контроль над мовами і даними. На додаток до можливості незалежного розгортання і масштабування кожен сервіс також отримує чітку фізичну межу, яка дозволяє різним сервісам бути написаними на різних мовах програмування.

Архітектура мікросервісів використовує бібліотеки, але їх основний спосіб розбиття додатку здійснюється завдяки ділення його на сервіси. У свою чергу сервіси - це компоненти, що виконуються в окремому процесі та взаємодіють між собою через веб-запити або remote procedure call (RPS), а також містить безліч процесів, які завжди розробляються і розгортаються паралельно.

Додатки, побудовані з використанням мікросервісної архітектури, містять власну доменну логіку. Принцип їх дії складається з отримання запиту, застосування логіки і надсилання відповіді. Замість складних протоколів, таких як WS або BPEL, вони використовують прості REST- протоколи.

Виходячи з вищезазначеного, мікросервіси варто розглядати як пріоритетне архітектурне рішення при розробці сучасних обчислювальних систем. Проте існує потреба в розробці алгоритмічного, математичного та програмного забезпечення для спрощення використання мікросервісного підходу в реальних рішеннях.

Авторами здійснюється проектування високопродуктивної обчислювальної системи для задач реалізації розподілених програмних рішень та автоматизації власних бізнес-процесів.

Література

1. Микросервисы (Microservices). [Електронний ресурс]. Режим доступу: URL: http://habrahabr.ru/company/cybersafe/blog/229719/
2. Сравнение службы приложений, облачных служб и виртуальных машин Azure. [Електронний ресурс]. Режим доступу: URL: https://azure.microsoft.com/ru-ru/documentation/articles/choose-web-site-cloud-service-vm/
3. Знакомство с Windows Azure. Для ИТ-специалистов/ Таллоч М.; пер. с нгл. – М.: ЭКОМ Паблишерз, 2014. — 154 с.
4. Подробное описание возможностей разработки с Microsoft Azure Cloud Services. [Електронний ресурс]. Режим доступу: URL: http://habrahabr.ru/company/microsoft/blog/242543/
5. Гибридное облако Microsoft: Руководство по типовым решениям. [Електронний ресурс]. Режим доступу: URL: https://www.microsoftvirtualacademy.com/ru/training-courses/-microsoft--8242?l=frbVWKWCB_8604984382
6. Библиотека технической документации по Azure. [Електронний ресурс]. Режим доступу: URL: https://msdn.microsoft.com/ru-ru/library/dn578280.aspx