

Przedmiot: Zastosowania informatyki

semestr: II

Programowanie w chmurze (Cloud Computing)

Programowanie w chmurze to **tworzenie, wdrażanie i zarządzanie aplikacjami przy wykorzystaniu usług świadczonych przez dostawców chmury obliczeniowej (cloud computing)**.

Zamiast budować własną infrastrukturę serwerową, programiści korzystają z zasobów udostępnianych przez dostawców jak AWS, Google Cloud, Microsoft Azure czy Oracle Cloud.

Kluczowe cechy:

- **Elastyczność skalowania** - automatyczne dopasowanie zasobów do obciążenia
- **Model płatności "pay-as-you-go"** - płacisz tylko za faktycznie użyte zasoby
- **Zarządzana infrastruktura** - dostawca zajmuje się serwerami, siecią, bezpieczeństwem
- **Globalna dostępność** - aplikacje działają w wielu regionach świata

Zastosowania programowania w chmurze

1. Aplikacje webowe i mobilne

2. Przetwarzanie danych Big Data

- Analiza dużych zbiorów danych (Hadoop, Spark w chmurze)
- Machine Learning (Google AI Platform, AWS SageMaker)
- Streamowanie danych w czasie rzeczywistym

3. Internet Rzeczy (IoT)

IoT (Internet Rzeczy, ang. *Internet of Things*) to koncepcja sieci fizycznych obiektów („rzeczy”) wyposażonych w czujniki, oprogramowanie i moduły łączności, które komunikują się ze sobą oraz z użytkownikami przez internet. Umożliwia to automatyzację, zbieranie danych i zdalne sterowanie urządzeniami (smart home, przemysł) przy minimalnej ingerencji człowieka.

Zastosowania programowania w chmurze

4. DevOps i CI/CD

- Automatyczne wdrażanie aplikacji (AWS CodeDeploy, Azure DevOps)
- Konteneryzacja (Docker + Kubernetes w chmurze)
- Monitorowanie i logowanie

5. Aplikacje bezserwerowe (Serverless)

Serverless to model programowania w chmurze, w którym nie zarządzasz serwerami w ogóle. Dostawca chmury (AWS, Azure, Google) automatycznie obsługuje całą infrastrukturę serwerową. Płacisz tylko za czas rzeczywistego wykonania kodu, a nie za działające 24/7 serwery.

Rodzaje (modele) progr. rozproszonego c.d.

Rodzaj:

Klient–serwer (Client–Server) - Klient wysyła żądania, serwer odpowiada.
Przykład: Strona WWW (przeglądarka ↔ serwer)

Wielowarstwowe (n-warstwowe) - System podzielony na warstwy: prezentacji, logiki i danych.

Przykład: Aplikacja webowa: frontend – backend – baza danych

Peer-to-Peer (P2P) - Każdy komputer (węzeł) może być klientem i serwerem jednocześnie.
Przykład: Torrent, komunikatory (np. Skype, BitTorrent)



Rodzaje (modele) programow. rozproszonego

Zdalne wywołania procedur (RPC) - Program na jednym komputerze wywołuje funkcję na innym.

Przykład: Java RMI, gRPC

Systemy oparte o komunikaty (Message-Oriented) - Komputery komunikują się przez kolejki wiadomości.

Przykład: RabbitMQ, Apache Kafka

Mikroserwisowe (Microservices) - System składa się z wielu małych usług, które komunikują się przez sieć.

Przykład: Aplikacje w chmurze (np. Netflix, Amazon)

Grid Computing / Cloud Computing - Wiele serwerów współdzieli zasoby obliczeniowe lub pamięć.

Przykład: AWS, Azure, Google Cloud



Technologie używane w progr. rozproszonym

Obszar:	Przykłady
Backend	Java (Spring Cloud), .NET, <u>Node.js</u>
Komunikacja	REST API, gRPC, WebSocket, message queues (RabbitMQ, Kafka)
Bazy danych	MongoDB, Cassandra, PostgreSQL Cluster
Architektura	mikroserwisy, chmura (AWS, Azure, GCP)

Cechy systemów rozproszonych

Cecha	Znaczenie
Współbieżność	wiele komputerów działa jednocześnie
Komunikacja przez sieć	wymiana danych np. przez HTTP, TCP/IP
Niezależność sprzętowa	różne komputery i systemy operacyjne mogą współpracować
Odporność na awarie	gdy jeden element padnie, reszta może działać dalej
Skalowalność	łatwo dodać więcej maszyn, żeby system działał szybciej

Przykład z życia

Wyobraź sobie aplikację bankową:

- Serwer A obsługuje logowanie,
- Serwer B zapisuje dane klientów w bazie,
- Serwer C przetwarza płatności.

Współpracują razem w sieci — **to właśnie programowanie rozproszone.**