ESPE
UNIVERSIDAD DE LAS FUERZAS ARMADAS
INNOVACIÓN PARA LA EXCELENCIA

MATTER	Advanced Web Programming	**NRC**	8011
RACE	Software Engineering	**Teacher**	Dr. Edison Lascano
THEME	Microservices		
Name	Yeshua Amador Chiliquinga Amaya		

1. Introduction:

Microservices are an architectural approach that structures an application as a set of small, self-contained, independently deployable services. This architecture has revolutionized enterprise software development, allowing for greater scalability and maintainability.

2. Research

Microservices are a software architecture approach where an application is divided into a set of small, independent, and specialized services. Each microservice is autonomous and performs a specific function or task within the overall system. This approach allows each service to be developed, deployed, and scaled independently.

Modern microservices implement specific patterns to solve common distributed architecture challenges. The Circuit Breaker pattern, popularized by Netflix [5], prevents cascading failures by detecting and handling failures in dependent services. The Event Sourcing pattern, together with CQRS (Command Query Responsibility Segregation), allows an immutable record of changes in the state of the application to be maintained, facilitating data consistency and auditing (Richardson, 2022). These patterns are complemented by DevOps practices such as continuous integration and continuous deployment (CI/CD), which automate the development lifecycle and enable frequent and reliable updates

Key Features

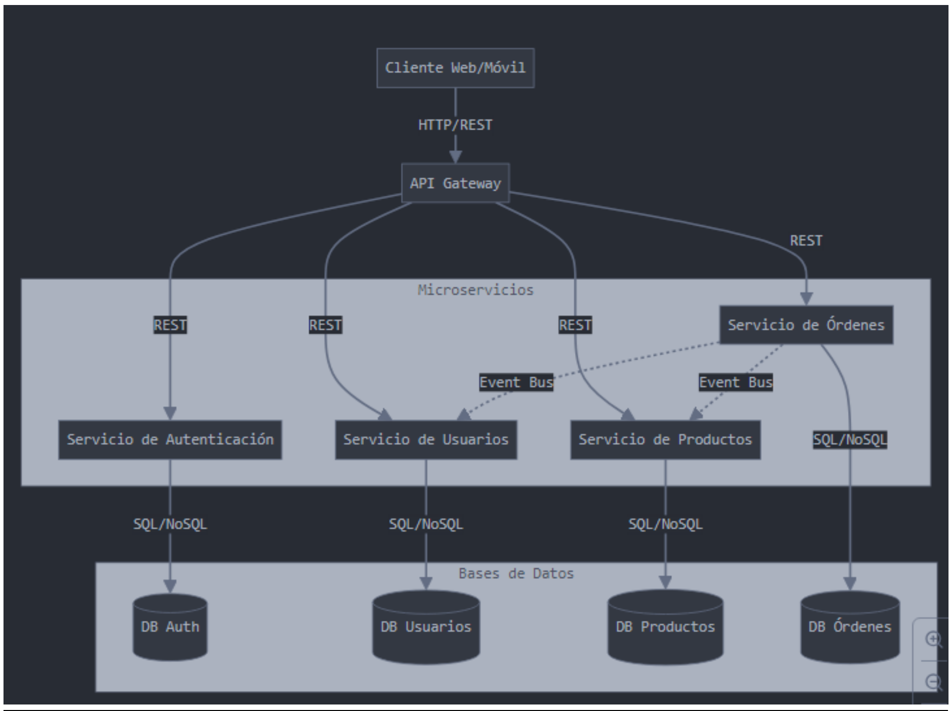
- Independent and autonomous services
- Standalone deployment
- Communication through well-defined APIs
- Decentralized database
- Granular scalability

Core Technologies

- Docker for containerization
- Kubernetes for orchestration
- Spring Boot/Spring Cloud for Java Development
- Node.js with Express for light duty
- API Gateway for Endpoint Management

The diagram represents a typical microservices architecture where:

1. The flow starts with the Client (web/mobile) communicating with the API Gateway
2. The API Gateway acts as a single point of entry and distributes requests to 4 microservices:
    - Authentication
    - Users
    - Products
    - Orders
3. Each microservice has its own independent database, following the principle of "Database per Service"
4. Communication is carried out in two ways:
    - Synchronous: using REST APIs (solid lines)
    - Asynchronous: through an Event Bus (dotted lines) between Orders-Products and Orders-Users



3. Analysis

Deploying microservices with APIs and distributed databases represents a crucial balance between complexity and operational benefits. While separating databases by service increases resilience to failures and allows for better scalability, it also introduces challenges in data consistency and distributed transaction management.

Advantages	Challenges
Increased agility in development	Complexity in distributed management
Selective scalability	Need for robust automation
Improved Failability	Monitoring and traceability
Ease of maintenance	Data consistency across services
Technological freedom by service	Increased operational overhead

4. Conclusion

Microservices represent a significant evolution in software architecture, especially beneficial for complex business applications. Its adoption requires a careful assessment of the needs of the project and the maturity of the development team.

5. References

Newman, Sam. "Building Microservices" (O'Reilly Media)
Evans, Eric. "Domain-Driven Design" (Addison-Wesley)