

**Microservices Architecture**

**Course Number:** MSV-106  
**Duration:** 2 days

**Overview**

This Microservices Architecture training course teaches attendees how to design microservice-based systems for on-prem and cloud deployment. Students learn the top microservices design patterns, how microservices integrate with containerized systems, strategies for integration with existing systems, and more.

**Prerequisites**

All students must know programming fundamentals and software design principles.

**Materials**

All Microservices Architecture training students receive comprehensive courseware.

**Software Needed on Each Student PC**

Attendees will not need to install any software on their computers for this class. The class will be conducted in a remote environment that Accelebrate will provide; students will only need a local computer with a web browser and a stable Internet connection. Any recent version of Microsoft Edge, Mozilla Firefox, or Google Chrome will work well.

**Objectives**

* Understand when to break up/not break up monolithic code when transitioning to microservices
* Explore fundamentals of microservices architecture
* Apply design patterns to ensure the optimal architecture
* Manage APIs
* Integrate microservices with existing systems
* Ensure the stability/robustness of microservices

**Outline**

* Introduction
* Breaking Up Monoliths – Pros and Cons
  + Traditional Monolithic Applications and Their Place
  + Disadvantages of Monoliths
  + Developer's Woes
  + Architecture Modernization
  + Architecture Modernization Challenges
  + Microservices Architecture is Not a Silver Bullet!
  + What May Help?
  + In-Class Discussion
* Microservices
  + What is a "Microservice"?
  + Unix Analogy
  + Principles of Microservices
  + Services within an SOA vs Microservices
  + Properties and Attributes of Microservices
  + Benefits of Using Microservices
  + The Two-Pizza Teams
  + Beware of Microservices Cons
  + Anti-Pattern: Nanoservices
  + The Twelve-Factor App Methodology
  + The Select Factors
  + Serverless Computing
  + Microservices – Operational Aspects
* Microservices Architecture Defined
  + The Microservices Architecture
  + SOA Promises and Expectations
  + Microservices Architecture vs SOA
  + The ESB Connection
  + Microservices Architecture Benefits
  + Microservices Architecture Choices and Attributes
  + Example: On-Line Banking Solution Based on MsA
  + Distributed Computing Challenges
  + Replaceable Component Architecture
  + The Actor Model
  + MapReduce Distributed Computing Framework
  + Hadoop's MapReduce Word Count Job Example
  + What Can Make a Microservices Architecture Brittle?
  + 4+1 Architectural View Model
* Containerization Systems for Microservices
  + Infrastructure as Code
  + Why Not Just Deploy My Code Manually?
  + What is Docker
  + Docker Containers vs Traditional Virtualization
  + Docker is a Platform-as-a-Service
  + Docker Integration
  + Docker Services
  + Docker Application Container Public Repository
  + Container Registries
  + Your Own Docker Image Registry
  + Starting, Inspecting, and Stopping Docker Containers
  + One Process per Container
  + The Dockerfile
  + Kubernetes
  + What is OpenShift
* Commonly Used Patterns
  + Why Use Patterns?
  + Performance-Related Patterns
  + More Performance-Related Patterns
  + Pagination vs. Infinite Scrolling - UX Lazy Loading
  + Integration Patterns
  + More Integration Patterns
  + The Service Mesh Integration Pattern
  + Mesh Pros and Cons
  + Service-to-Service Communication with Mesh
  + Resilience-Related Patterns
  + Summary
* API Management
  + API Management Defined
  + The Traditional Point-to-point Integration Example
  + It Raises Some Questions …
  + The Facade Design Pattern
  + API Management Conceptual Diagram
  + Complimentary Services for Microservices
  + What Else is Needed?
  + The Driving Forces
  + API Management Offerings
  + The Mashery API Management System Overview
  + AWS API Gateway Call Flow
* Designing and Implementing Microservices
  + Two Types of IT Projects
  + What is In Scope for a Robust Microservices Design?
  + Scoping Your Microservice via the Bounded Context
  + Scoping Your Solution's Microservices Architecture
  + External / Shared and Internal Service Models
  + General Architectural and Software Process Organizational Principles
  + Loose Coupling, the OOD Perspective
  + Crossing Process Boundary is Expensive!
  + Cross Cutting Concerns
  + More Cross Cutting Concerns
  + To Centralize or Decentralize Client Access?
  + Decentralized Client Access
  + Centralized Client Access
  + The Facade Pattern
  + The Facade Service Conceptual Diagram
  + The Naked Objects Architectural Pattern
  + When to Use Naked Objects Pattern
  + Dealing with the State
  + How Can I Maintain State?
  + Micro Front-ends (a.k.a. MicroUI)
  + How can MicroUI Help Me?
  + Your Clients Are Diverse
  + The "Rich Client" - "Thin Server" Paradigm
  + The "Rich Client" - "Thin Server" Architecture
  + RIA as a Driving Force to Turn the "Thin Server" into a Set of Microservices
  + Design for Failure
  + Managing Failures Effectively
  + The Immutable Infrastructure Principle
  + Implementing Microservices
  + JAX-RS
  + Microservice-Oriented Application Frameworks and Platforms
  + Embedding Databases
  + Embedded Java Databases
* Microservices Integration
  + One Common Observation
  + The “One Service - One Host” Deployment
  + Things to Consider when Integrating
  + Technology Options
  + The Data Exchange Interoperability Options
  + The Correlation ID
  + Enterprise Integration Patterns
  + Asynchronous Communication
  + Benefits of Message-Oriented Middleware (MOM)
  + Asynchronous Communication Models
  + Message Brokers
  + A Message Broker Diagram
  + Asynchronous Message Consumption Patterns
  + Popular Messaging Systems
  + Challenges of Managing Microservices
  + Options for Managing Microservices
  + In-Class Discussion
* Working with Data in Microservices
  + Monolithic Databases
  + The Traditional Two-phase Commit (2PC) Protocol
  + Table Sharding and Partitioning
  + The CAP Theorem
  + Mechanisms to Guarantee a Single CAP Property
  + The CAP Triangle
  + Eventual Consistency
  + Handling Transactions in Microservices Architecture
  + The Event-Driven Data Sharing Diagram
  + The Saga Pattern
  + The Saga Log and Execution Coordinator
  + The Saga Happy Path
  + A Saga Compensatory Request Example
  + In-Class Discussion
  + The Need for Micro Databases
  + Migrating Data from Existing Databases (Breaking up the Monolith Database)
  + One Data Migration Approach
  + One Data Migration Approach (Cont'd)
  + In-Class Discussion
  + Command Query Responsibility Segregation (CQRS)
  + The CQRS Communications Diagram
  + A Word of Caution
  + The Event Sourcing Pattern
  + Event Sourcing Example
  + Applying Efficiencies to Event Sourcing
* Robust Microservices
  + What Can Make a Microservices Architecture Brittle?
  + Making it Resilient – Mechanisms
  + Techniques and Patterns for Making Your Microservices Robust
  + Fail Fast or Quiesce?
  + Synchronous Communication Timeouts / Retries
  + Asynchronous Communication Timeouts / Retries
  + In-Class Discussion
  + The Circuit Breaker Pattern
  + The Circuit Breaker Pattern Diagram
  + The Bulkhead Pattern
  + Factor IX of the 12 App Methodology
  + Feature Enablement
  + Designing for Test and Failure
  + Making Microservices Testable
  + Test for Failure
  + Continuous Testing and Integration
  + Continuous Release and Deployment
  + SLAs
  + Where and What to Monitor
  + Logging and Monitoring
* Conclusion