

**API Management Fundamentals for Architects Training**

**Course Number:** PROG-106WA  
**Duration:** 2 days

**Overview**

This API Management Fundamentals for Architects training course teaches attendees the basics of API management and new methods of managing APIs, such as those provided by MuleSoft, Apigee, analytics, security, and even microservices. Participants confidently develop their API skills while staying up to date with industry standards.

**Prerequisites**

Participants must understand software design principles and modern choices for application deployment.

**Materials**

All API training students receive comprehensive courseware.

**Software Needed on Each Student PC**

* Google Chrome
* Other modern browsers as desired
* Other free software and lab files that Accelebrate would specify

**Objectives**

* Understand Google's Geocoding API
* Compare API management platforms
* Work with MuleSoft Anypoint Studio
* Understand monolith vs. microservices design
* Understand data availability and consistency
* Work with AWS API Gateway
* Use API management on Azure

**Outline**

* API Management Introduction
  + Data is King
  + Data Exchange Payloads
  + API Management Defined
  + The Driving Forces
  + APIs to Monetize on Your Information Assets and Services
  + The Traditional Point-to-point Integration Example
  + It Raises Some Questions …
  + The Facade Design Pattern
  + API Management Conceptual Architecture
  + Other "Complimentary" Services
  + What Else is Needed?
  + The Digital Transformation Strategies
  + Gartner's Magic Quadrant for Full Life Cycle API Management
  + Apigee
  + MuleSoft
  + Amazon Web Services (AWS)
  + WSO2
  + IBM API Connect
  + KONG
* MuleSoft & WSO2 Overview
  + What is MuleSoft?
  + MuleSoft AnyPoint Platform
  + MuleSoft Cloud Community Manager
  + Anypoint Studio
  + Advanced REST Client (ARC)
  + What is WSO2?
  + User Roles in WSO2 Workflows
  + The Main Client-Facing Web Applications and Capabilities
  + WSO2 Main Features
  + The API Gateway
  + Workflow for Invoking an API
  + Workflow for Creating an API
  + Access Control and Security
  + User Authentication via Access Token
  + Manage and Scale API Traffic
  + Monitor and Monetize
  + WSO2 Cloud
* Apigee Overview
  + What is Apigee?
  + The Big Picture
  + API Consumers
  + Apigee Main Components
  + Apigee Edge API Management Platform Functional Diagram
  + Apigee Capabilities and Actors
  + Apigee Policies for Traffic Management and Data Transformation
  + Apigee Sense
  + Developer Portal
  + Apigee Monetization
  + Core Monetization Components
  + API Runtime
  + API Proxies
  + API Proxy Virtual Hosts
  + Flows
  + OAuth 2.0 Integration
  + API Beauty (an API Runtime's Capability)
  + API Edge Caching
  + Service Composition
  + Apigee Edge Monitoring and Analytics
  + Samples of Monitoring Charts
  + Edge Microgateway
  + Edge Microgateway Integration with Apigee Edge Analytics
  + Examples of Edge Microgateway Deployments (1 of 3)
  + Examples of Edge Microgateway Deployments (2 of 3)
  + Examples of Edge Microgateway Deployments (3 of 3)
  + Pricing: Standard Plan
  + Apigee Hybrid
  + Apigee Hybrid Capabilities
  + Apigee Hybrid Collaboration Diagram
* API Management Analytics
  + API Metrics
  + API Proxy (API Gateway) Performance
  + The Apigee Proxy (API Service Gateway) Performance Chart
  + The Apigee Target Service Performance Chart
  + The Apigee Cache Performance Chart
  + The Apigee Error Code Chart
  + Geolocation Metric
  + The Apigee GeoMap Chart
  + Developer Engagement
  + The Apigee Developer Engagement Chart
  + Reports
* RESTful Services Overview
  + Many Flavors of Services
  + SOAP or REST?
  + Understanding REST
  + Manipulation of Resources through Representations
  + Principles of RESTful Services
  + HTTP Methods
  + HTTP Status Codes
  + Related Standards: MIME
  + Anatomy of a URL
  + GET
  + The GET Method
  + Passing Parameters
  + HTTP Methods That Modify/Create Resources
  + POST
  + POST Request Example
  + POST Response Example
  + PUT
  + PUT Example - Update
  + PUT Example - Create
  + PATCH
  + Patch Example - Update
  + To PUT or POST?
  + SOAP Examples
  + A RESTful API for Tracking Video Games
  + What is gRPC?
  + How It Works
  + gRPC Request - Response Diagram
  + REST vs. gRPC
  + So, REST or gRPC?
* Describing RESTful APIs
  + Describing RESTful APIs
  + History
  + Ad-hoc Interface definitions
  + Modern Choices
  + Stacking Up Open API vs RAML
  + Open API vs RAML: Advantages and Disadvantages
  + RAML
  + Build A RAML API Description
  + ROOT
  + List Resources
  + Define Methods
  + Define Query Parameters
  + Enter Responses
  + Open API (Swagger)
  + Swagger Versions
  + Swagger Editor
  + Swagger Codegen
  + Swagger UI
  + OpenAPI Service Description
  + Pretty Printed OpenAPI Service Definition
* Securing APIs
  + What are the Security Domains?
  + How to Secure an API?
  + DDoS Protection
  + Authentication
  + OAuth2 Overview
  + OAuth Components
  + OAuth Authentication and Authorization Overview
  + OAuth Collaboration Diagram
  + Example: OAuth Client Credentials Grant with Apigee Edge
  + Getting Authorization Code Through a Federated External Directory (Facebook or Google)
  + What is Security Assertion Markup Language (SAML)?
  + SAML 2.0 Web SSO Authentication
  + SAML vs. OAuth2
  + Identity mediation
  + Encryption of Data in Transit with TLS
  + Key and Certificate Management
  + Threat Detection
  + Apigee Content-based Security
  + JSON Web Tokens (JWT)
  + How JWT Works
  + OAuth vs. JWT
  + Data-masking
  + Last-mile security
* Microservices Architecture Overview
  + What is a "Microservice"?
  + Principles of Microservices
  + Properties and Attributes of Microservices
  + Benefits of Using Microservices
  + The Microservices Architecture
  + 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
  + What Can Make a Microservices Architecture Brittle?
* Designing and Implementing Microservices
  + Two Types of IT Projects
  + What is In Scope of 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 Service Mesh Integration Pattern
  + Istio
  + Mesh Pros and Cons
  + Service-to-Service Communication with Mesh
  + 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
  + Resilience-Related Design Patterns
  + The Immutable Infrastructure Principle
  + Implementing Microservices
  + Microservice-Oriented Application Frameworks and Platforms
  + Embedding Databases
  + Embedded Java Databases
* Conclusion