

# Title Page

**Enterprise Application Architecture with .NET Core**

An architectural journey into Microsoft .NET open source platform

Ganesan Senthilvel  
Ovais Mehboob Ahmed Khan  
Habib Ahmed Qureshi



**BIRMINGHAM - MUMBAI**

# Table of Contents

## Preface

- What this book covers
- What you need for this book
- Who this book is for
- Conventions
- Reader feedback
- Customer support
  - Downloading the example code
  - Downloading the color images of this book
- Errata
- Piracy
- Questions

## 1. Enterprise Architecture Concepts

- Why do we need Enterprise Architecture?
  - Definition of Enterprise Architecture
  - Stakeholders of Enterprise Architecture
  - Business benefits
- Knowing the role of an architect
  - Role comparison between EA and SA
  - Degree of Comparisons
- Commonly known EA Frameworks
  - General Purpose Frameworks
  - Domain Specific Frameworks
- Architecture segregation
  - Business Architecture
  - Data Architecture
  - Application Architecture
  - Technology Architecture
- Introduction to TOGAF
  - Evolution of TOGAF 9.1
  - Core components
  - Industry usage
- Introduction to Zachman
  - Evolution
  - Core components
- Summary

## 2. Principles and Patterns

- Getting started with principles and patterns
  - Why follow design principles?
  - What are design patterns?
  - Why use design patterns?
- SOLID design principles

SRP - Single Responsibility Principle  
SRP example - The decorator pattern

OCP - Open Closed Principle

LSP - Liskov Substitution Principle

ISP - Interface Segregation Principle

DIP - Dependency Inversion Principle

Dependency injection

Introducing dependency injection

Knowing about the Service Locator pattern

Dependency injection support with .NET Core

GoF design patterns

What are software design patterns?

Creational patterns

The singleton pattern

Variations on the singleton pattern

The factory method pattern

Abstract factory pattern

Builder pattern

A commentary on creational patterns

Structural patterns

Adapter pattern

Bridge pattern

Flyweight pattern

A commentary on structural patterns

Behavioral patterns

The template method pattern

The observer pattern

The chain of responsibility pattern

The visitor pattern

The strategy pattern

The state pattern

A commentary on behavioral patterns

Summary

### 3. Distributed Computing

Understanding Distributed applications

Definition

Comparison

Multiprogramming

Thread synchronization

Storage

Process

Concurrency

Parallelism

Multithreading exercise

ThreadStart

ThreadPool

Task Parallel Library (TPL)

Design challenges

Transparency

Reliability

Fault tolerance

Performance

Decompose

Caching

Scalability

Scale up

Scale out

Comparing scale up with scale out

Connecting the dots

Security

Goals

Attack

Threats

Summary

#### 4. Software Development Life Cycle

What is SDLC?

Need for a process

Insight of core phases

SDLC models

The Waterfall model

Core phases

Business requirement

System analysis

System Design

Coding

Testing

Maintenance

Understanding the Spiral model

Core phases

Comparing the Waterfall model with the Spiral model

Benefits

Challenges

Usage recommendation

Agile model

Top five reasons to adopt Agile

Ambiguous requirements

Requirement changes

Big planning is not practical

Software review is better than document

Iterative incremental delivery is preferred

Industry evidence

Scaled Agile Framework (SAFe)

History

Success Factors

Microsoft open source strategy to life cycle

Traditional Microsoft model and its origin from MS-DOS

Driving factors of the open source model

Twin tracks of .NET Framework and .NET Core

Comparing .NET with .NET Core

Current stack of open source tools and techniques

Summary

## 5. Enterprise Practices in Software Development

What is ALM?

Core aspects

ALM vs SDLC

Source Code Control System

Git

TFS

Git vs TFS

Visual Studio Integration

Team Foundation Version Control (TFVC)

Git

Developing .NET Core project templates for enterprise applications

Creating a custom .NET Core project template using .NET command-line interface tools

Performance measuring for .NET applications

CPU utilization

Using the Sampling method in Visual Studio to collect performance statistics

Measuring UI responsiveness

Analysing memory leaks

Identifying memory leaks

Summary

## 6. Layered Approach to Solution Architecture

Layers in layered architecture

Presentation layer

Service layer

Business layer

Transaction Script pattern

Table Module pattern

Active Record pattern

Domain Driven Design (DDD) pattern

Data access layer

Objectives of layered architecture

Practical implementation of layered architecture in .NET Core

Scope

Logical architecture

Presentation layer

Service layer

- Business layer
- Data access layer
- Common layer
- Setting up the environment
- Creating the solution
  - Creating the common layer
    - Entities mapped to database tables
    - Business objects
    - Logging events
    - Logging helper
  - Data access layer
    - Creating Data Context
    - Creating DbFactory
    - Repository pattern
    - Unit of Work pattern
    - Running migration
  - Business layer
    - Develop core classes
    - Developing business managers
    - Logging in .NET Core
  - Creating the service layer
    - Creating base controller
    - Adding Custom Action Filters
    - Add controllers
  - Creating the presentation layer
    - Single Page Applications
    - Benefits of a SPA
    - Developing the presentation layer using ASP.NET Core and Angular
    - Setting up frontend packages
    - Configuring the ASP.NET Core pipeline
    - Adding the Angular components
    - Creating MVC Controllers and Views

Summary

## 7. SOA Implementation with .NET Core

SOA definition

What is SOA?

SOA modeling

SOA Reference Model

Reference model and reference architecture relationship

SOA Reference Architecture

Common reference information architecture

Common reference Infrastructure architecture

SOA features and components

Service Component Architecture

Service types

- Service composition
  - Service orchestration
  - Service choreography
- Common technology standards
- Service discovery
- Message broker
- Enterprise Service Bus (ESB)
  - ESB Segments
  - ESB features
- Data
  - Master Data Management (MDM)
  - Common data model
  - Live business metrics
- Services gateway
- SOA services library
- Tracking, logging, and error handling in SOA
- Notes
- Sample SOA implementation
  - Introduction
  - Sample enterprise
    - Departments of a sample enterprise
    - Sample data models for departments
    - Sample business processes for departments
    - Sample database models for departments
  - Bounded contexts
  - Services implementation
    - Solution structure
    - Sample database
    - Sample development and system services
    - Sample information service
      - Employee information SOA service
      - Employee Information business logic layer
      - Repositories in the data access layer
      - Employee information core data access layer
      - Entity in an employee information model
    - Sample adapter service
    - Sample background service
    - Sample interaction (notification) service
    - Sample mediation service
    - Sample scenario of a service choreography
- Summary
- 8. Cloud-Based Architecture and Integration with .NET Core
  - Cloud Computing Models
    - Infrastructure as a Service (IaaS)
    - Platform as a Service (PaaS)

- Software as a Service (SaaS)
- Azure compute
  - Virtual machines
  - Cloud services
    - Worker Role
    - Web Role
  - App Services
  - Azure Service Fabric
  - Features comparison between virtual machines, cloud services, Azure App Services, and Service Fabric
- Rapid application development using Azure App Services
  - Web Apps
    - Hosting an ASP.NET Core application on Azure
    - Deployment slots
  - API Apps
    - Configuring Swagger in ASP.NET Core Web API and deploying on Azure
    - Creating proxy classes using AutoRest in .NET Core
    - Enable CORS
  - Mobile Apps
    - Offline sync
    - Push notifications
  - Logic Apps
    - Connectors
    - Trigger
    - Actions
    - Creating Logic App in Azure
  - Scaling Azure App Services
- Background services and event handling in cloud
  - WebJobs
    - Developing WebJob using .NET Core
    - Developing WebJobs using WebJobs SDK and .NET Framework 4.5
    - Azure WebJobs versus Azure WorkerRoles
    - Using WebHooks for event-based scenarios
      - Using WebHook of WebJob from VSTS
  - Azure Functions
    - Creating a basic Azure Function to listen for Queue events
- Scalability and performance options in Azure App Services
  - Increasing storage performance
    - Command-Query Responsibility Segregation (CQRS) pattern
    - Denormalization
      - Azure Table storage
      - MongoDB
  - Caching
    - Local cache
    - Shared cache
    - Using Redis Cache in Azure



Creating the Redis Cache

Configuring the .NET Core app to use Redis Cache on Azure

Queuing

Logging and monitoring in Azure

Logging

ASP.NET Core logging in Azure

Web server diagnostics

Application diagnostics

Accessing logs

Accessing logs via FTP

Accessing Logs via Azure PowerShell

Monitoring

SCOM (System Center Operations Manager)

Application Insights

Application hosted on Azure

Application hosted on-premise

Use Application Insights API

Setting up Application Insights in ASP.NET Core Application

Summary

## 9. Microservices Architecture

Microservices architecture definition

What is microservices architecture?

Microservices and SOA

Microservices and monolithic applications

Web API and web services

Characteristics of a microservices architecture

Best for microservices architecture

Documentation

Business capabilities

Business processes

Microservice interfaces

Microservice code

Microservice data store

Logging and monitoring

Immutable Infrastructure

Containerization

Stateless

Architectural elements

Bounded Context in Domain Driven Design

DDD (Domain Driven Design)

Guiding principles

Foundational concepts

Bounded context

Microservices come in systems

Service discovery

Client-side service discovery

- Server-side service discovery
  - Service registry
  - API gateway
- Architectural motivations
  - Agile Manifesto
  - Reactive Manifesto
    - Reactive systems
  - Reactive microservices architecture
    - Key aspects of Reactive Microservices
  - Serverless architecture
    - Backend as a Service (BaaS)
    - Function as a Service (FaaS)
    - Key aspects of serverless architecture
      - Type of code
      - Stateless
      - Short-lived
      - Almost zero administration
      - Automatic scaling
      - Event-driven
  - Let's wrap it up
- Azure for microservices
  - Azure Functions
  - Azure Service Fabric
  - Azure Container Service
  - Bringing it together
- Implementation samples
  - Microservices architecture for our sample enterprise
    - Problem domains
      - Publishing team
      - Marketing team
      - Sales team
      - Platform administration team
      - Other teams
    - Contexts for the respective teams
      - Customer Relationship Management system
      - Document Management System
      - Understanding the Microservices Bounded Team Contexts
      - General service information flow
      - Sales Team Context
      - Marketing Team Context
      - Publishing Team Context
      - Platform Administration Team Context
  - Enterprise portal mockup
  - Overall microservices architecture
    - Common communication mechanisms in microservices

- Serverless architecture for a sample application
  - Our sample application - Home automation
  - High-level application design
  - Serverless architecture in Azure
  - Let's wrap it up

- Summary

## 10. Security Practices with .NET Core

- Authentication and authorization modes

- Securing applications with ASP.NET Core Identity

- Security architecture in ASP.NET Core

- Getting to know the core APIs of the Identity system

- HttpContext and AuthenticationManager

- Understanding the authentication and authorization process

- Authentication

- Implementing authentication using ASP.NET Core Identity and customizing the Identity data store

- Configuring authentication using Identity in an empty web application project

- Configuring Entity Framework Core

- Defining data context and user classes

- Configuring database connection and application configuration settings

- Configuring Entity Framework and Identity services

- Enabling authentication using Identity

- Creating an identity data store in SQL server

- Customizing existing Identity data store and adding new entities

- Creating and Signing-in/Signing-out users

- Adding claims in ASP.NET Identity membership

- How authorization works

- Using cookie middleware without ASP.NET Core Identity

- Claims transformation

- Cookie middleware events

- Implementing external authentication in ASP.NET Core applications

- Configuring external authentication in ASP.NET Core

- Creating a web application project

- Configuring apps on Facebook

- Enabling Facebook middleware

- Two-factor authentication

- Setting up an SMS account

- Enabling two-factor authentication

- Security in an enterprise

- Getting started with IdentityServer4

- Understanding OAuth

- Actors in OAuth

- Flows of OAuth 2.0

- Client credentials flow

- Implicit flow

- Authorization code flow

- Resource owner password credentials flow
- Understanding OpenID Connect
  - OpenID Connect flows
    - Authorization code flow
    - Implicit flow
    - Hybrid flow
    - Claims and scopes
    - Endpoints
      - Discovery endpoint
      - Authorize endpoint
      - Token endpoint
      - UserInfo endpoint
- Developing a Centralized Authorization System using IdentityServer4
  - Creating a Centralized Authentication Service/Authorization Server
    - Setting up IdentityServer4
    - Defining scopes, clients and users
    - Adding UI to enable authentication using OpenID Connect
- Creating an MVC web application project
  - Adding OIDC and cookie middleware in HTTP pipeline
  - Enabling MVC and controller
  - Adding a Web API
- Authorization
  - Declarative authorization techniques
  - Basic authorization
  - Authorization filters
    - Filtering based on authentication schemes
    - Filtering based on authorization
    - Filtering based on policy
  - Custom policies
  - Imperative authorization techniques
- Safe storage
  - Storing and retrieving safe storage values
- Summary

## 11. Modern AI Offerings by Microsoft

- Virtual machines and containerization
  - Virtual machine
    - Simulation
    - Emulation
    - Virtual machine implementation base
  - Containerization
    - Evolution of containerization concepts
      - Chroot
      - FreeBSD Jails
      - Solaris Zones
      - OpenVZ
      - Cgroups

- LXC
- Lmctfy
- Docker
  - Modern container fundamentals
  - Docker components
    - Docker Engine
    - Docker Compose
    - Docker Machine
    - Docker registry
    - Docker Kitematic
    - Docker Swarm
      - Swarm mode
    - Docker Cloud
  - Docker containerization in Windows
    - Docker for Windows
    - Windows Containers
- Modern development
  - Development editors
  - Development environment setup
    - Vagrant
  - Cloud development and test environment
- DevOps
  - The Culture
  - Key motivational aspects
    - Sharing
    - Automation
    - Measurement
  - Software development and delivery process
  - Continuous Integration
    - Best practices
    - Benefits of CI
      - Improvement in Developer productivity
      - Quick identification and addressing of bugs
      - Faster Updates Delivery
  - Continuous Delivery
    - Continuous Delivery Pipeline
  - DevOps toolchain
- A sample microservices application based on Docker containers
  - The sample application
    - Problem statement
    - Application architecture
    - Technical architecture
      - Setup in Azure Container Service
      - Architecture diagram
    - Network architecture
      - What is visible in this diagram?

- Hands-on prerequisites
- Why Azure Container Service?
  - Azure App Service (on) Linux
  - Creating VM directly on Azure
  - Azure Service Fabric (ASF)
  - Azure Container Service (ACS)
- Implementing the Math app
  - Implementation approach
  - Implementation Steps
    - Installing the Hypervisor
    - CentOS virtual machine
    - CentOS configuration
    - Container installation and execution
    - Uploading container images to container registry
    - Creating Azure Container Service
    - Container installation and execution on ACS

## Big Data and Microsoft

- Definition of Schema
- Schema free - NoSQL
- Fixed vs no schema
- NoSQL types
- Architectural best practices
- Microsoft HDInsight
  - HDInsight ecosystem

## Introduction to Business Intelligence (BI)

- Current trend
- Road map
- Power BI architecture
- Power BI layers

## Artificial intelligence (AI)

- Core components
- Machine learning (ML)
- Data mining
- Interconnectivity
- AI at Microsoft
- Industry Bots
- Microsoft open source strategy
- Cognitive Services
- Microsoft Bot

## Summary