



Web Servislerine Giriş

Zafer CÖMERT



**Bölüm
9**

Yazılım Mimarisi

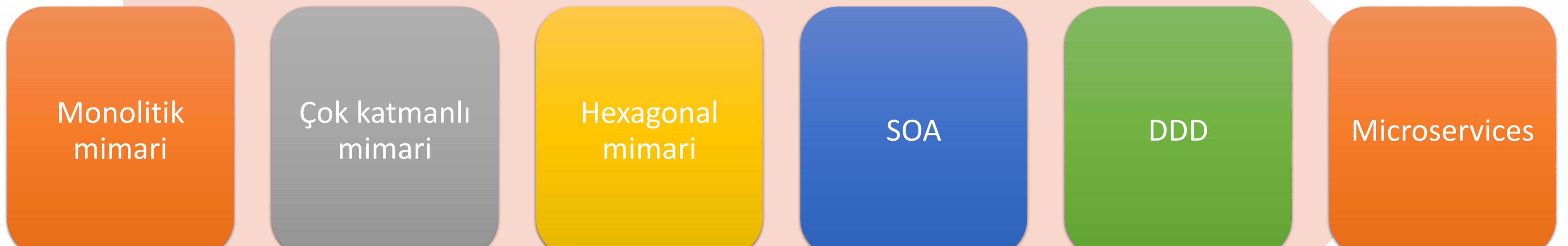
Web Servislerine Giriş



İçerik

- Monolitik Mimari
- Çok Katmanlı (n-Tier) Mimari
- Altıgen (Hexagonal) Mimari
- Servis Odaklı Mimari (SOA)
- Olay Tabanlı Mimari (DDD)
- Mikroservis Mimarisi

Mimariler



Advances in networking
(More distributed computing)

Monolitik Mimari

Monolitik mimari, bir yazılım uygulamasının tüm bileşenlerinin tek bir yapıda bulunduğu bir yazılım tasarım yaklaşımıdır.

Monolitik Mimari

Tek bir
codebase

Tek bir
repository

Tek bir
process

Tek bir
host

Tek bir dil

Yeni projeler için uygundur

Kod yeniden kullanımı (ortak işlevler)

Yerel olarak çalıştırması daha kolay

Build edilecek tek şey

Deploy edilecek tek şey

Üçtan uca test edilecek tek şey

Ölçeklenecek tek şey

Monolitik Mimari

Kolayca
anlaşılamayacak
kadar karmaşık
hale gelir

Kod birleştirmek
zor olabilir

IDE'leri yavaşlatır

Build süreleri uzar

Yavaş ve seyrek
dağıtımlara neden
olur

Uzun test ve
stabilizasyon
sureleri

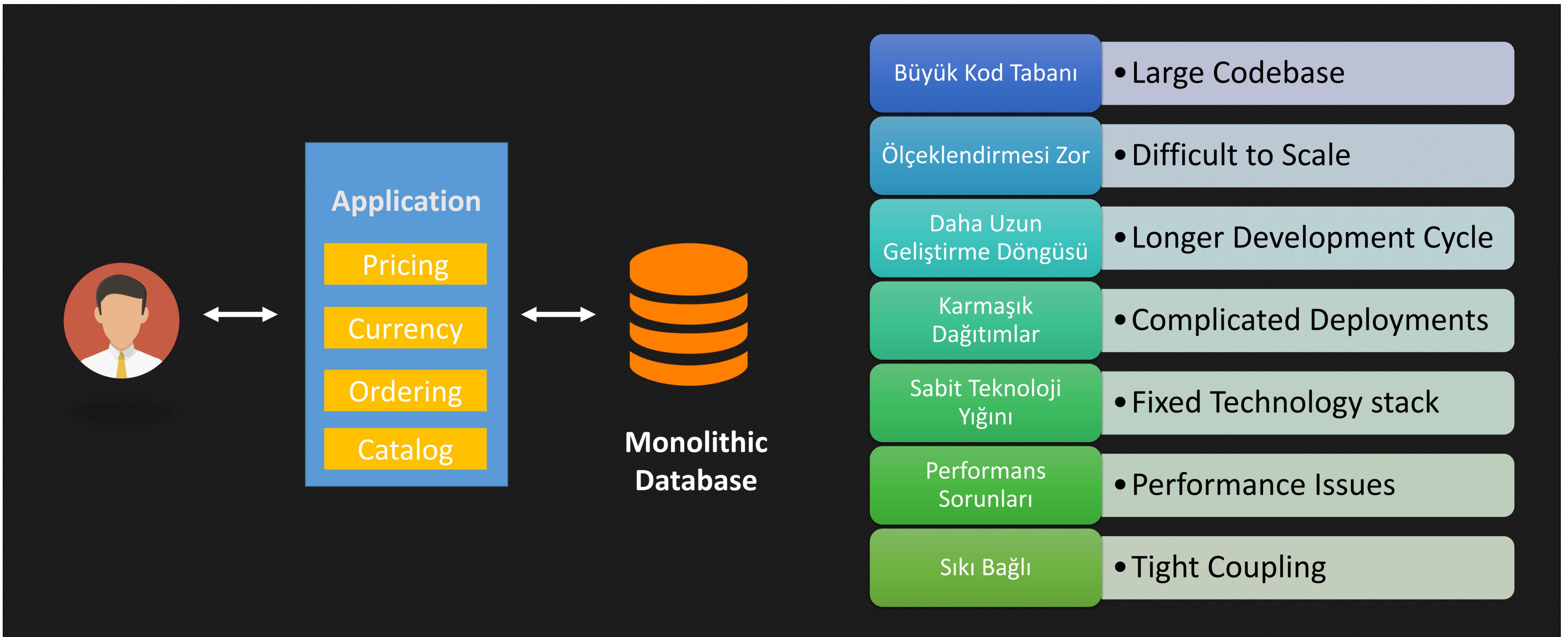
Geri alma ya hep ya hiçtir

Modüller arasında izolasyon yok

Ölçeklendirmek zor olabilir

Yeni teknolojilerin benimsenmesini
zorlaştırır

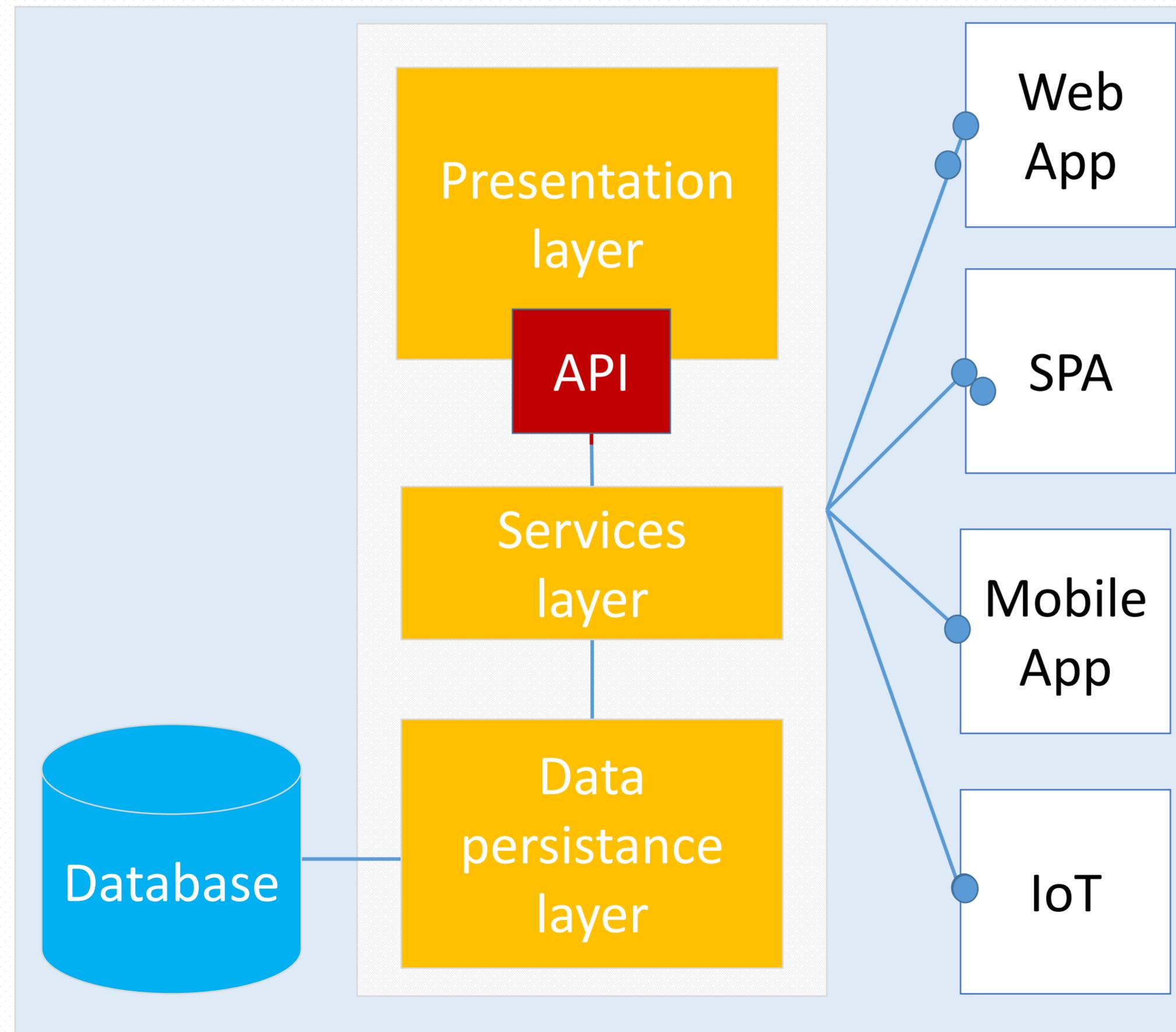
Monolitik Mimari



Katmanlı Mimari

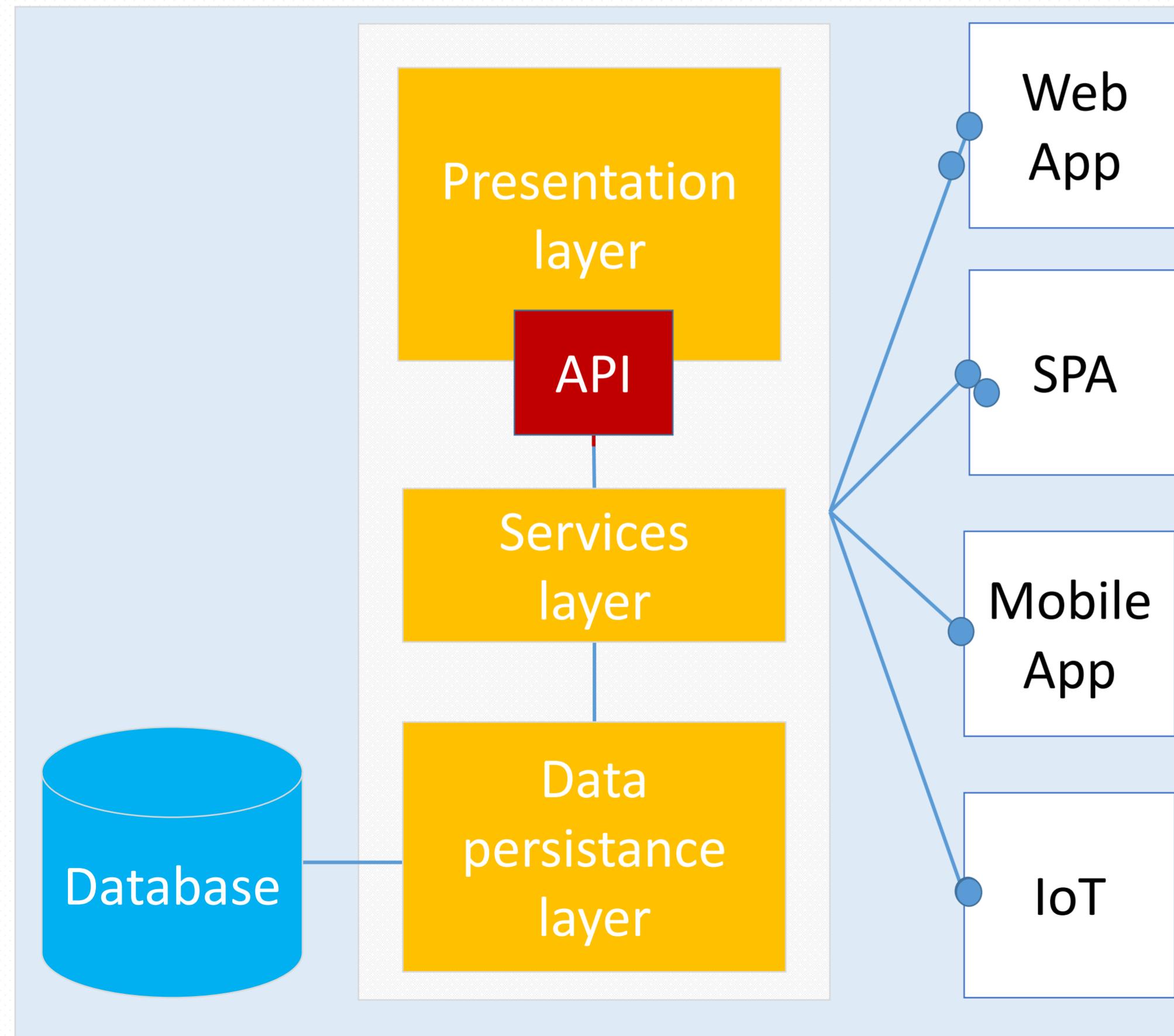
n-Tier Architecture

Çok Katmanlı Mimari



- Katmanlama
- Modülerlik
- Esneklik
- Yeniden Kullanılabilirlik
- Bakım Kolaylığı
- Test edilebilirlik
- Ölçeklenebilirlik

Çok Katmanlı Mimari



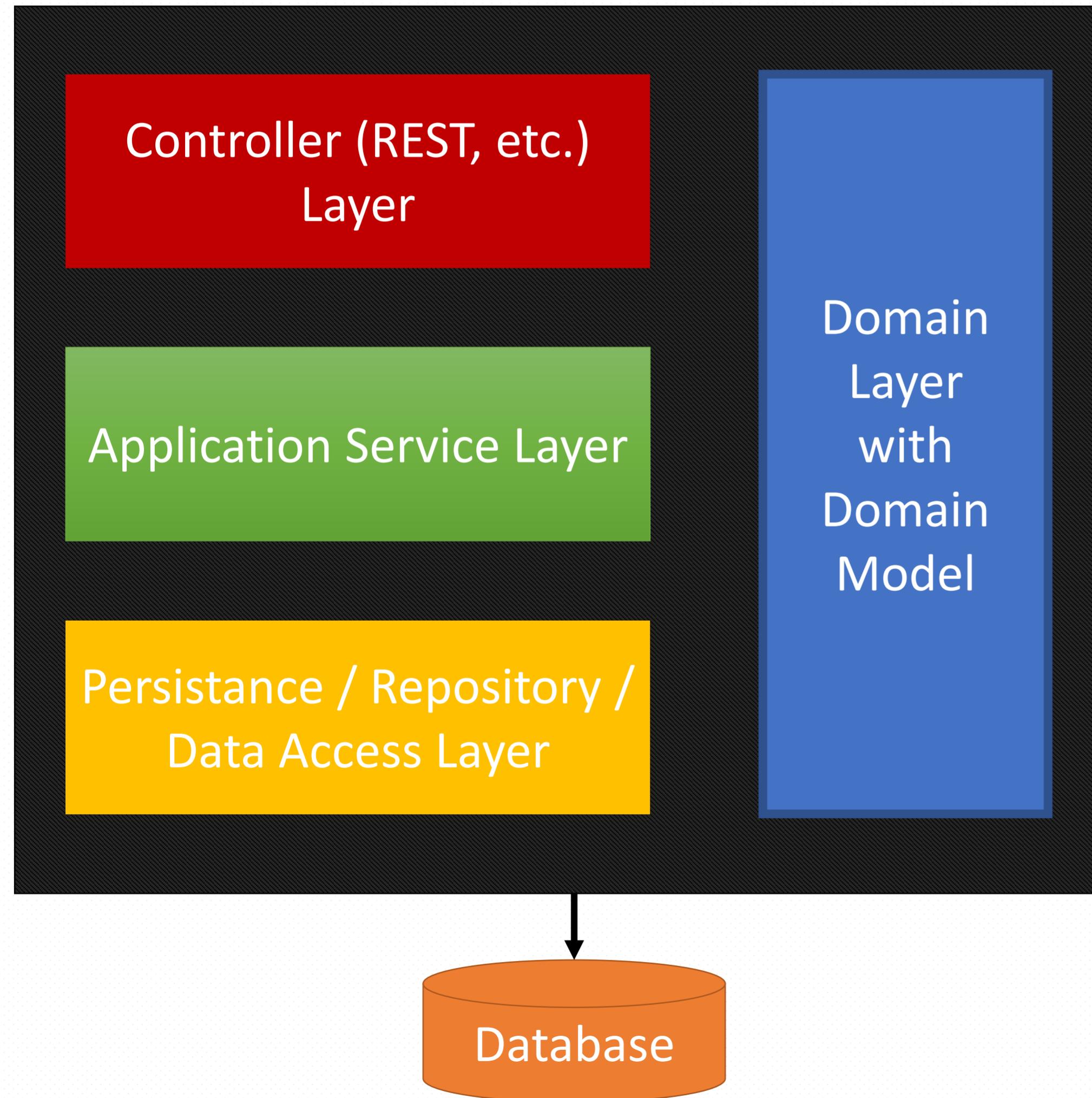
Karmaşıklık

Performans

Geliştirme Zamanı

Tekrarlanan Kod

n-Katmanlı Mimari



- Tipik tek bir mikroservis bu şekilde görülebilir.
- Uygulamanın temel düşüncesi domain model ile birlikte domain katmanında yer alır.

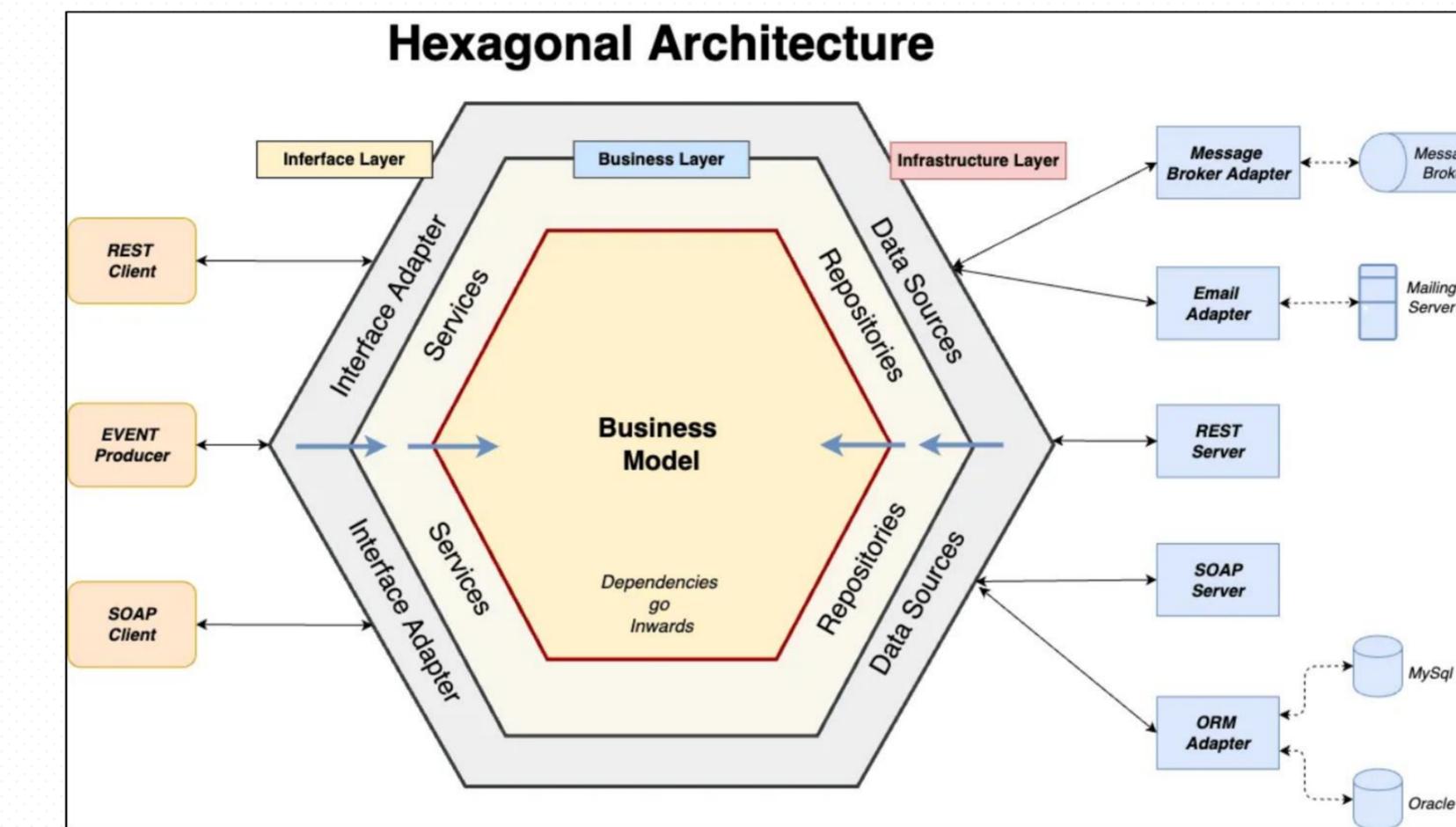
Altıgen Mimari

Hexagonal Architecture

Ports and Adapters

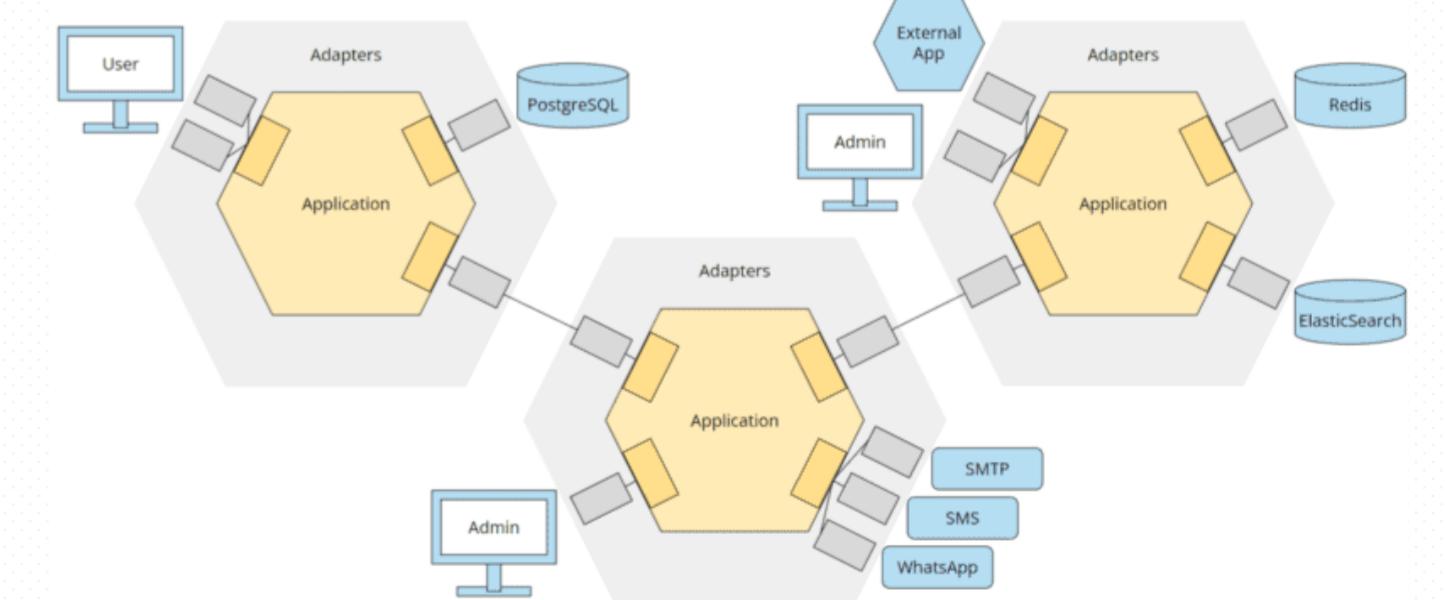
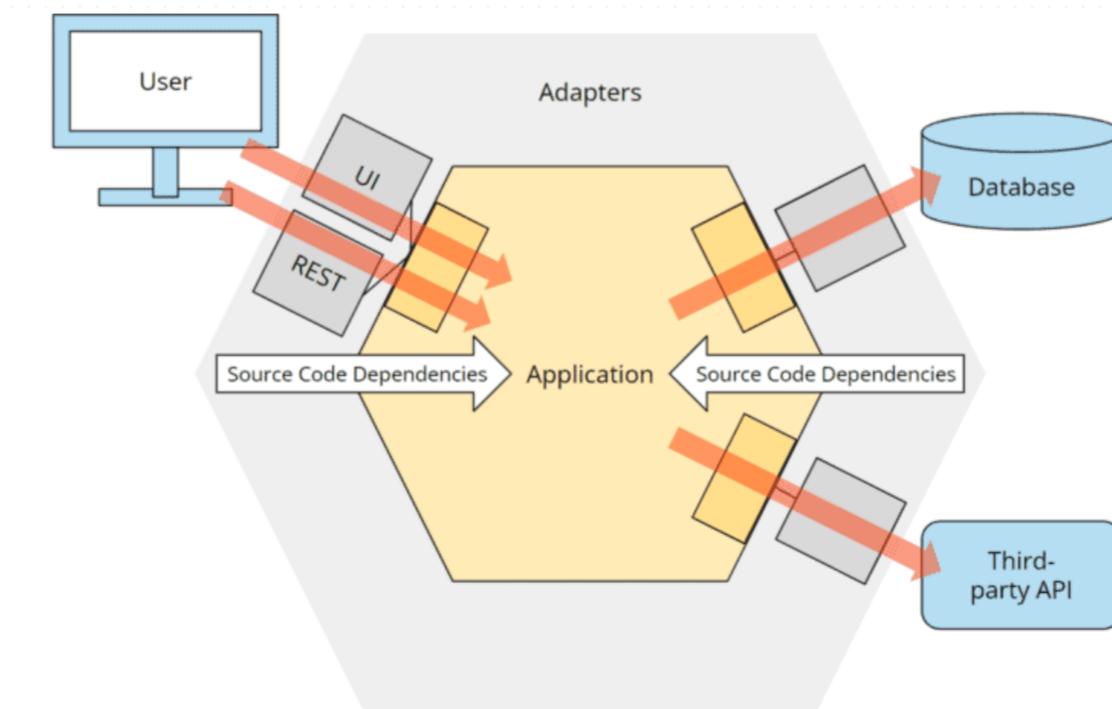
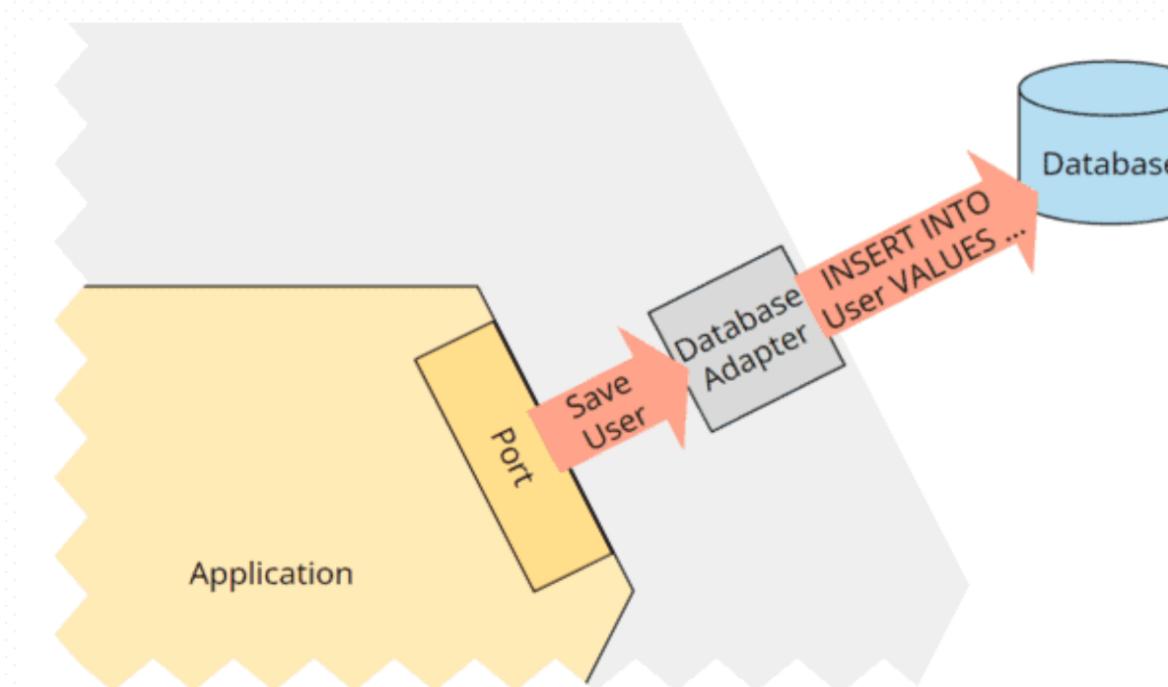
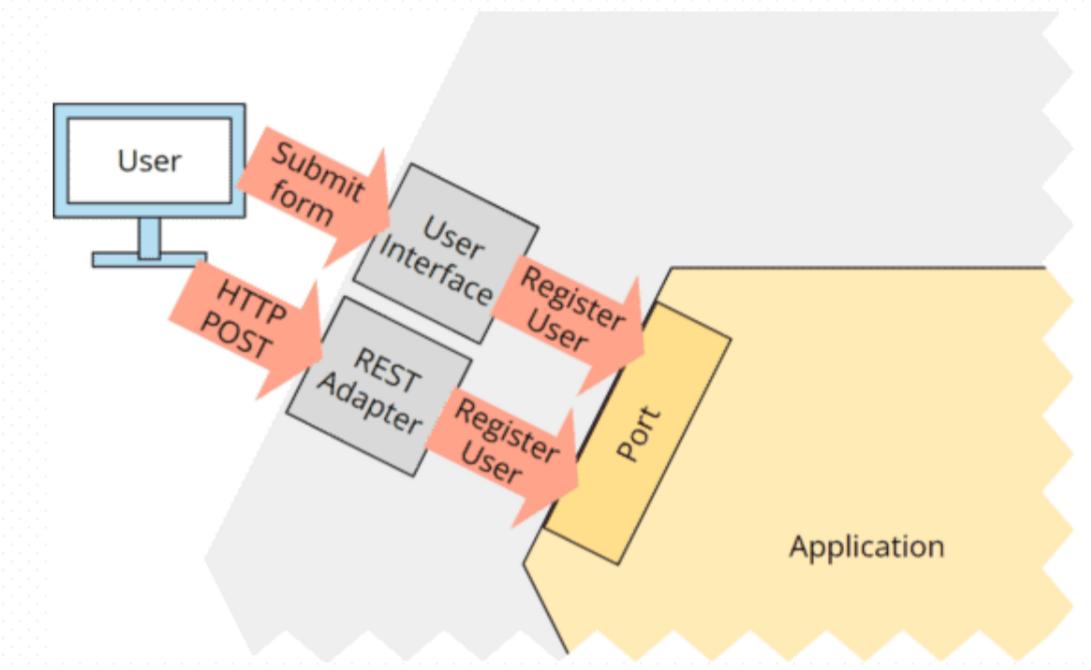
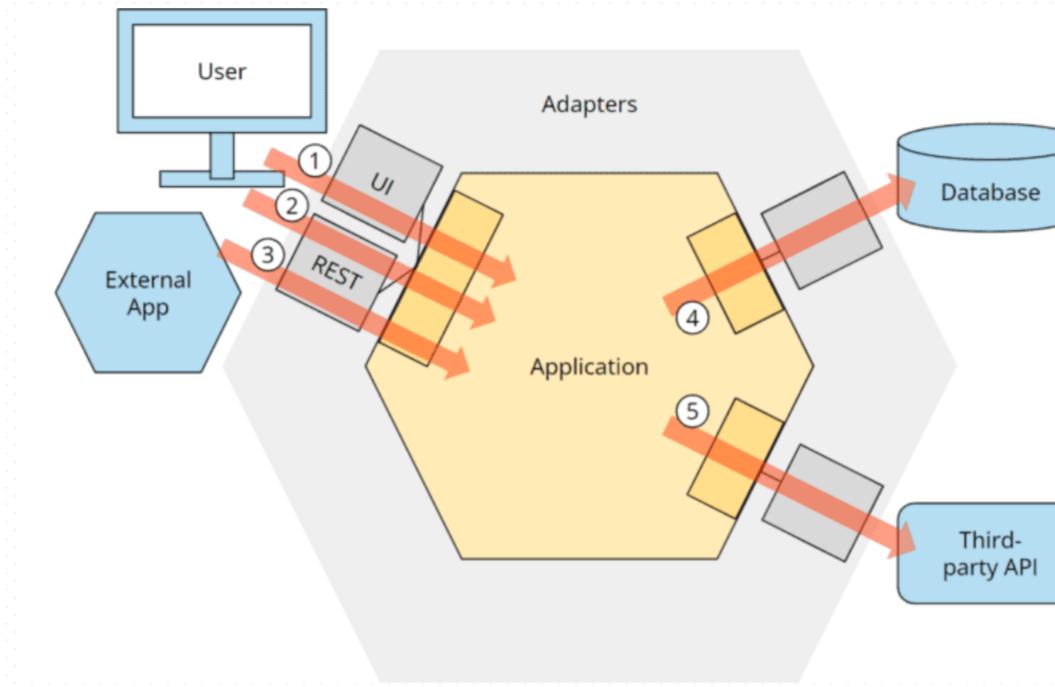
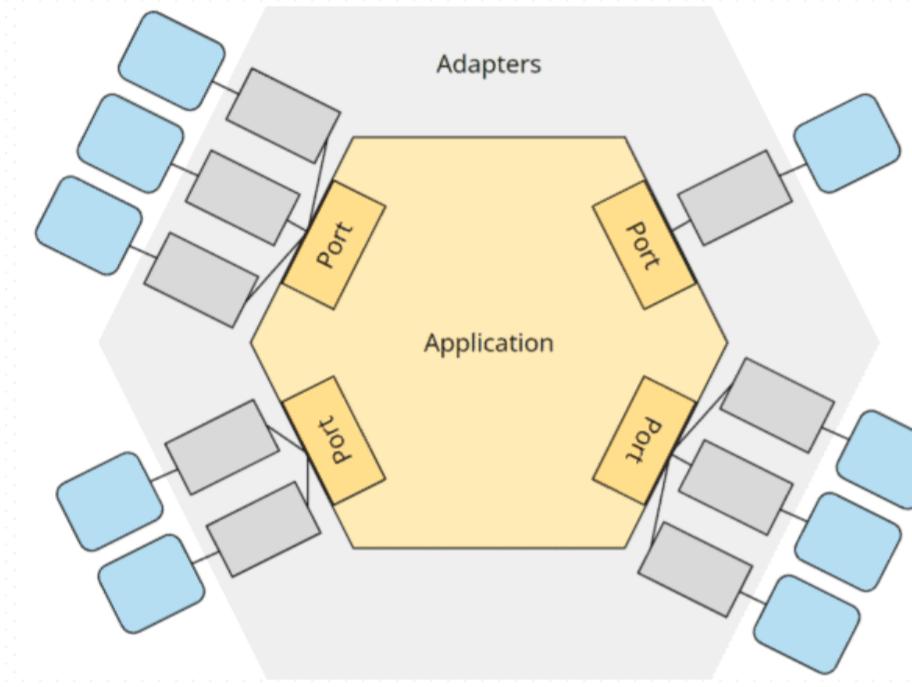
Onion Architecture

Altıgen Mimari

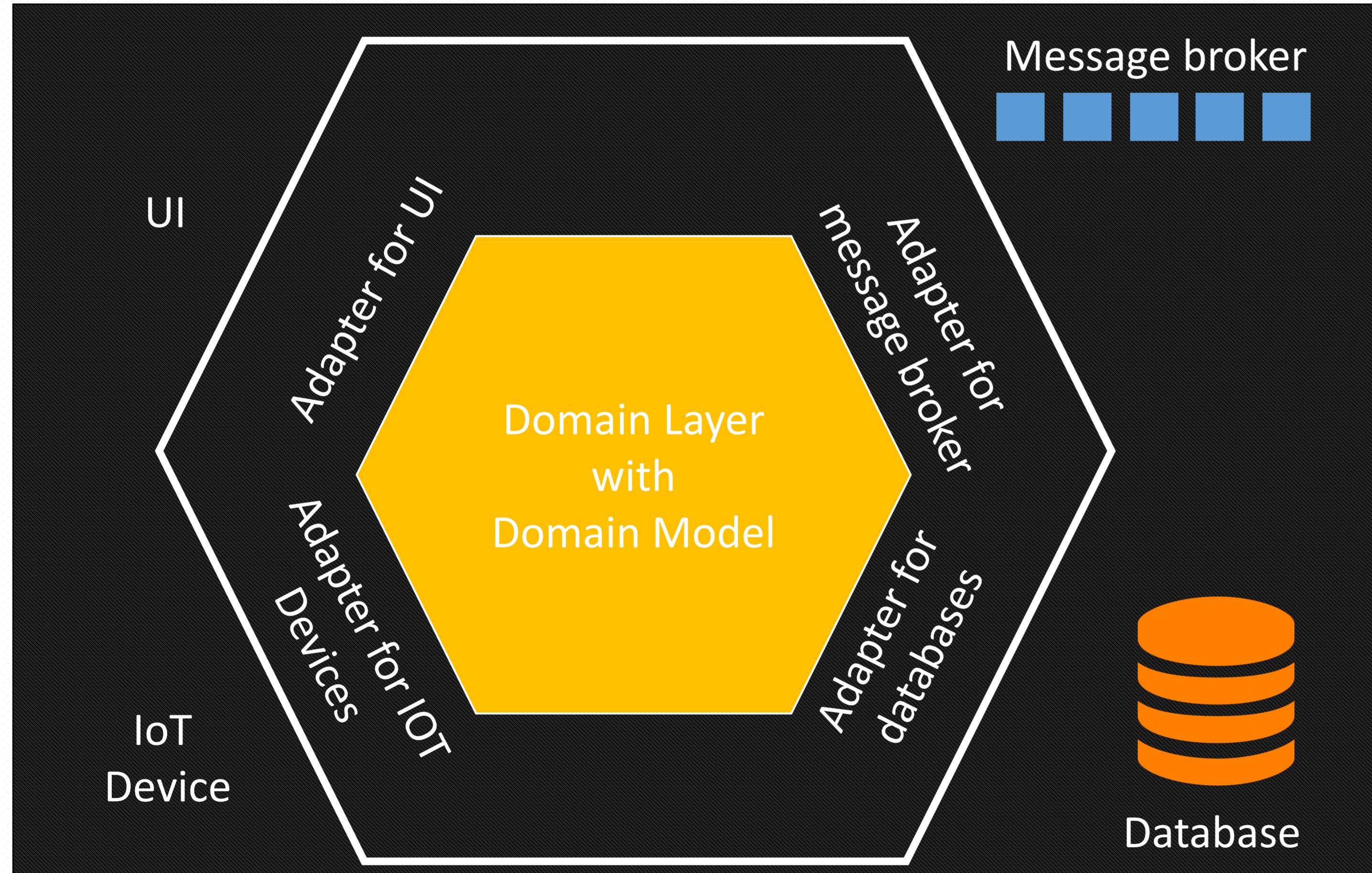


- Altıgen Mimarisi, Ports ve Adapters veya Soğan Mimarisi olarak da bilinen, Alistair Cockburn tarafından 2000'lerin başlarında, sürdürülebilir, adapte edilebilir ve kolayca test edilebilir uygulamalar oluşturmak için bir mimari desen olarak tanıtıldı.

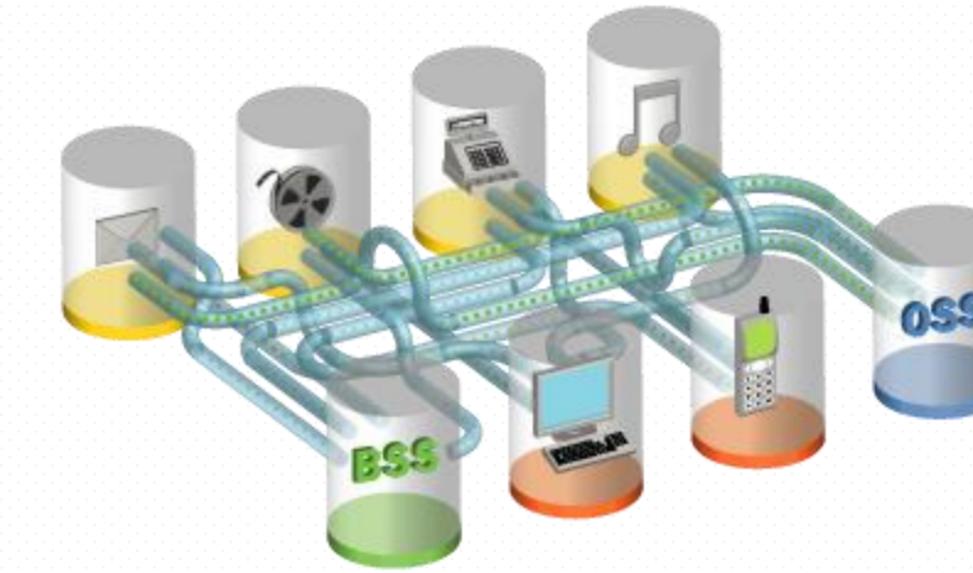
Altıgen Mimari



Altıgen Mimari



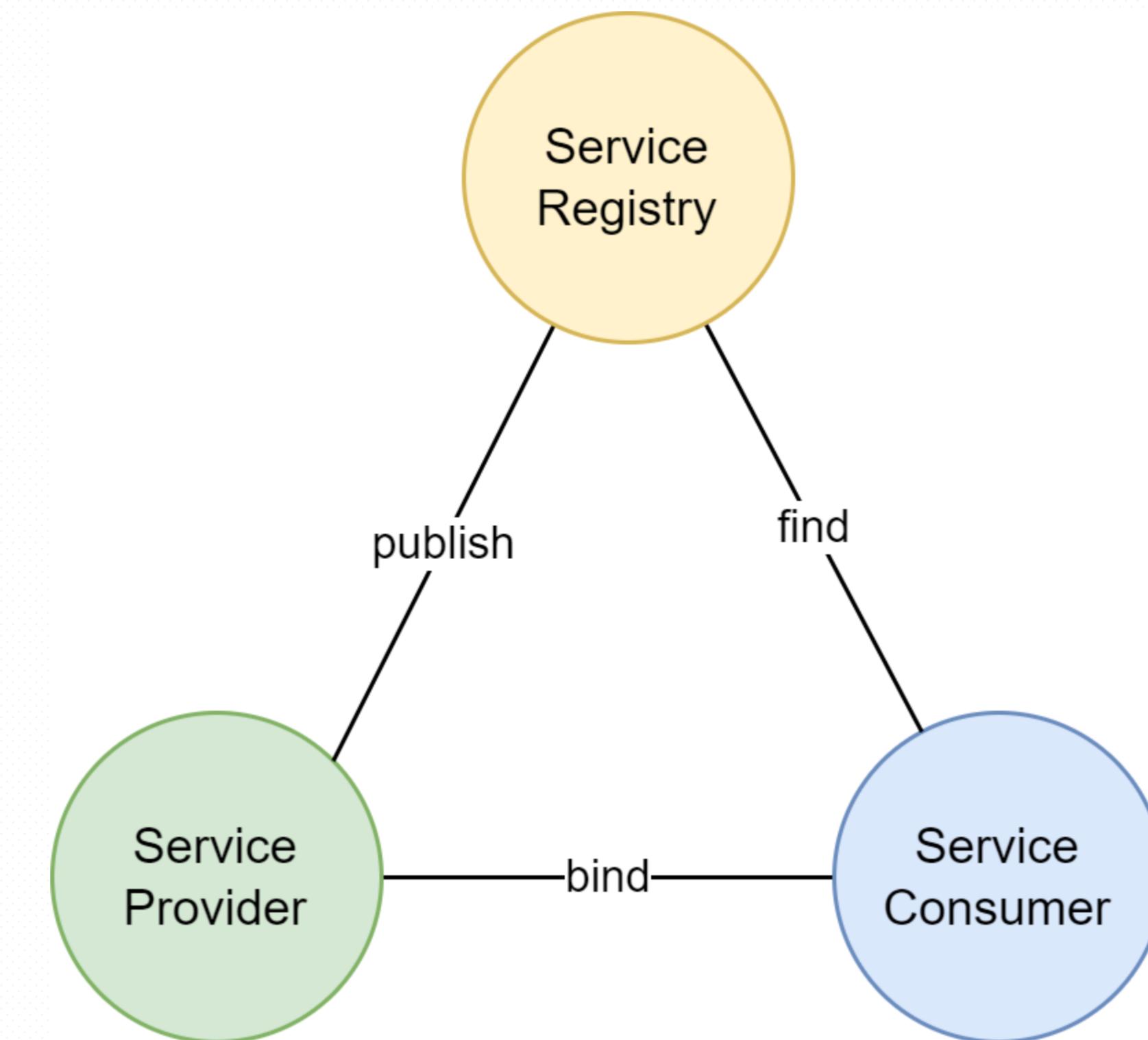
- Altıgen mimaride adaptörler, üç katmanlı/n-katmanlı/katmanlı mimarideki uygulama hizmetleri veya depo katmanlarına benzer şekilde çalışır.
- Her mikroservis, katmanlı mimari veya altıgen mimari kullanabilir.



Servis Odaklı Mimari

Service Oriented Architecture

SOA



SOA

SOA

- Hizmet odaklı mimari (SOA), iş uygulamaları oluşturmak için hizmet adı verilen yazılım bileşenlerini kullanan bir yazılım geliştirme yöntemidir.

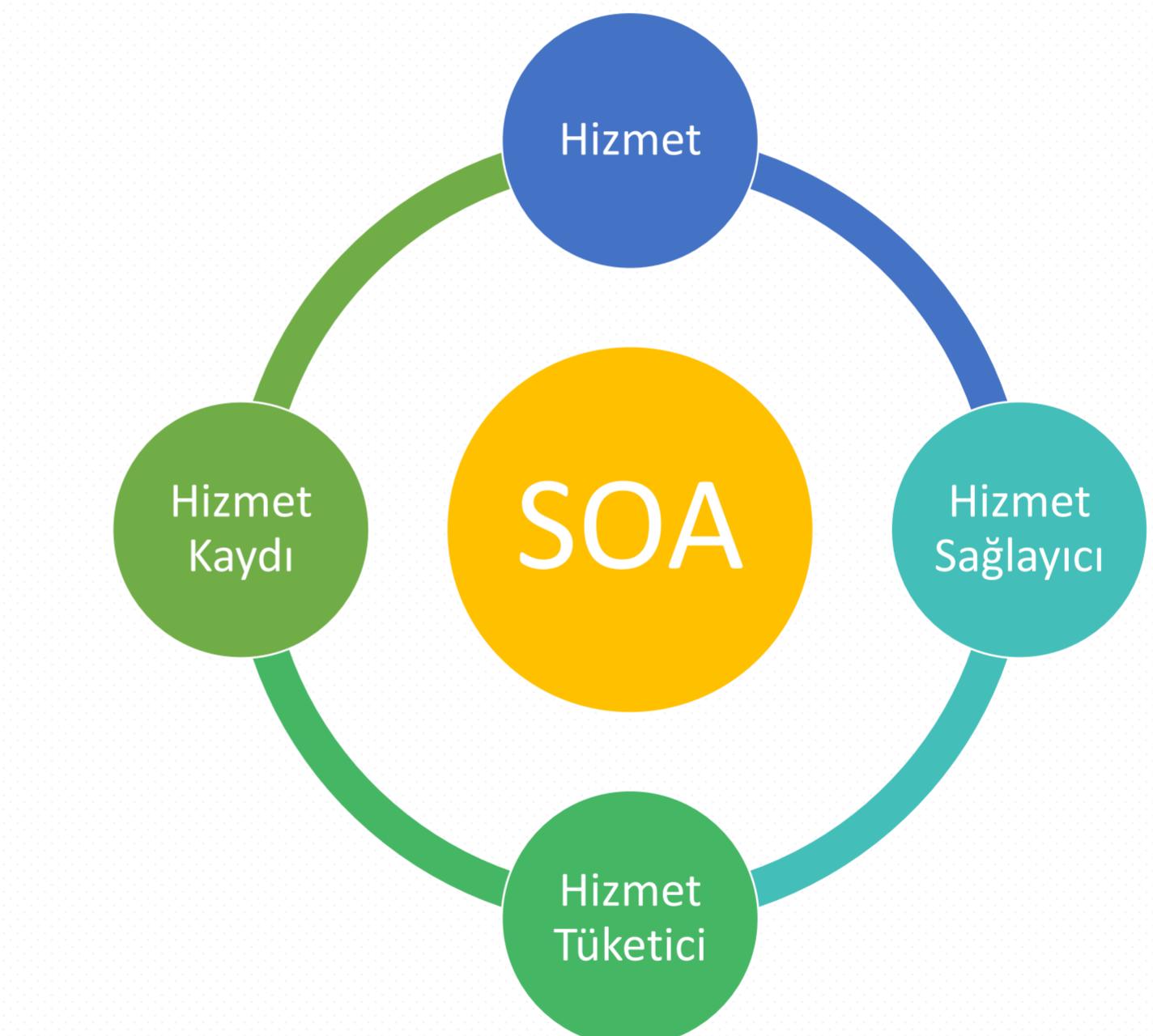
Prensipler

- Uyumluluk
- Gevşek bağlantı
- Soyutlama
- Granül yapı

Daha hızlı pazara giriş

Verimli bakım

Daha büyük uyum sağlama

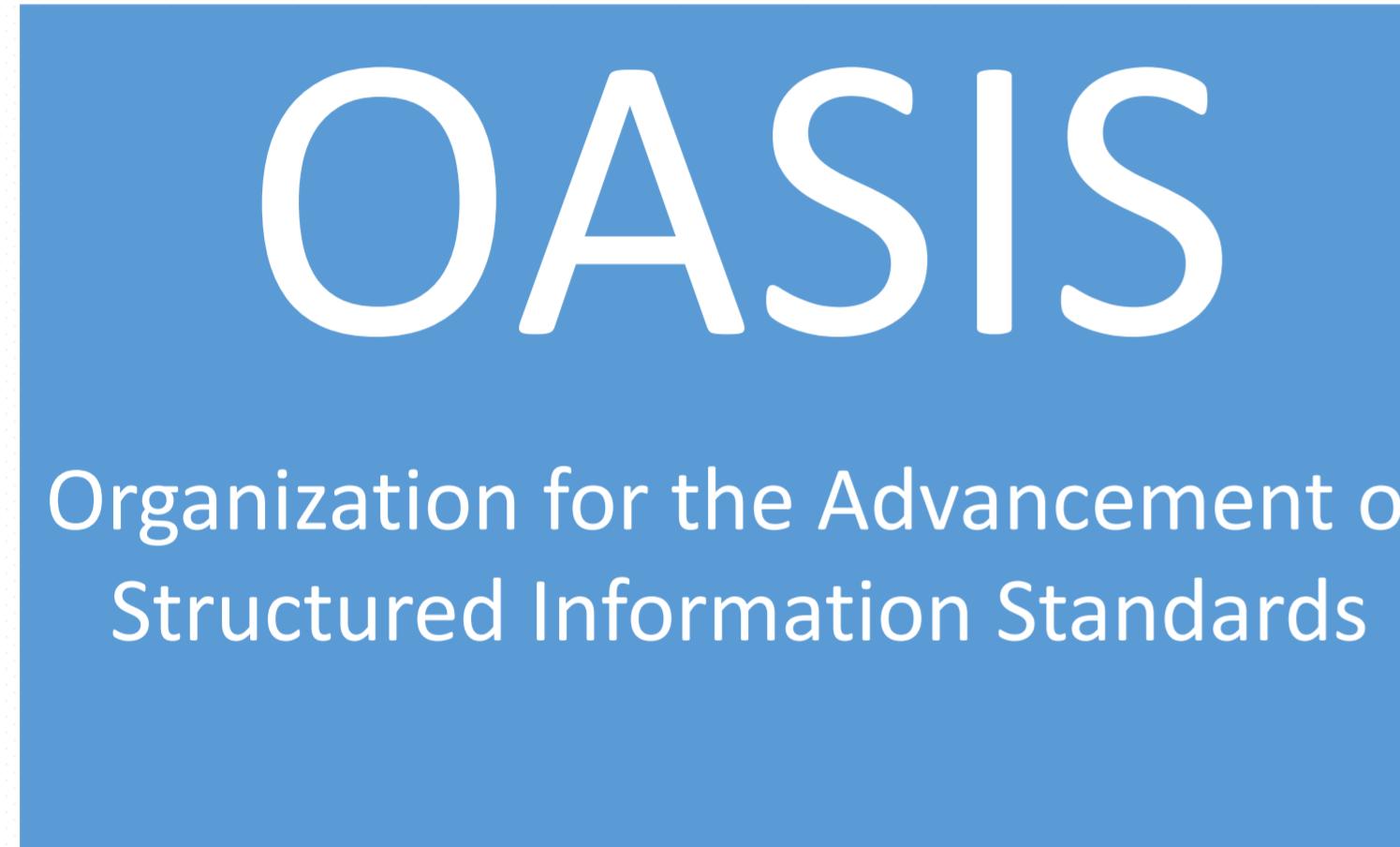


Sınırlı
ölçeklenebilirlik

Artan
bağımlılıklar

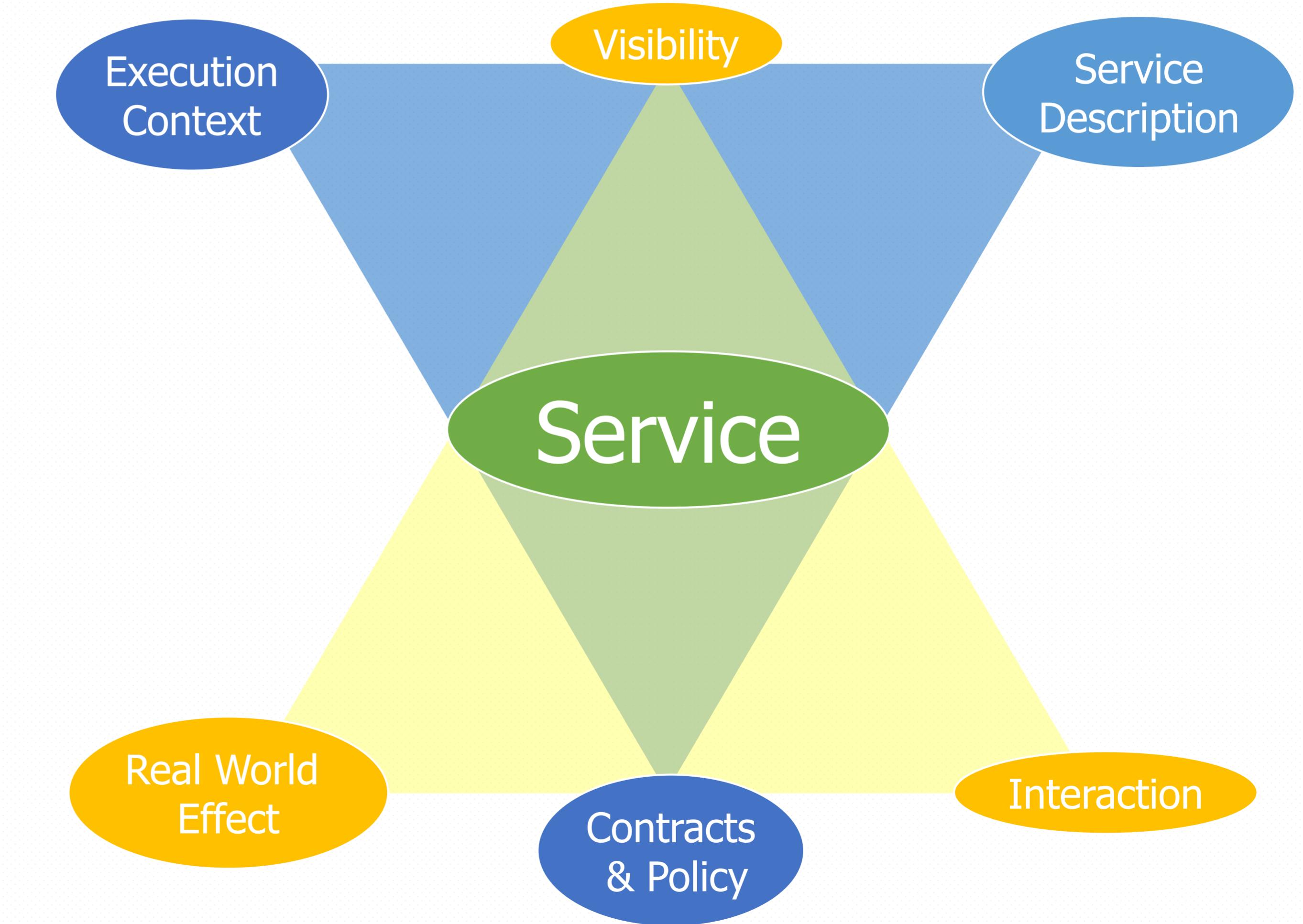
Tek nokta
problemi

SOA Referans Modeli

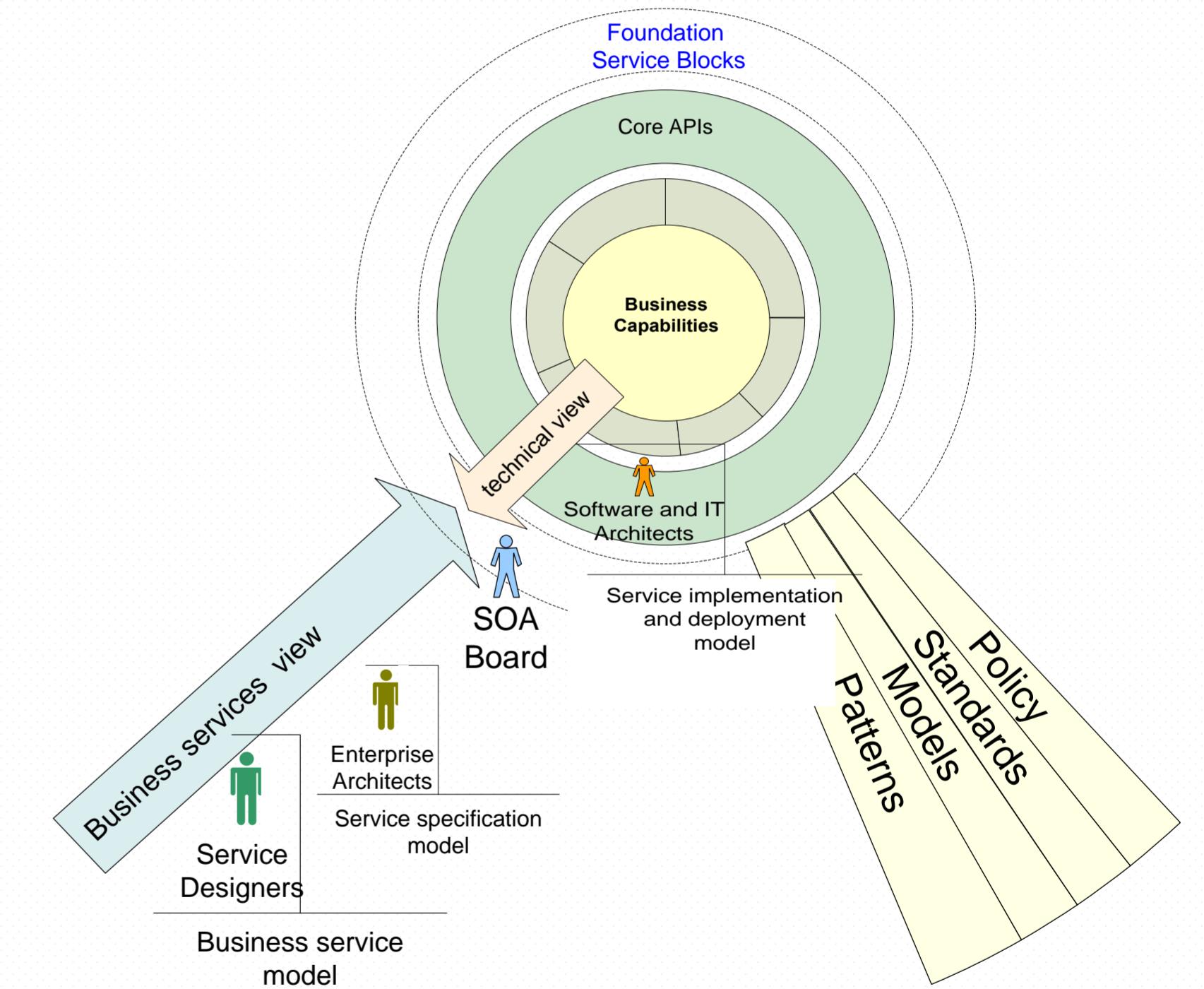
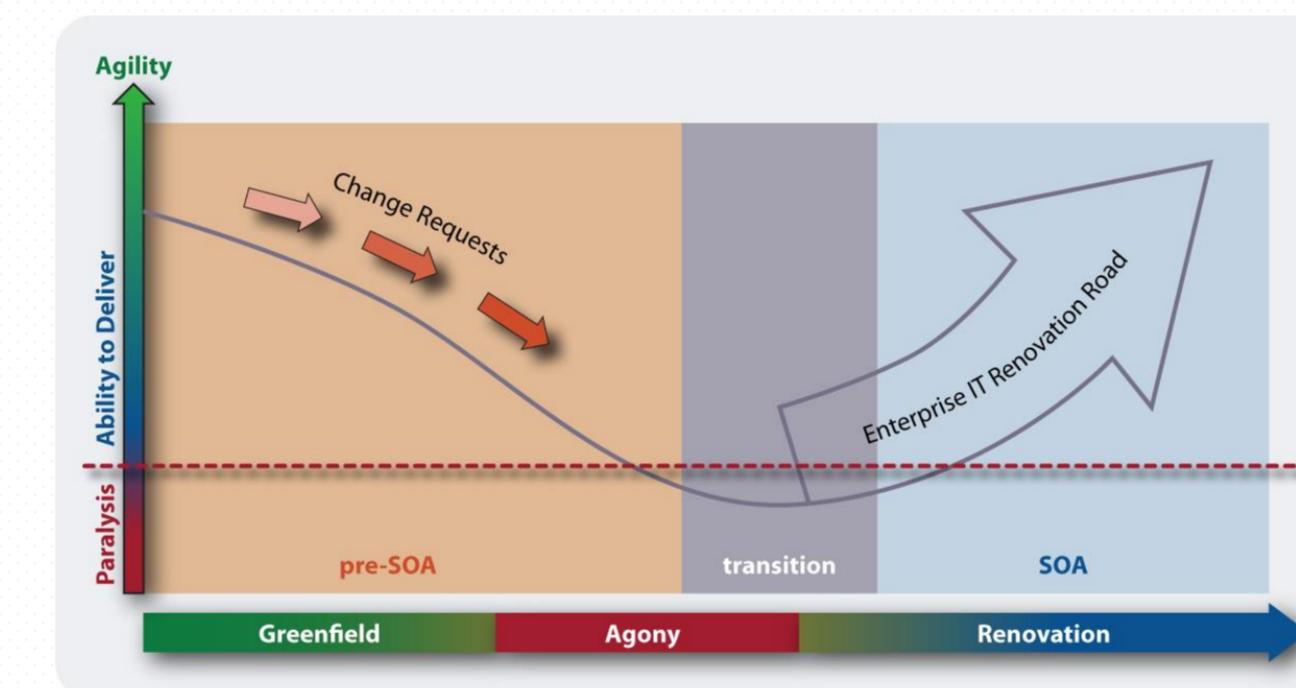
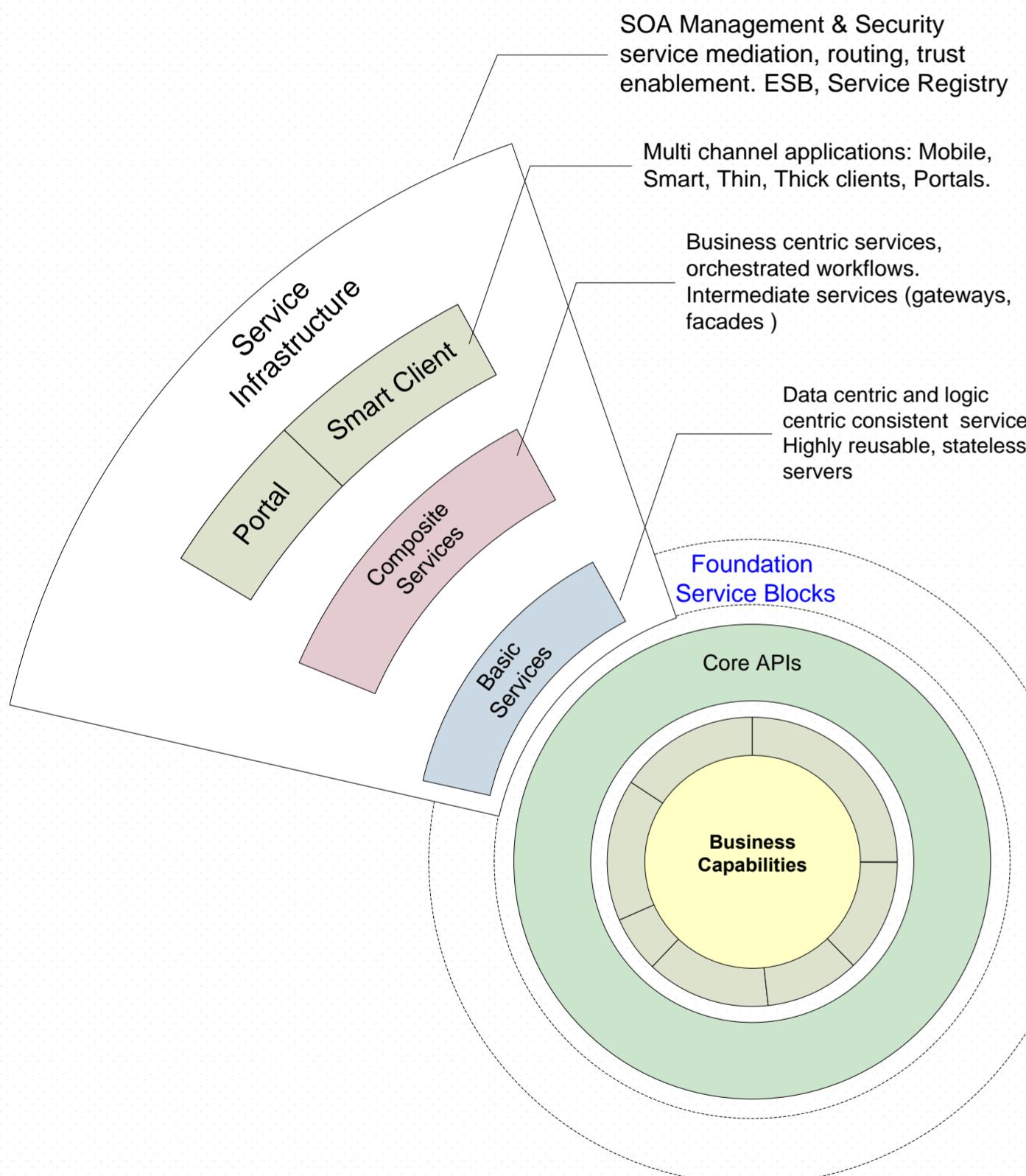


Dinamik servisler
- Görünürlük
- Hizmetler ile etkileşim
- Gerçek dünya etkileri

Servisler Hakkında
- Servis tanımları
- Politikalar ve Kontratlar
- Yürütme bağlılığı



SOA



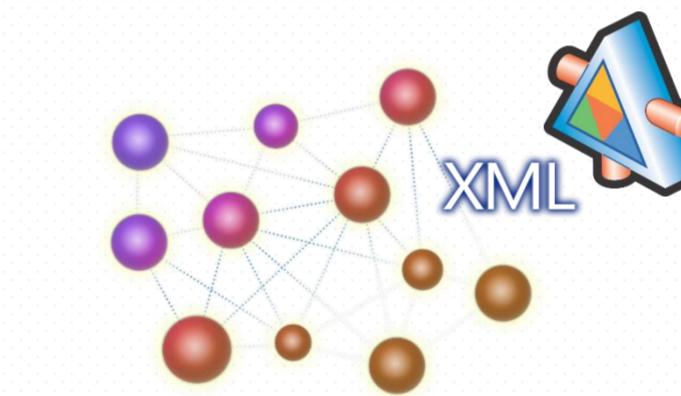
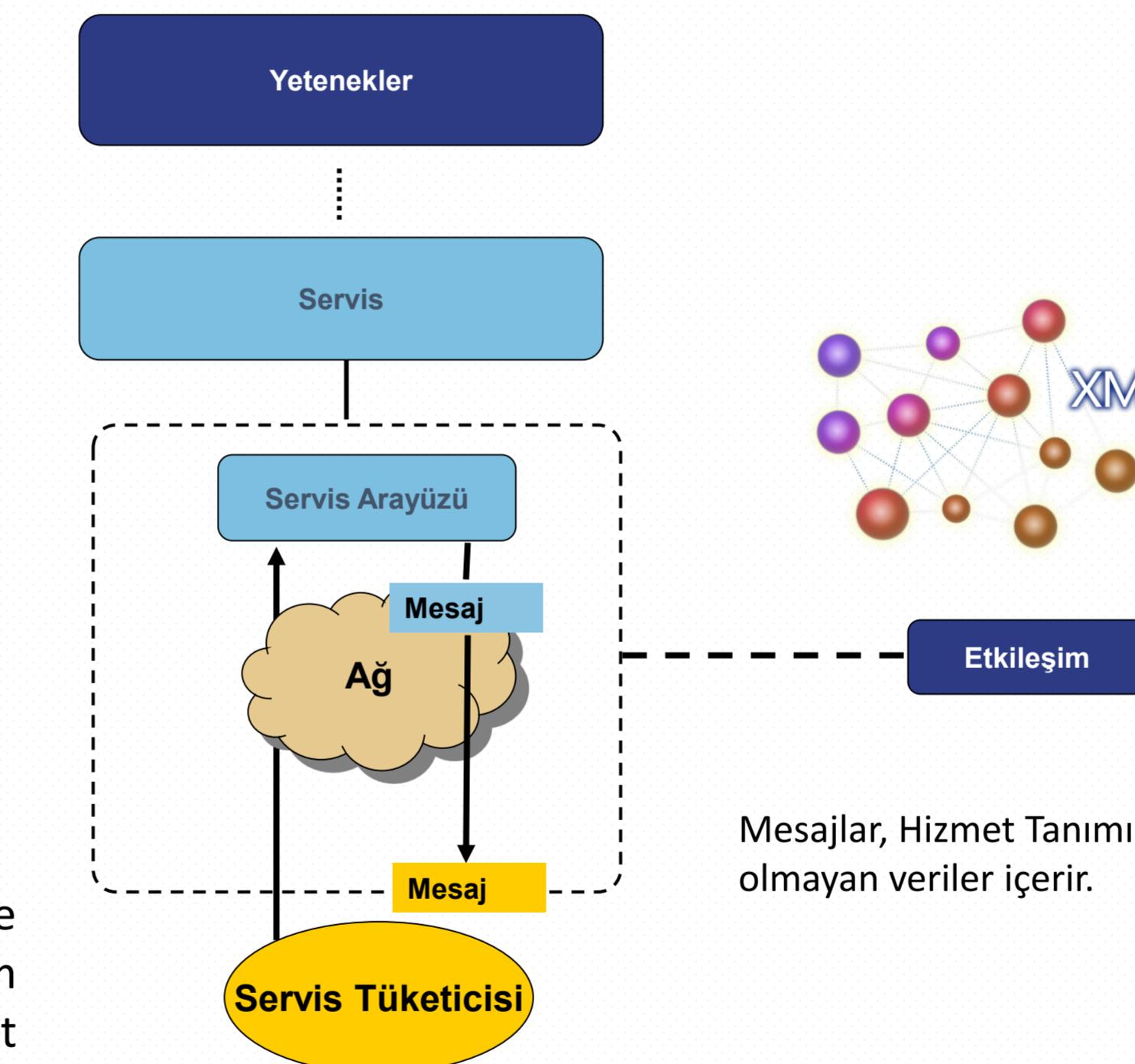
Düşünce ve Teknolojide Değişim



Tüm Servis Etkileşimleri Mesaj ile Gerçekleşir

Servis Tanımı

Depodan alınan bir dizi XML spesifikasyonu ve anlatımı. Spesifikasyonlar Mesaj(lar), Mesaj Değişim Kalıplarını, Güvenlik Gereksinimlerini, Hizmet Modelini, Hizmet Arayüzü, SLA'yi,... tanımlar.



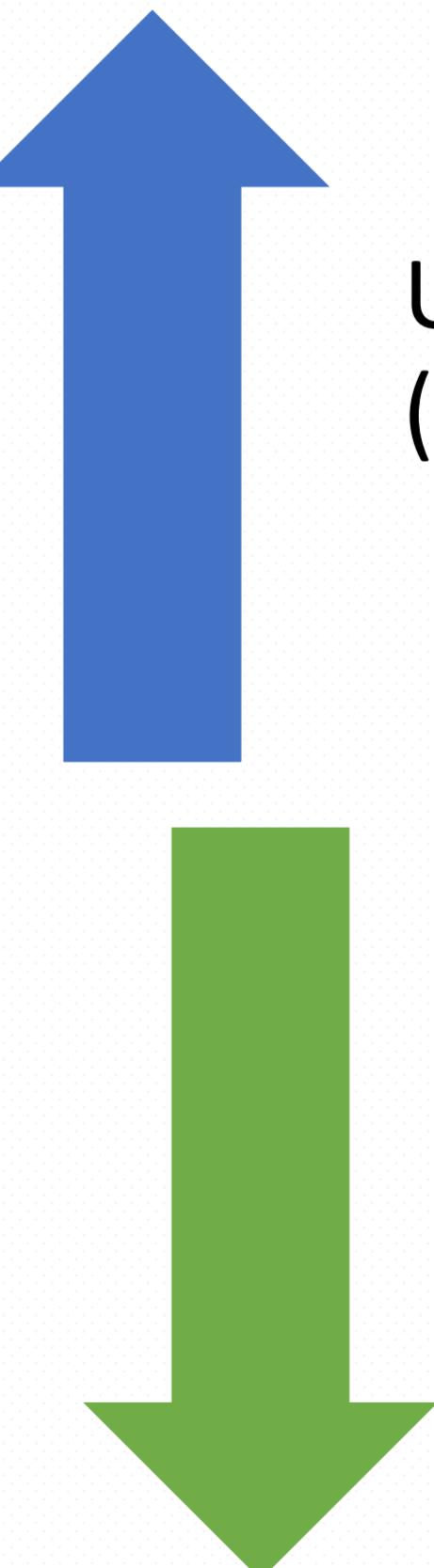
Mesajlar, Hizmet Tanımına uygun XML ve XML olmayan veriler içerir.

Alan Odaklı Tasarım

Domain Driven Design (DDD)

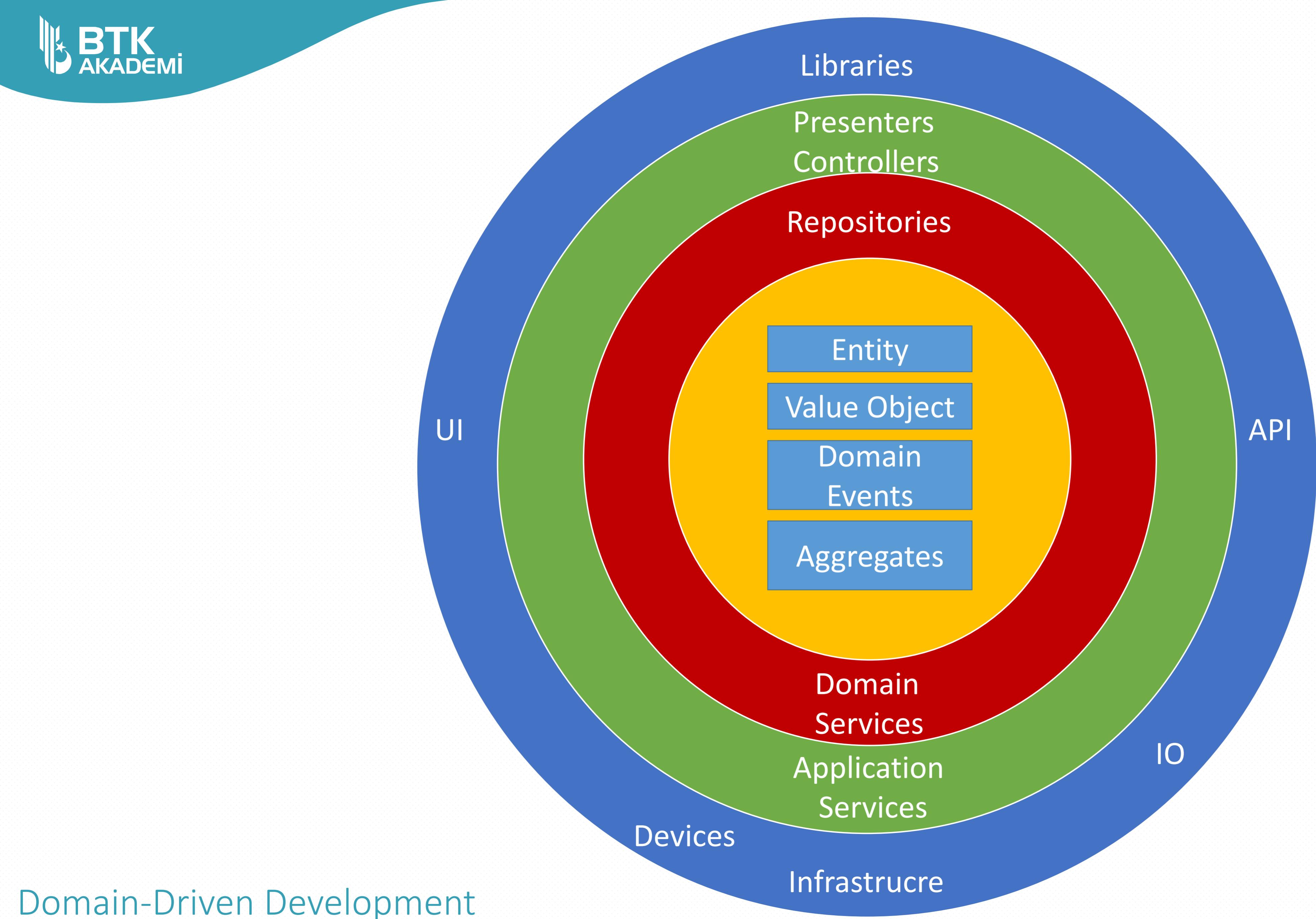
Ubiquitous Dil

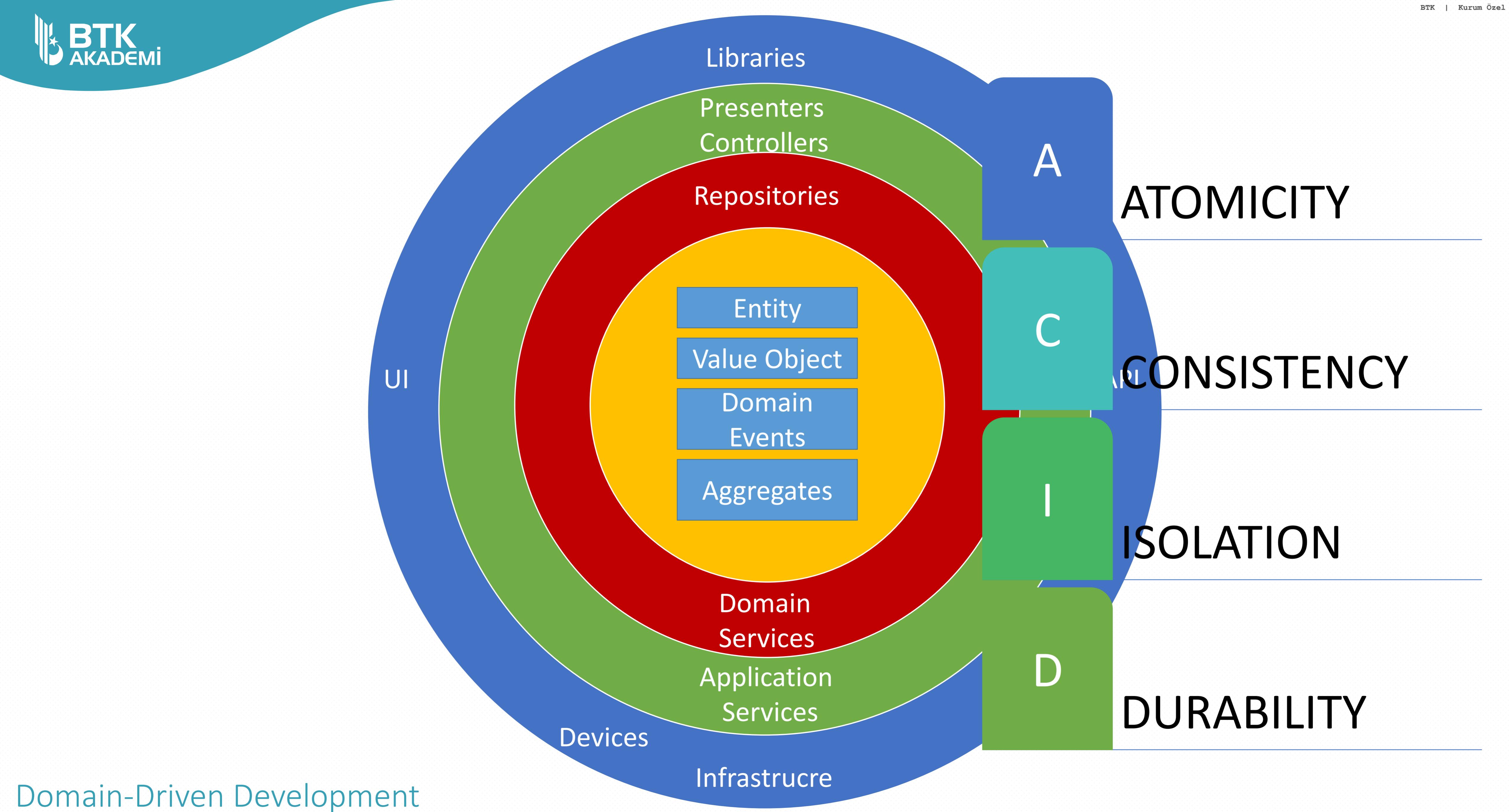
- Her yerde bulunan dil (Ubiquitous language), yazılım geliştirme süreçlerinde ve özellikle **Domain-Driven Design (DDD)** yaklaşımının bir parçası olarak önemli bir kavramdır.
- Bu kavram, yazılım geliştirme ekibinin ve iş paydaşlarının aynı terimleri, ifadeleri ve kavramları kullanarak iletişim kurmasını teşvik eder.
- Böylece, yazılım geliştirme süreci boyunca herkes aynı hızada olur ve anlam karmaşıklığı ve hatalar azalır.



Ubiquity
(Yaygınlık)

Proprietary
(Tescilli)





ACID

Atomicity

Transactionlar ya tamamı gerçekleşir ya da hiçbirini gerçekleştmez.

Consistency

Sadece geçerli verileri kayıt edilir.

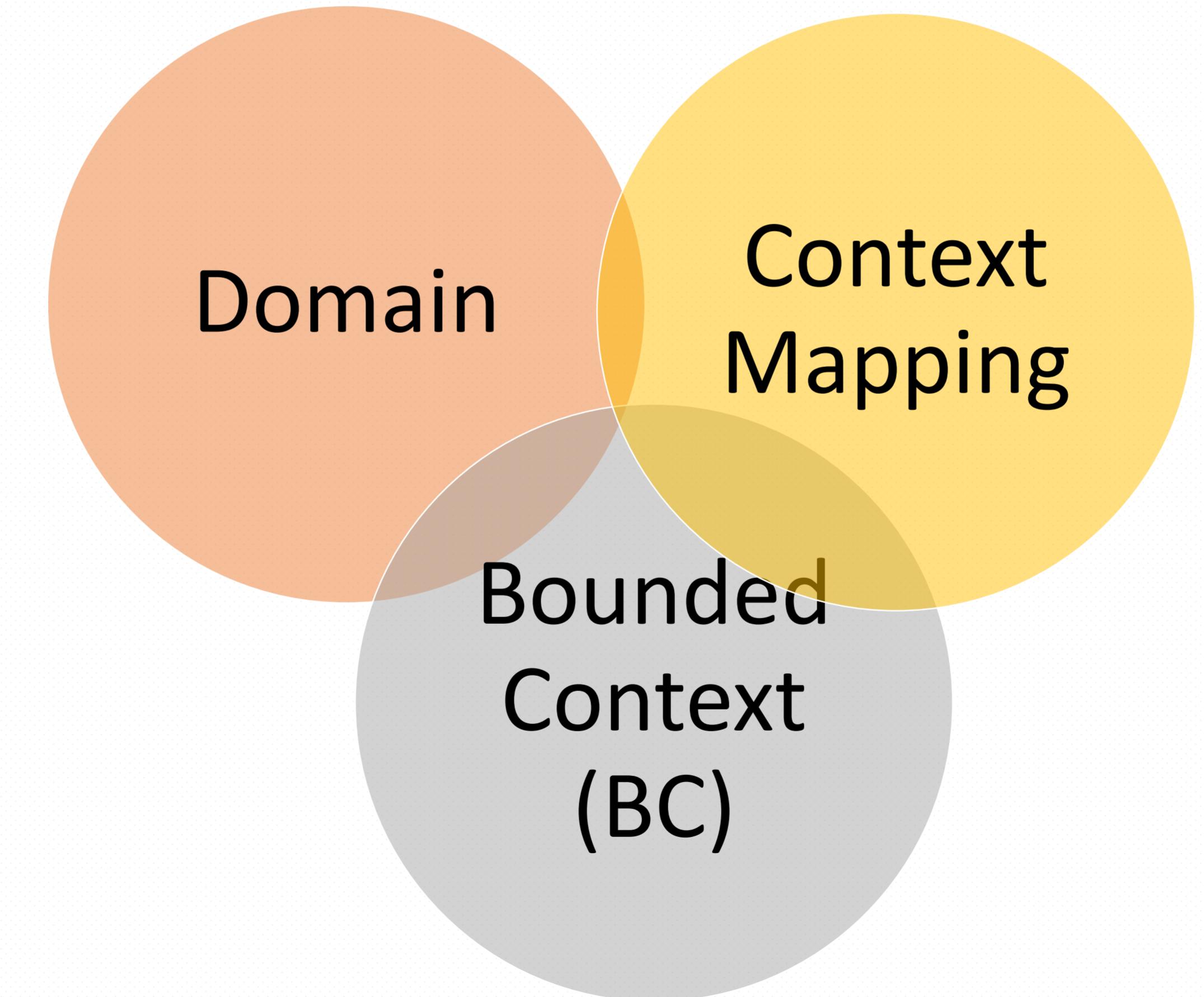
Isolation

Transactionlar bir birini etkilemez.

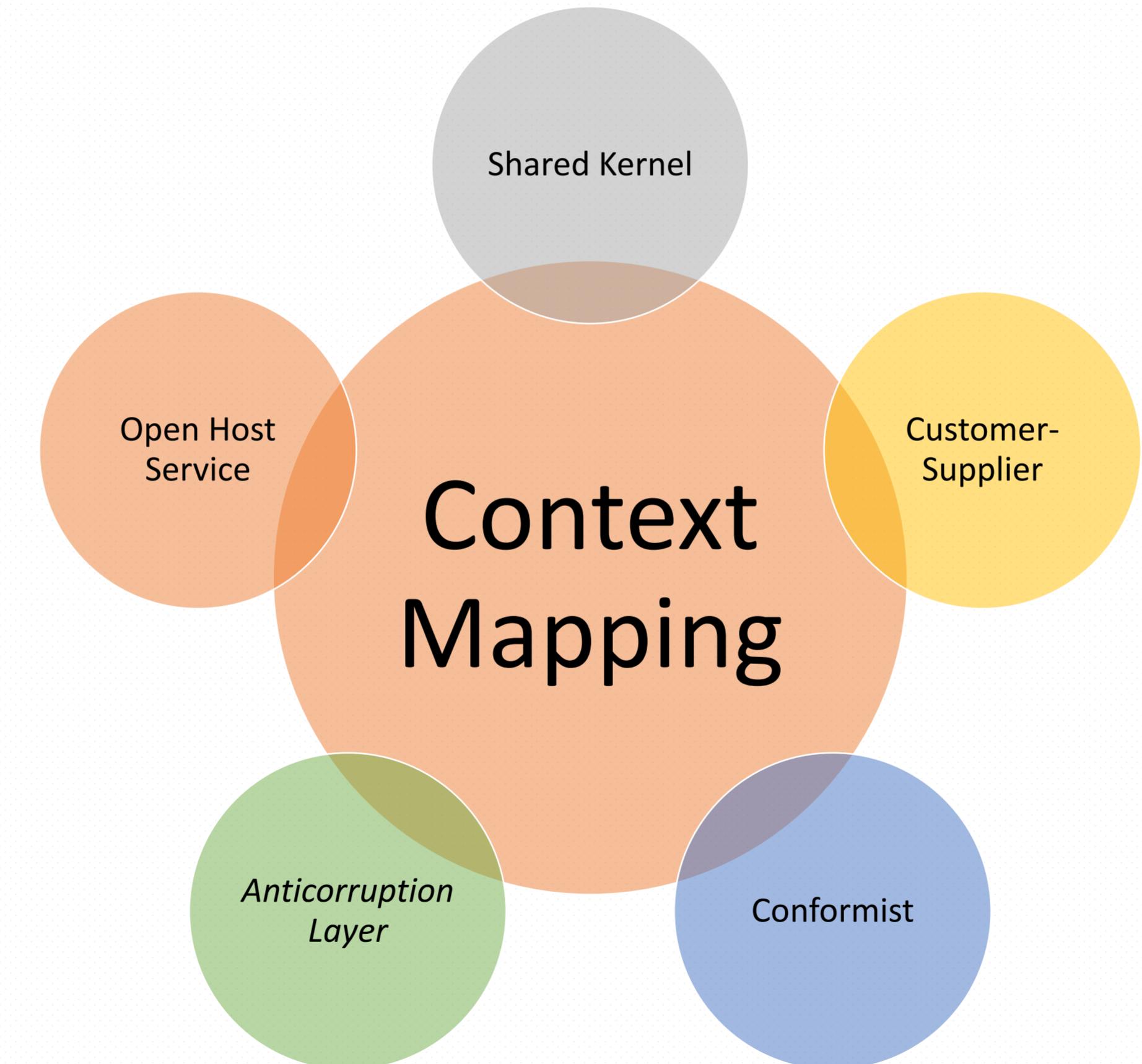
Durability

Yazılmış olan veriler kaybedilmez

Alan Odaklı Tasarım



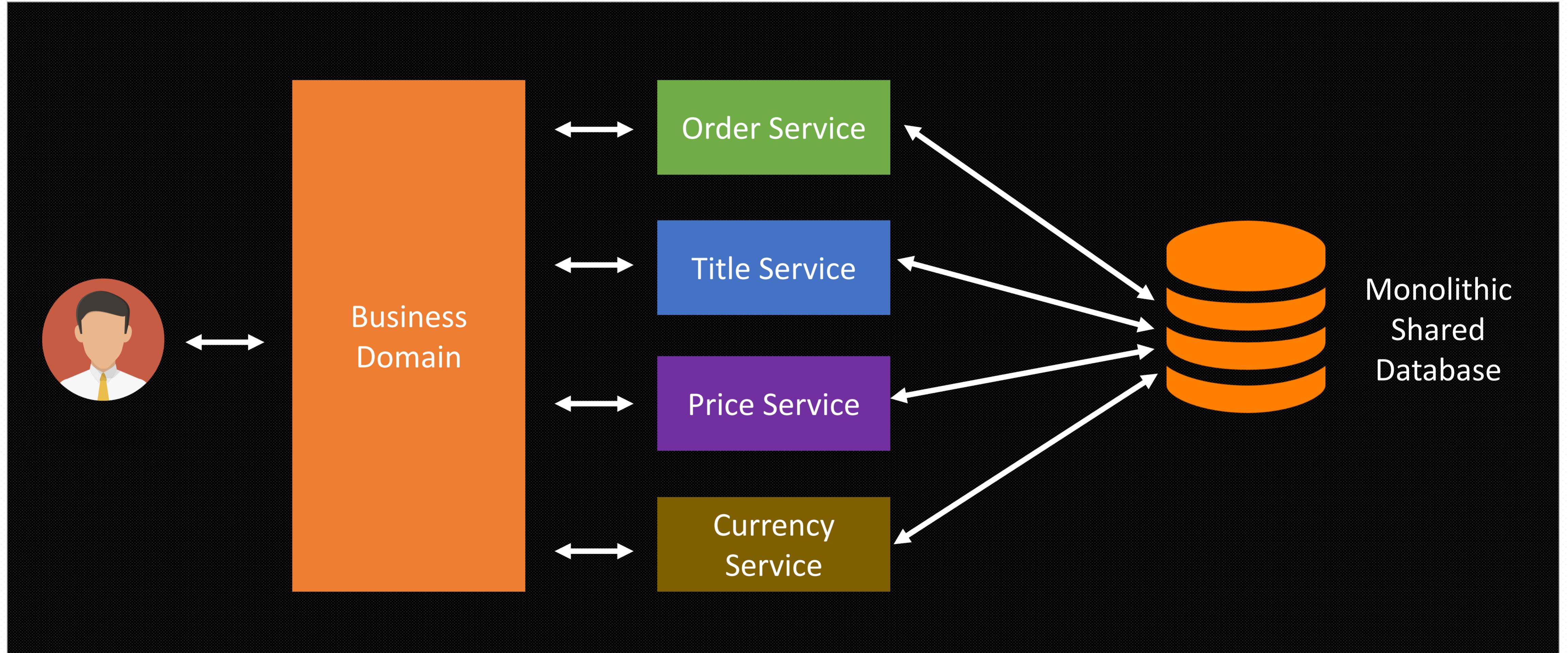
Alan Odaklı Tasarım



Monolitik Mimari ve Merkezi Veri Tabanı

Monolith Architecture – Centralized Database

Alan Odaklı Tasarım



Modüler Monolit

Modular Monolith

Modüler Monolit

(Mikro) servis sınırlarını belirlemek bir zorluk oluşturur

Alan hakkında daha fazla öğrendikçe daha kolay hale gelir.

Modül olgunlaşıkça ve onun birinci sınıf mikro servis olmaya başladığını görürsek, modüler monolit'ten kolayca çıkarabiliriz.

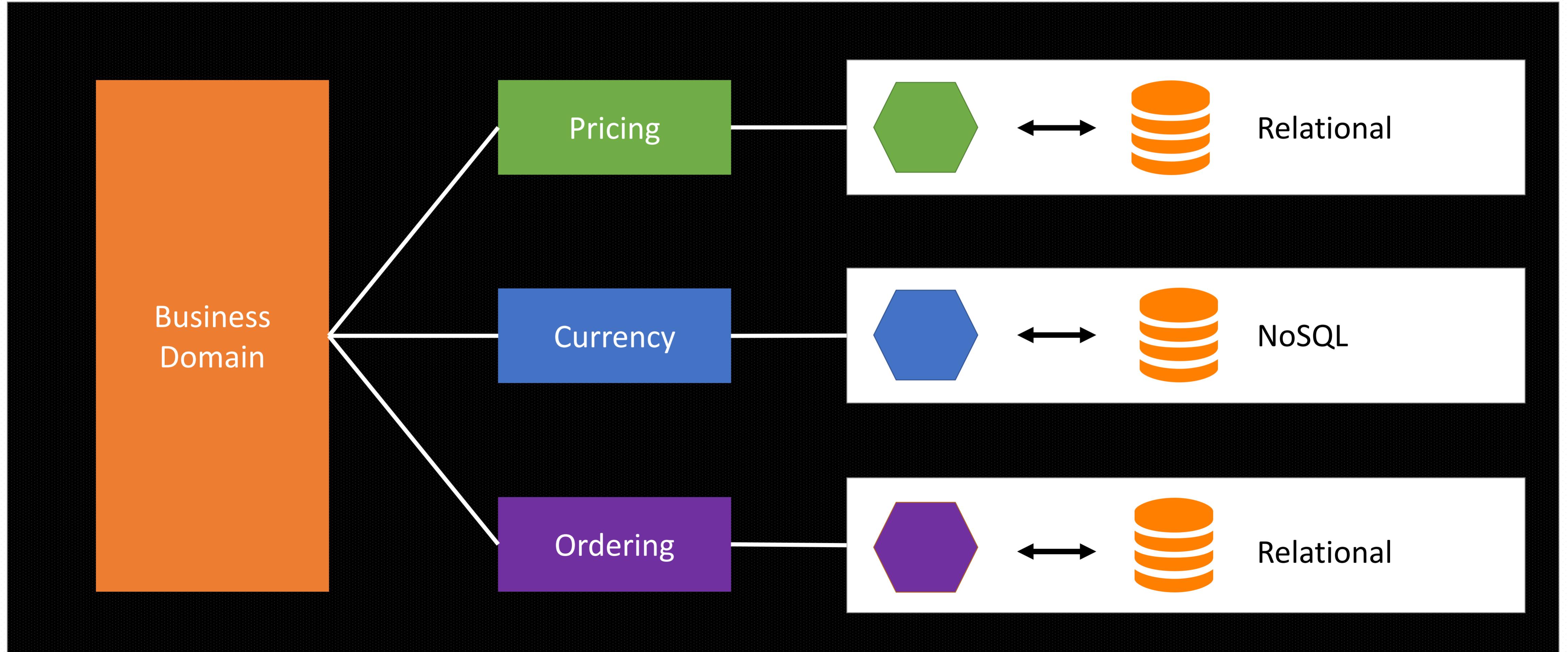
Modüler monolit bir güçlü kavramdır.

Modüler monolitlerle modül eklemek veya kaldırmak kolay ve ekonomiktir.

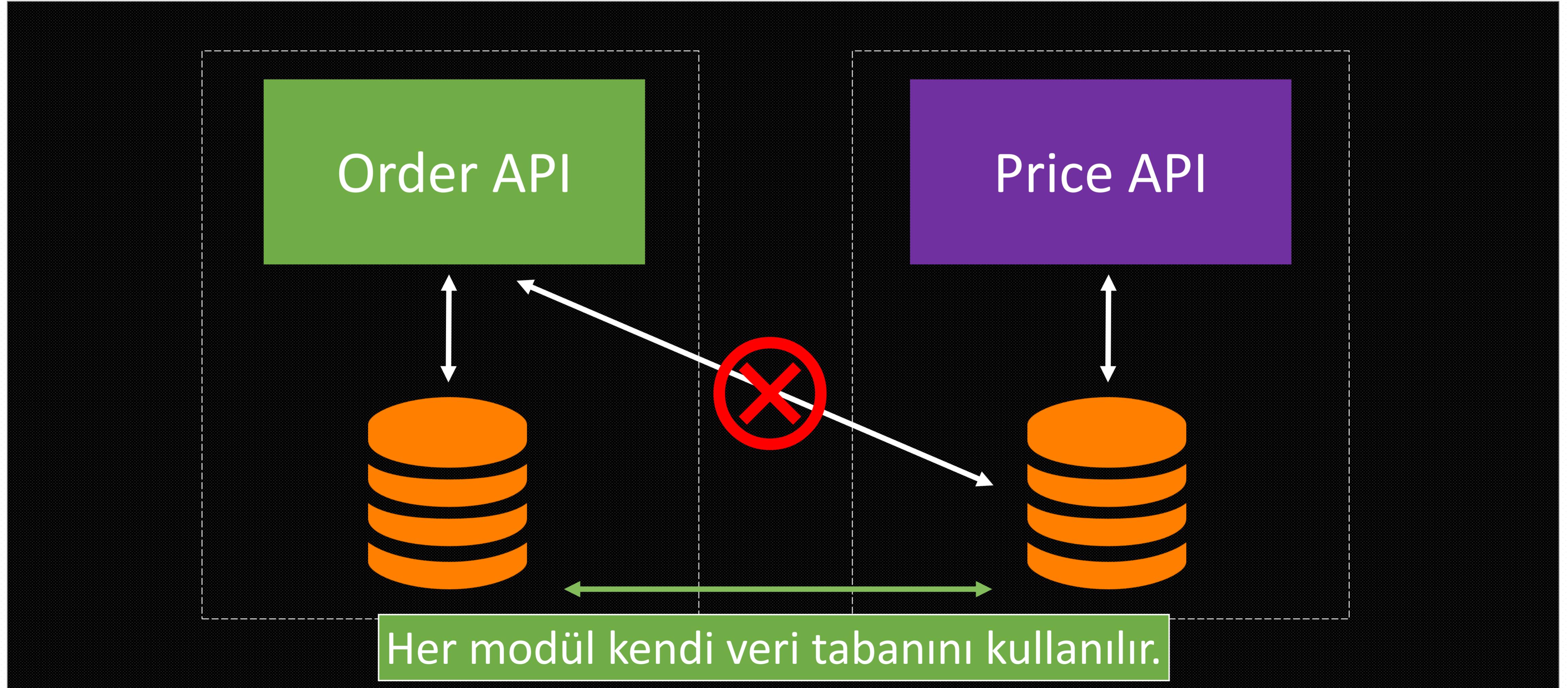
Doğru sınırları belirlemekte büyük yardımcı olur.

Her modülün kendi veri tabanı vardır.

Modüler Monolit



Modüler Monolit

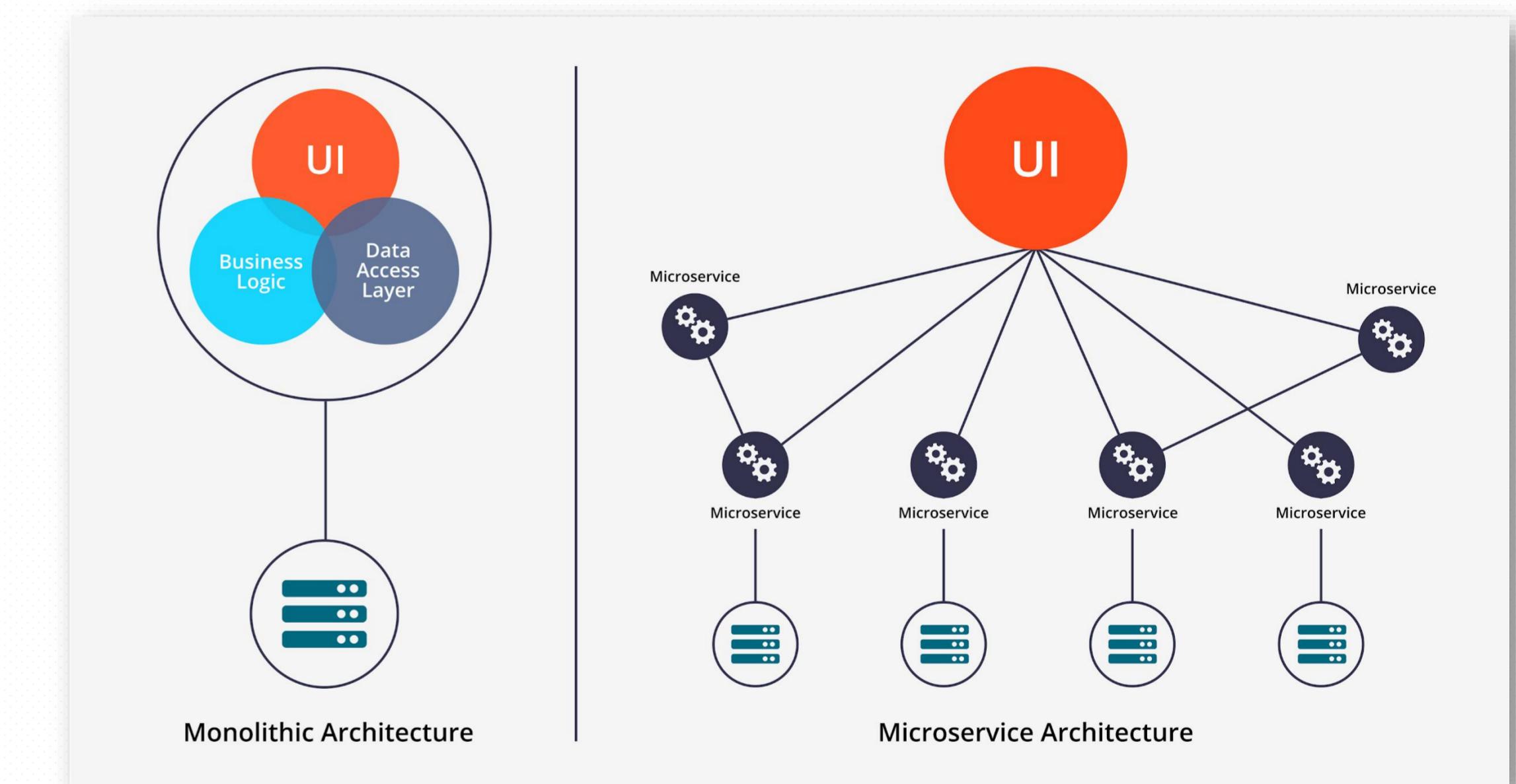


Mikroservis Mimarisi

Microservices Architecture

Mikroservis Mimarisi

- Mikroservis mimari, büyük ve karmaşık bir yazılım uygulamasını küçük, bağımsız ve özerk hizmetlere bölen bir yazılım tasarım yaklaşımıdır.
- Bu hizmetler, uygulamanın farklı işlevlerini veya bileşenlerini temsil eder ve kendi başlarına çalışabilirler.
- Mikroservis mimarisi, monolitik mimariye alternatif olarak ortaya çıkışmış ve özellikle büyük ve ölçeklenebilir uygulamaların geliştirilmesi ve sürdürülmesi için tercih edilen bir yaklaşım olmuştur.



Mikroservis Mimarisi

Bağımsızlık

Heterojen Teknoloji

Hafif İletişim

Ölçeklenebilirlik

Kolay Bakım

Hata Yalıtımı

Geliştirme Çevikliği

Özelleştirme

Hızlı Dağıtım

İşlevsel
Ayırma

Mikroservis Mimarisi

Karmaşıklık

İletişim Zorlukları

İzleme ve Hata Ayıklama

Dağıtım ve Yönetim Zorlukları

Veri Tabanı Zorlukları

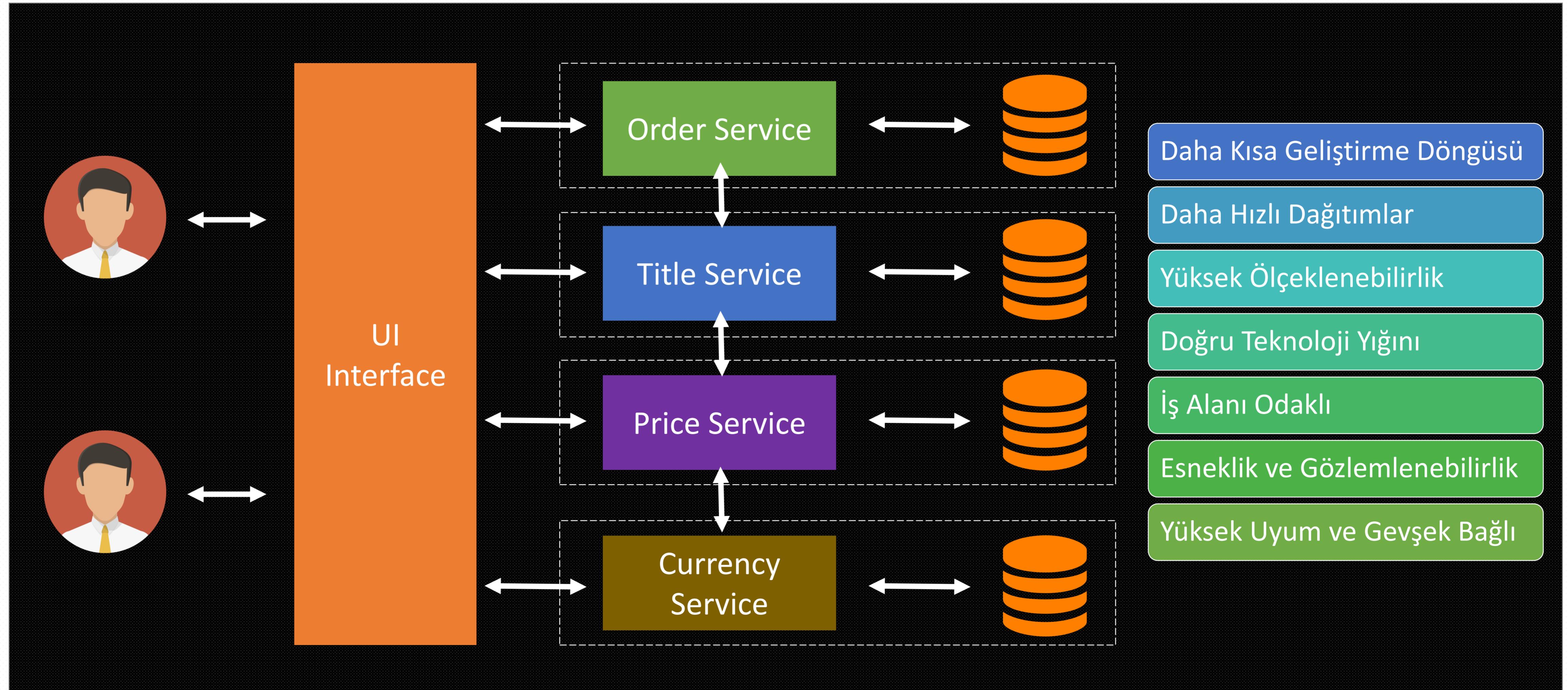
Teknoloji ve Dil Çeşitliliği

Maliyet

Güvenlik

Veri
tabanı
paylaşım
sorunları

Mikroservisler



Mikroservis Mimarısında İletişim

Senkron

Asenkron

Senkron Mesajlaşma

- REST, Thrift

Asenkron Mesajlaşma

- AMQP, STOMP, MQTT

Mesaj Formatları

- JSON, XML, Thrift, Protobuf, Avro

Mesaj Aracı

Message Broker

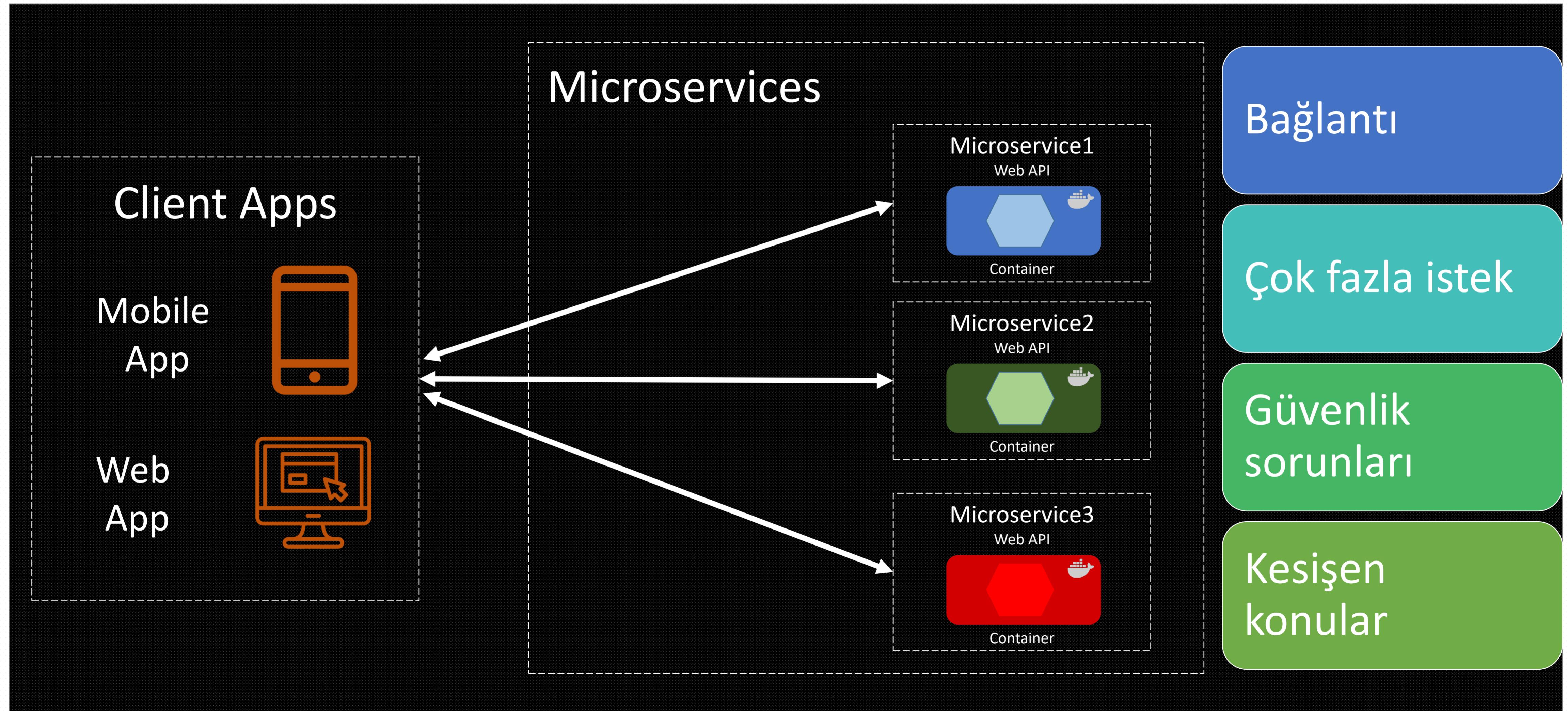
Farklı uygulamalar veya sistemler arasında iletişimi kolaylaştıran bir yazılım veya hizmettir.

Genellikle asenkron iletişim için kullanılır.

Mesajların iletilmesini, yönlendirilmesini ve işlenmesini sağlar.



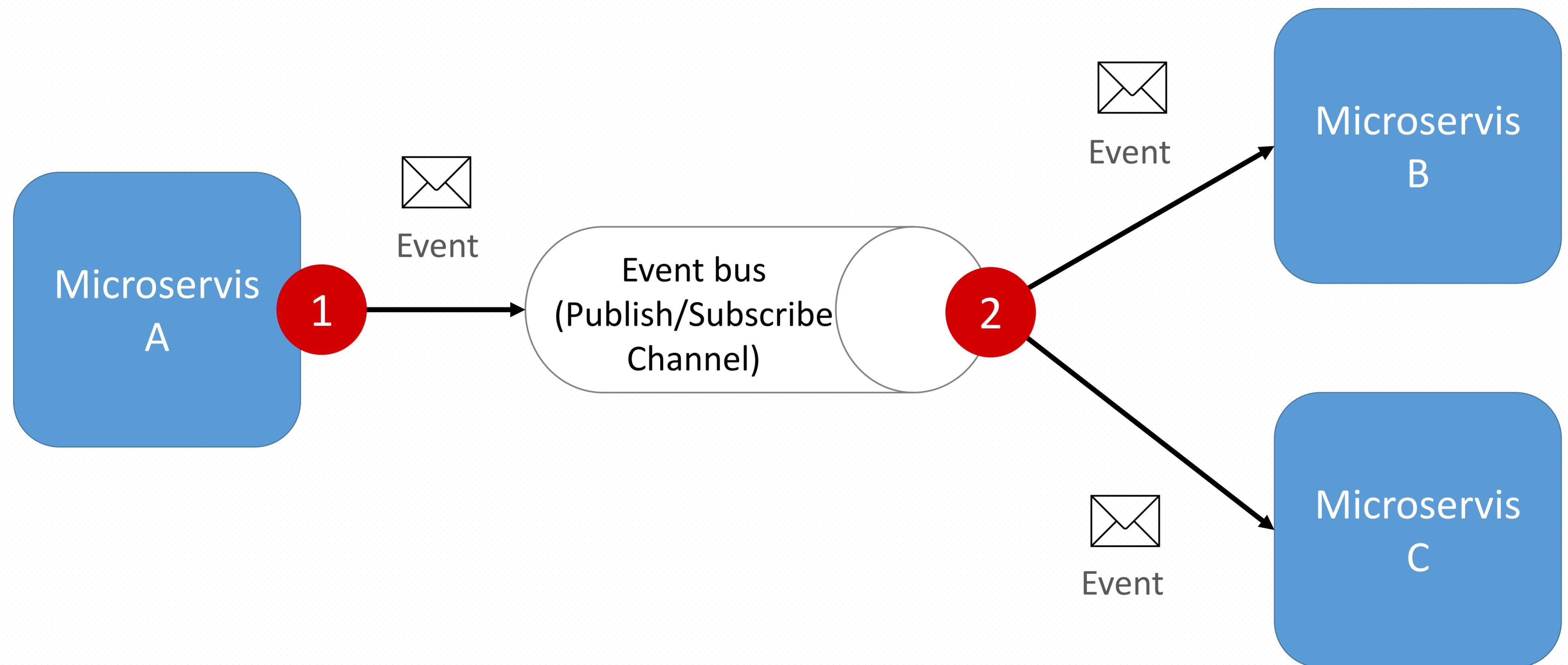
Doğrudan İstenciden Mikroservise İletişim



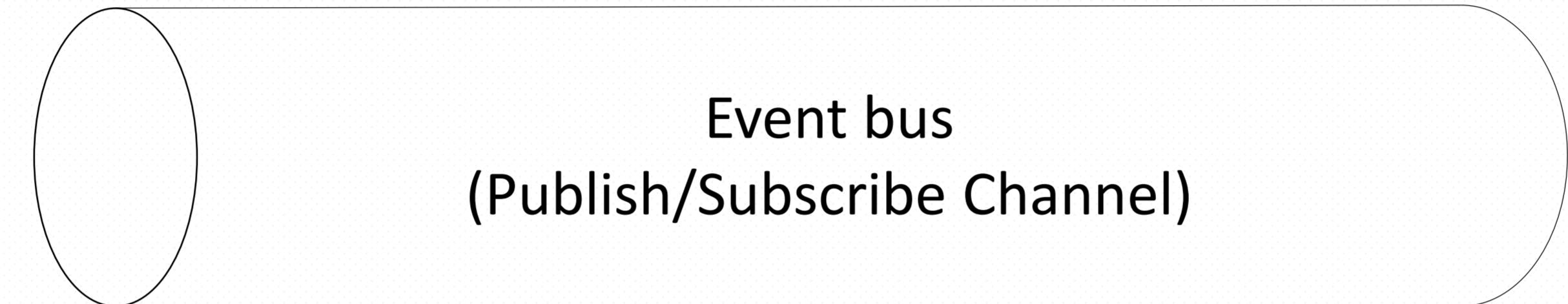
Event Bus

Olay yolu

Event bus



Event bus

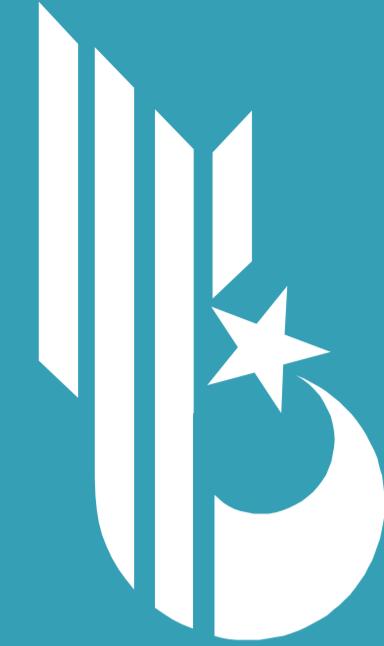


Event bus
abstraction/interface

RabbitMQ

Azure Servis
Bus

Other
message/event
broker



BTK
AKADEMİ

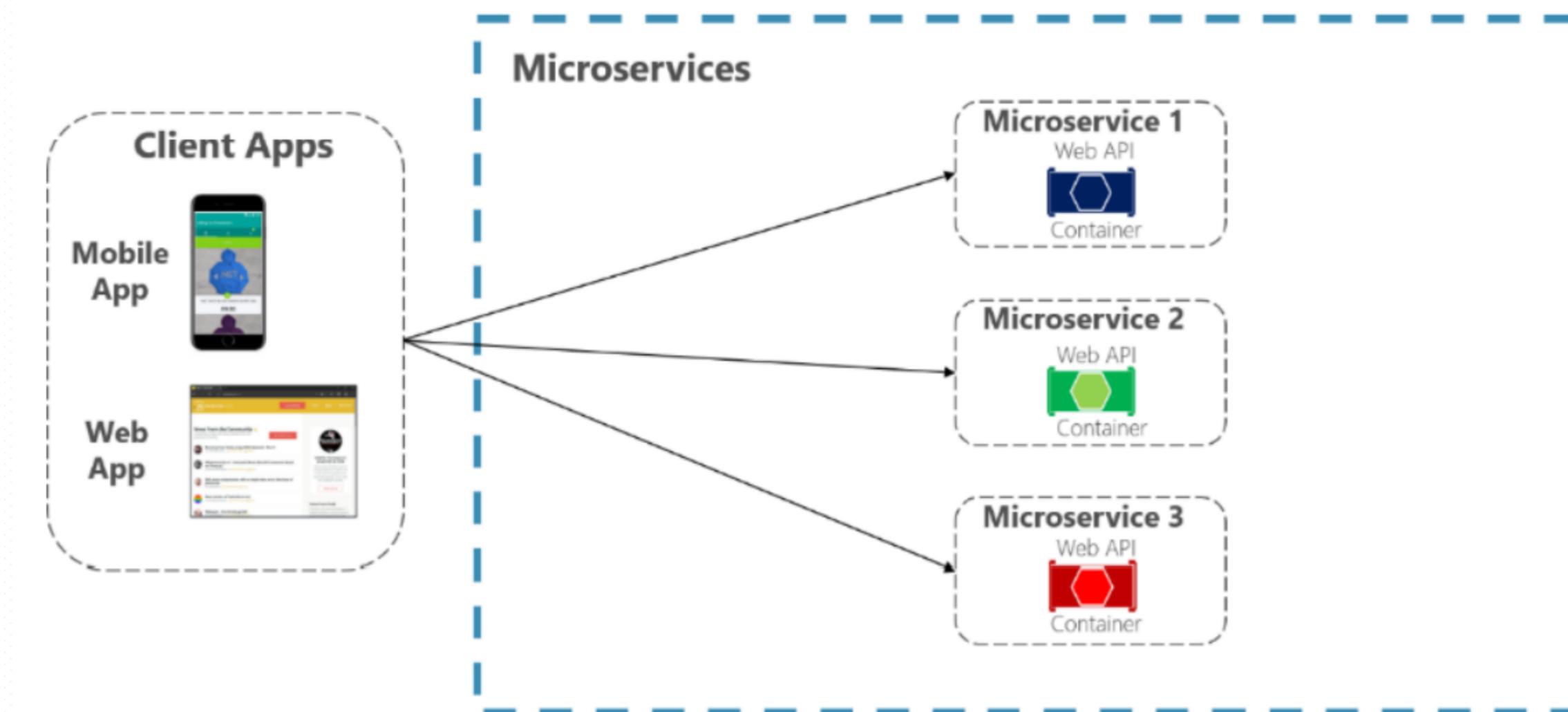
*Mikroservis mimarisinde kullanılan bir Event Bus,
mikroservisler arasında iletişimini sağlamak için kullanılan
bir araçtır.*

API Gateway

API Gateway deseni aynı zamanda bazen "ön yüz için arka uç" (*backend for frontend, BFF*) olarak da bilinir, çünkü istemci uygulamanın ihtiyaçlarını düşünerek oluşturulur.

API Gateway

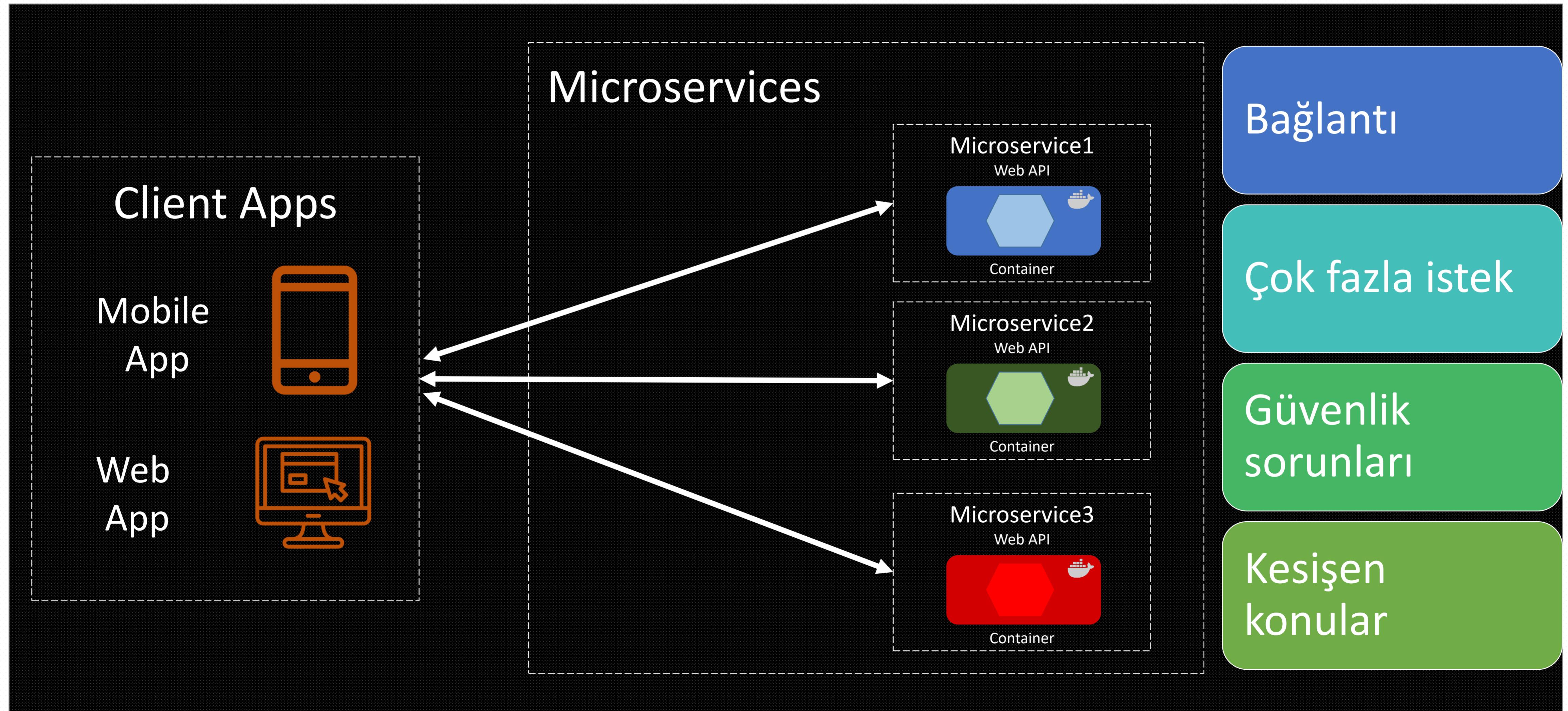
Direct Client-To-Microservice communication Architecture



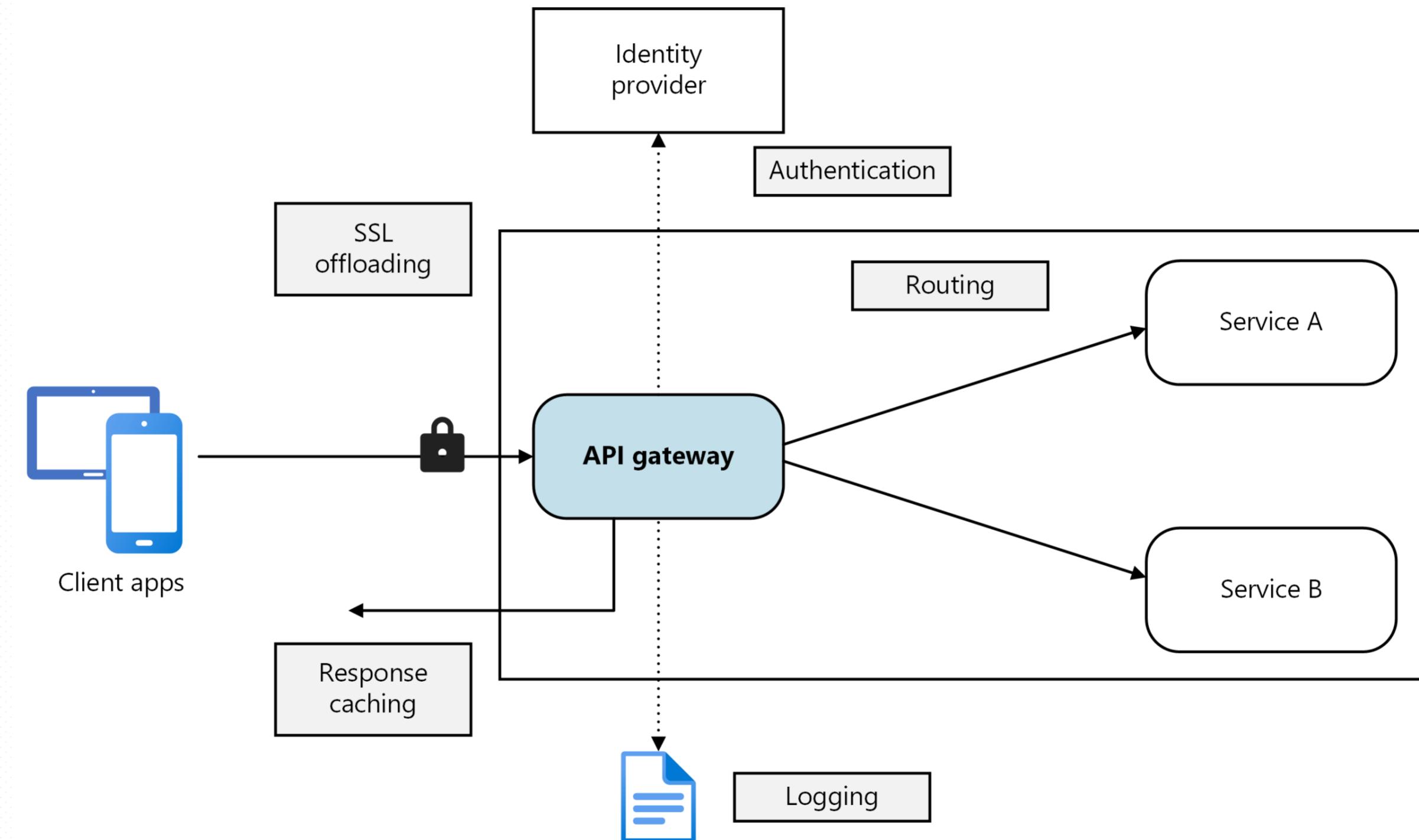
Neden doğrudan istemciden mikro hizmete
iletişim yerine API Ağ Geçitlerini düşünmeliiniz?

Why consider API Gateways instead of direct client-to-microservice
communication?

Doğrudan İstenciden Mikroservise İletişim



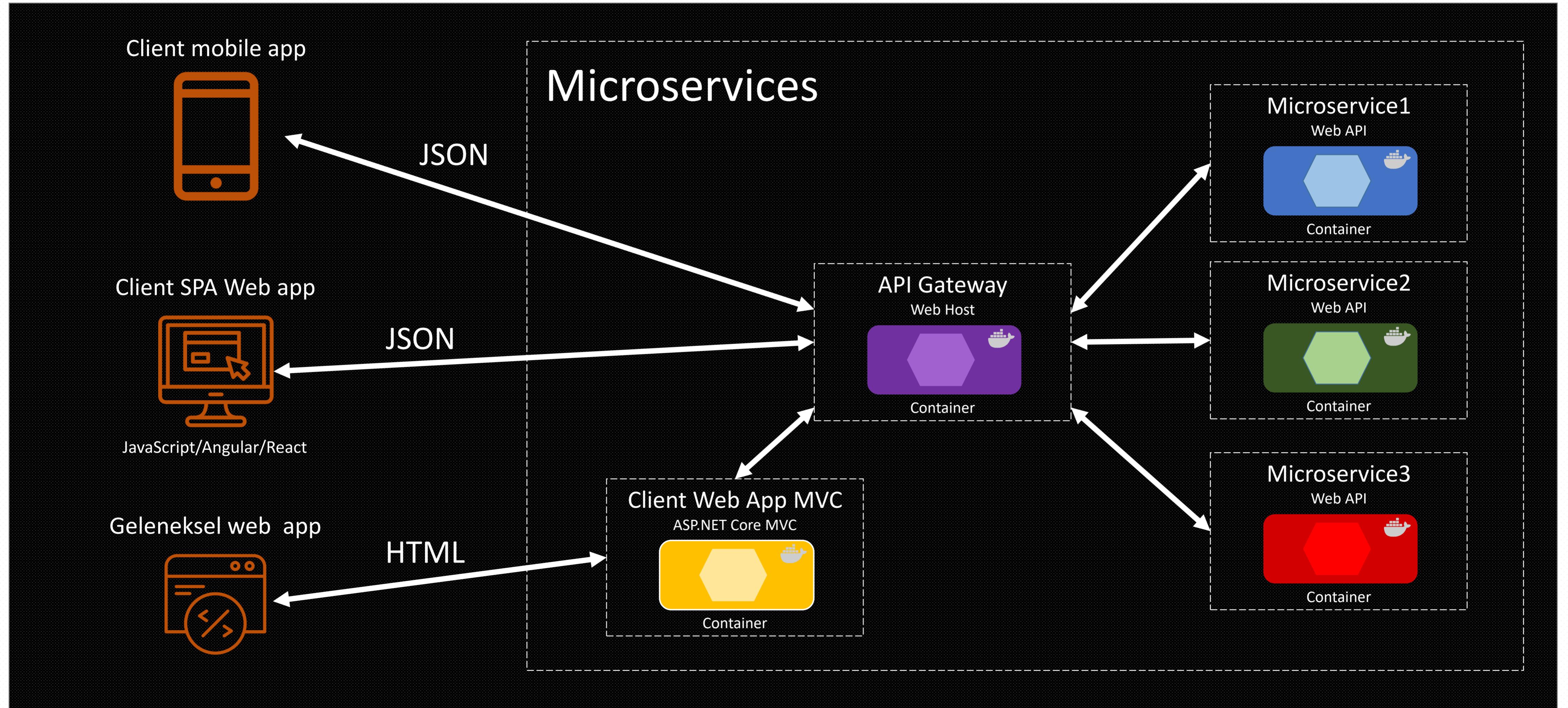
API Gateway



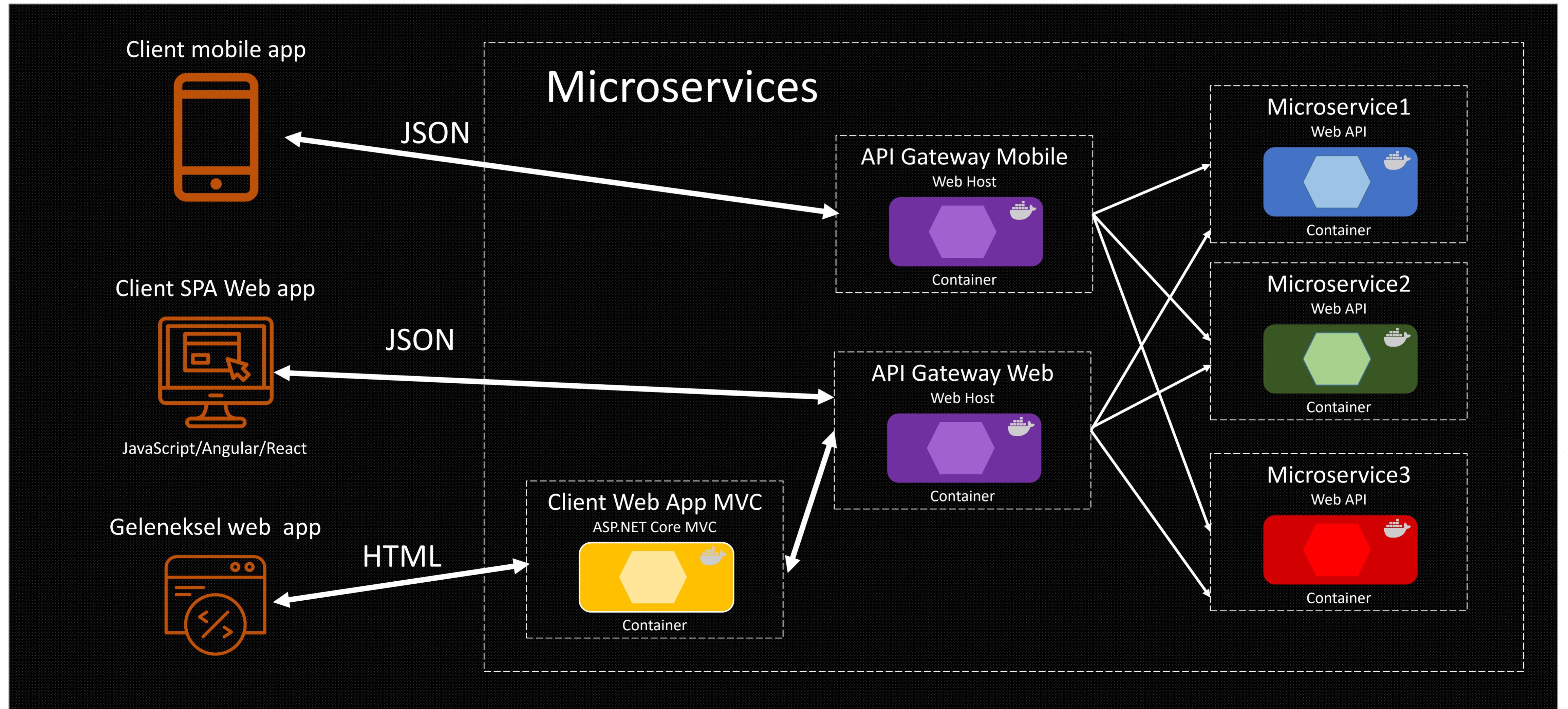
API Gateway

- API Gateway desenini uygularken dikkatli olmanız gerekiyor.
- Genellikle tüm iç mikro hizmetlerinizi bir araya getiren tek bir [API Gateway](#)'e sahip olmak iyi bir fikir değildir.
- Bu durumda, bir monolitik birleştirici veya düzenleyici gibi davranır ve tüm mikro hizmetleri birbirine bağlayarak mikro hizmet bağımsızlığını ihlal eder.

API Gateway



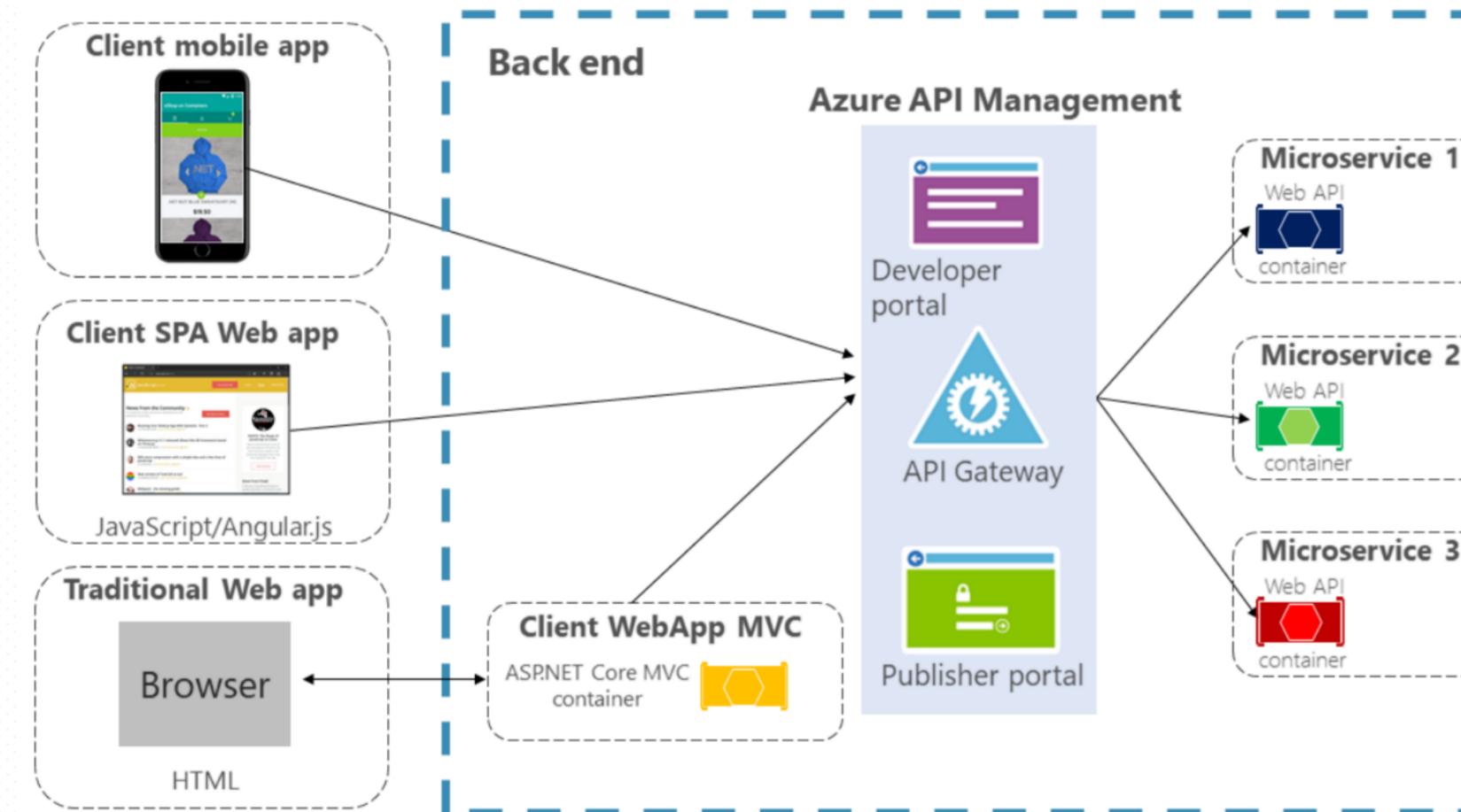
API Gateway



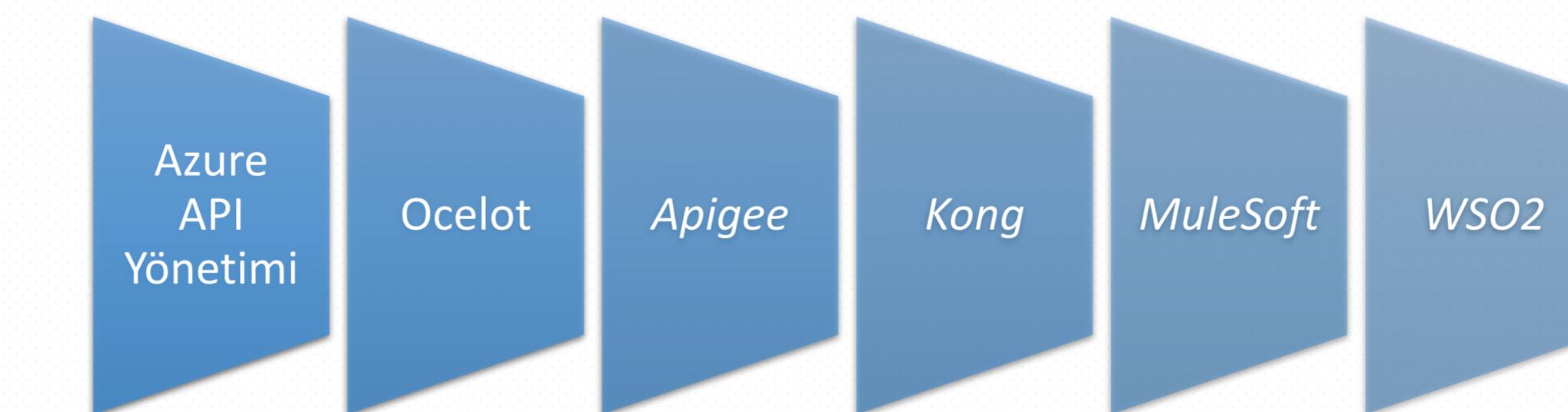
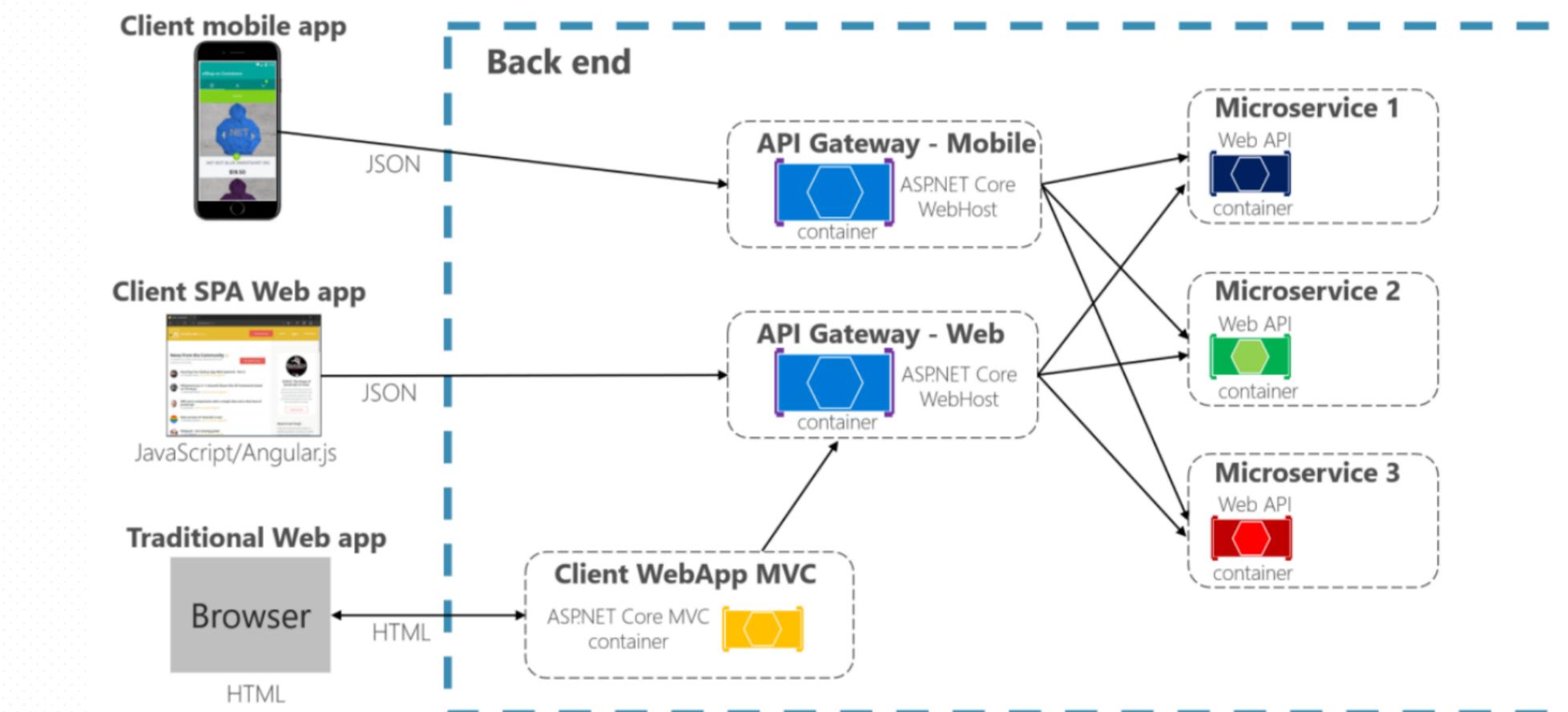
API Gateway

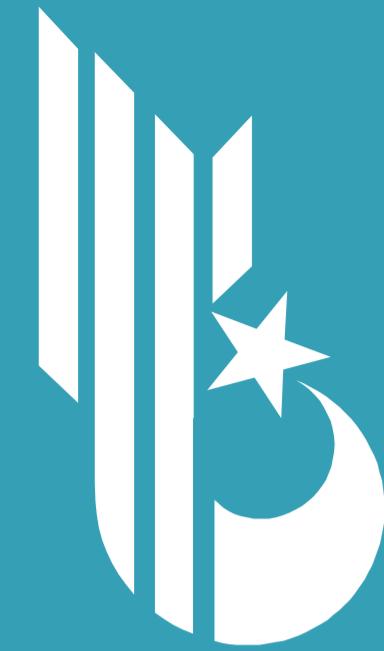
API Gateway with Azure API Management

Architecture



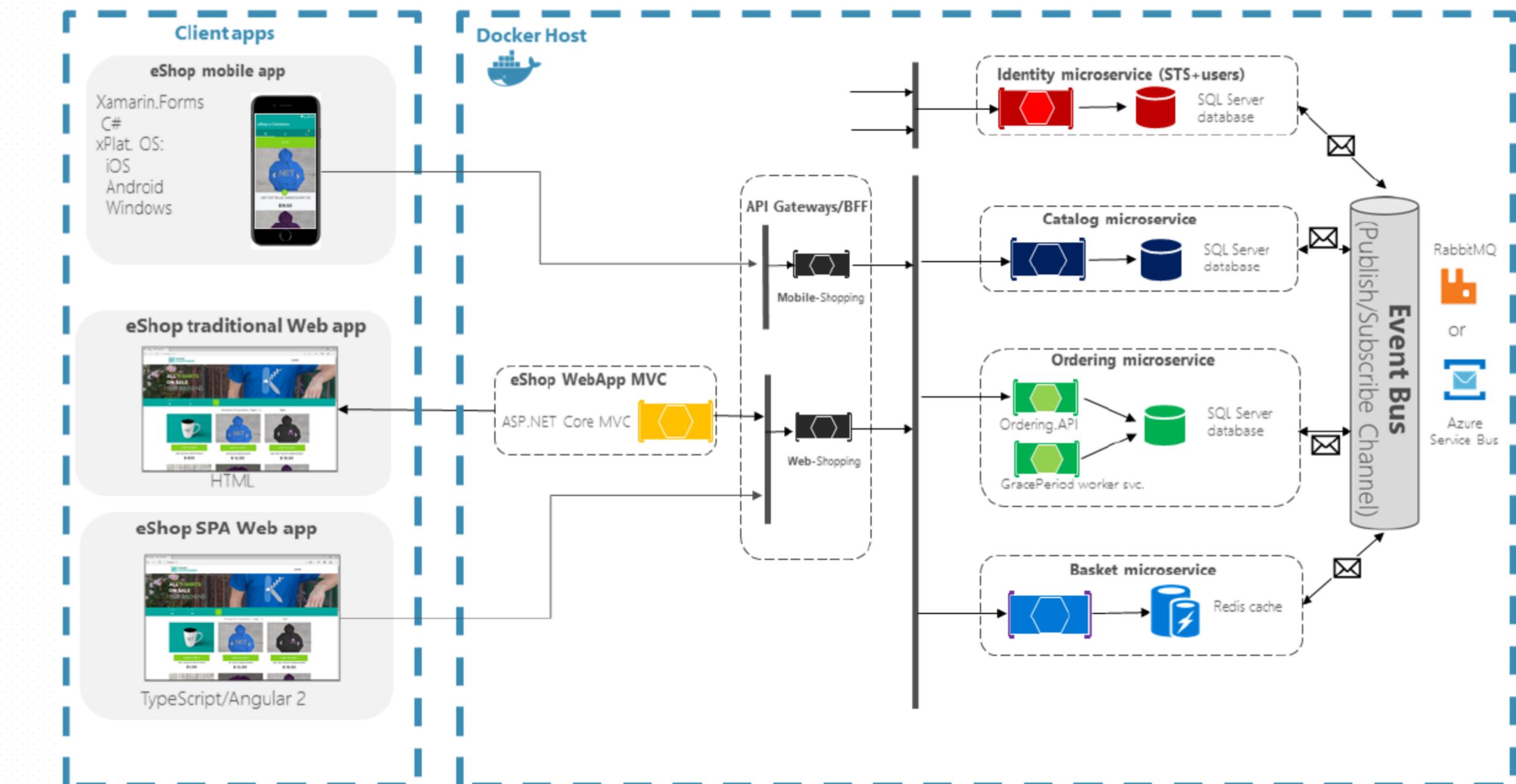
Using multiple API Gateways / BFF



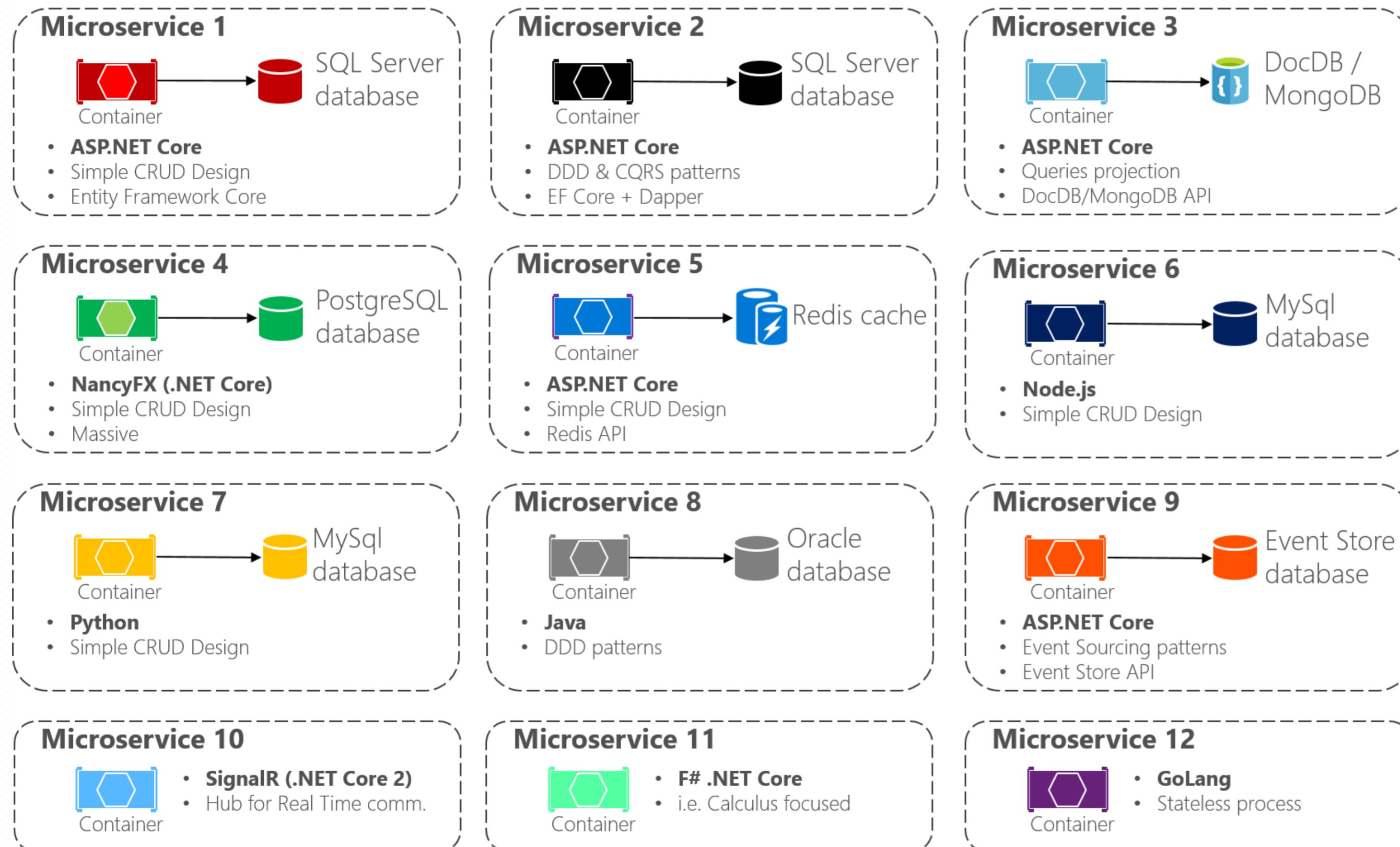


API Gateway, gelen istekleri yönlendirme, güvenlik, oturum yönetimi, hız ve önbellekleme gibi bir dizi işlevi sağlayarak API trafiğini yönetme görevini üstlenir.

Microservices



Polyglot



En iyi teknoloji kullanımı

Bağımsız geliştirme

Entegrasyon yeteneği

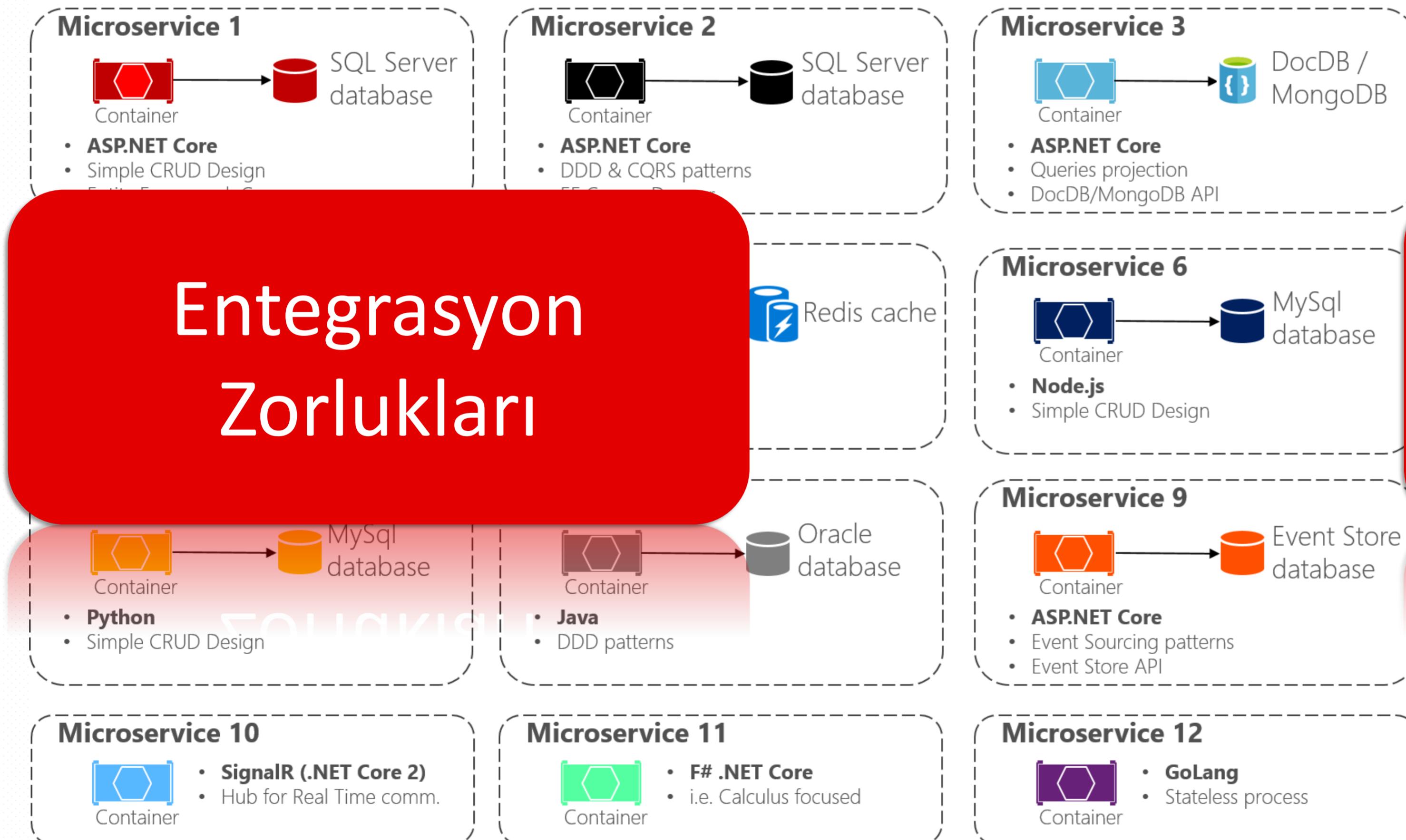
Ölçeklenebilirlik

Özelleştirme ve Uzmanlık

Bağımsız Geliştirme

Teknoloji Bağımsızlığı

Polygot



CAP Teoremi

Consistency



CP Kategorisi

Bazı veriler eşilebilir olmayıabilir.

Hbase
Google Big Table
Mongo DB
Redis
Hyper Table

Tüm istemciler veriyi aynı şekilde görür.

CA Kategorisi

Ağ sorunları sistemin durmasına sebep olabilir.

RDBMS
Oracle
MySQL
Aster Data

Pick Two



Partition
Tolerance

Ağ bölünmesine rağmen sistem işlevseldir.



Availability

AP Kategorisi

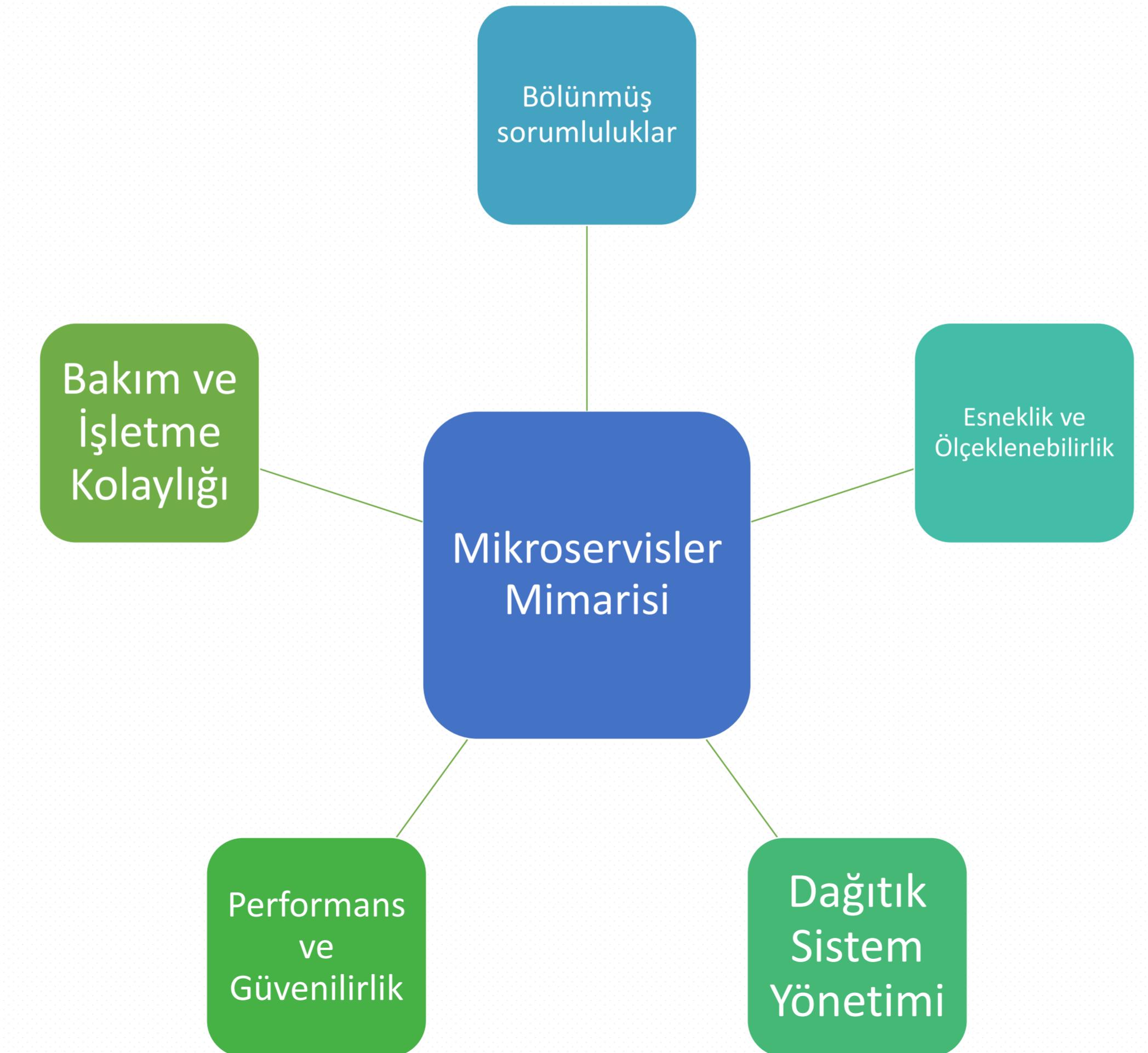
İstemciler tutarsız veri okuyabilir.

Dynamo DB
Cassandra
Voldemort
Riak
Couch DB

Mikroservisler Mimarısında Tasarım Desenleri

Microservices mimarisiyle çalışırken karşılaşılan belirli zorluklara çözüm sağlamak için çeşitliörüntüler kullanılır.

Mikroservisler Mimarısında Tasarım Desenleri



Mikroservisler Mimarısında Tasarım Desenleri

Decomposition Patterns

Decompose by Business Capability

Decompose by Subdomain

Strangler Pattern

Integration Patterns

API Gateway Pattern

Aggregator Pattern

Client-Side UI Composition Pattern

Database Patterns

Database per Service

Shared Database

CQRS Pattern

Saga Pattern

Mikroservisler Mimarısında Tasarım Desenleri

Communication Patterns

Request/Response Pattern

Messaging Pattern

Event Driven Pattern

Cross-Cutting Concern Patterns

Externalized Configuration

Service Discover Pattern

Circuit Breaker Pattern

Observability Patterns

Log Aggregation

Performance Metrics

Distributed Tracing

Health Check

Deployment Patterns

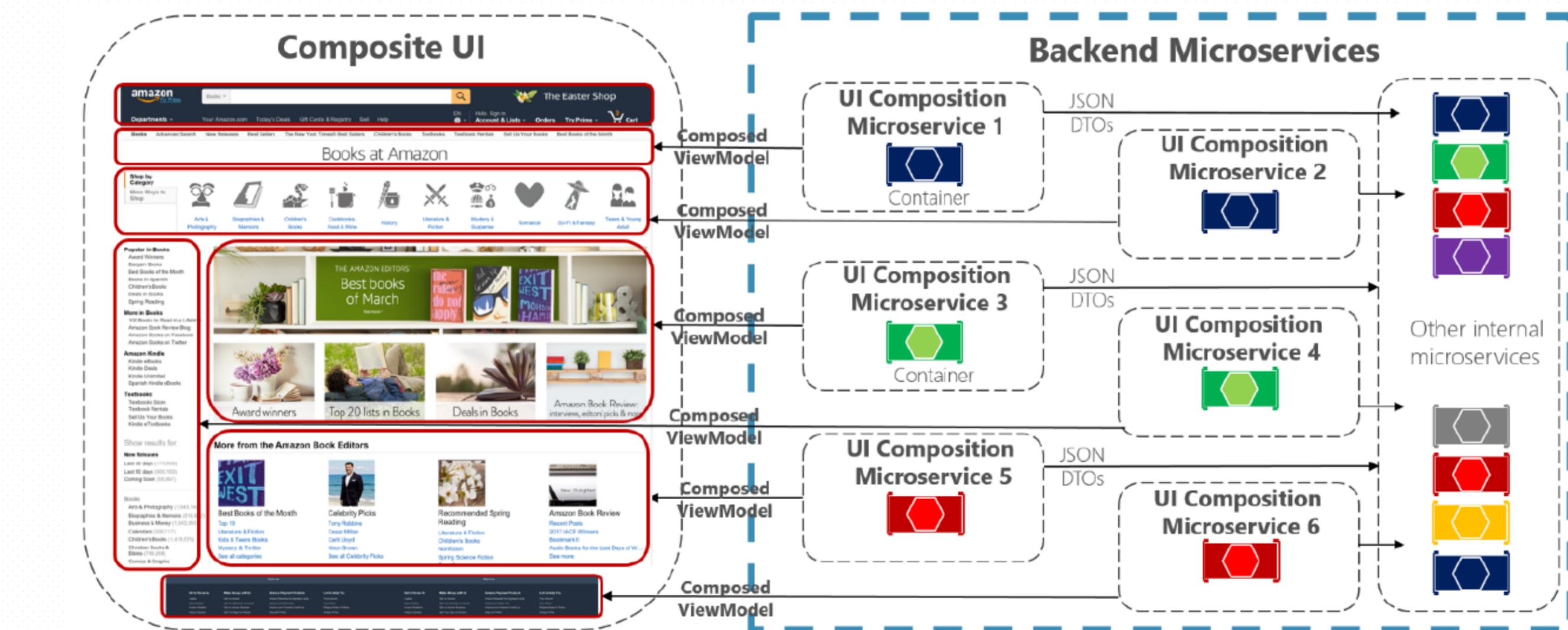
Multiple Service Instances per Host

Service Instance per Host

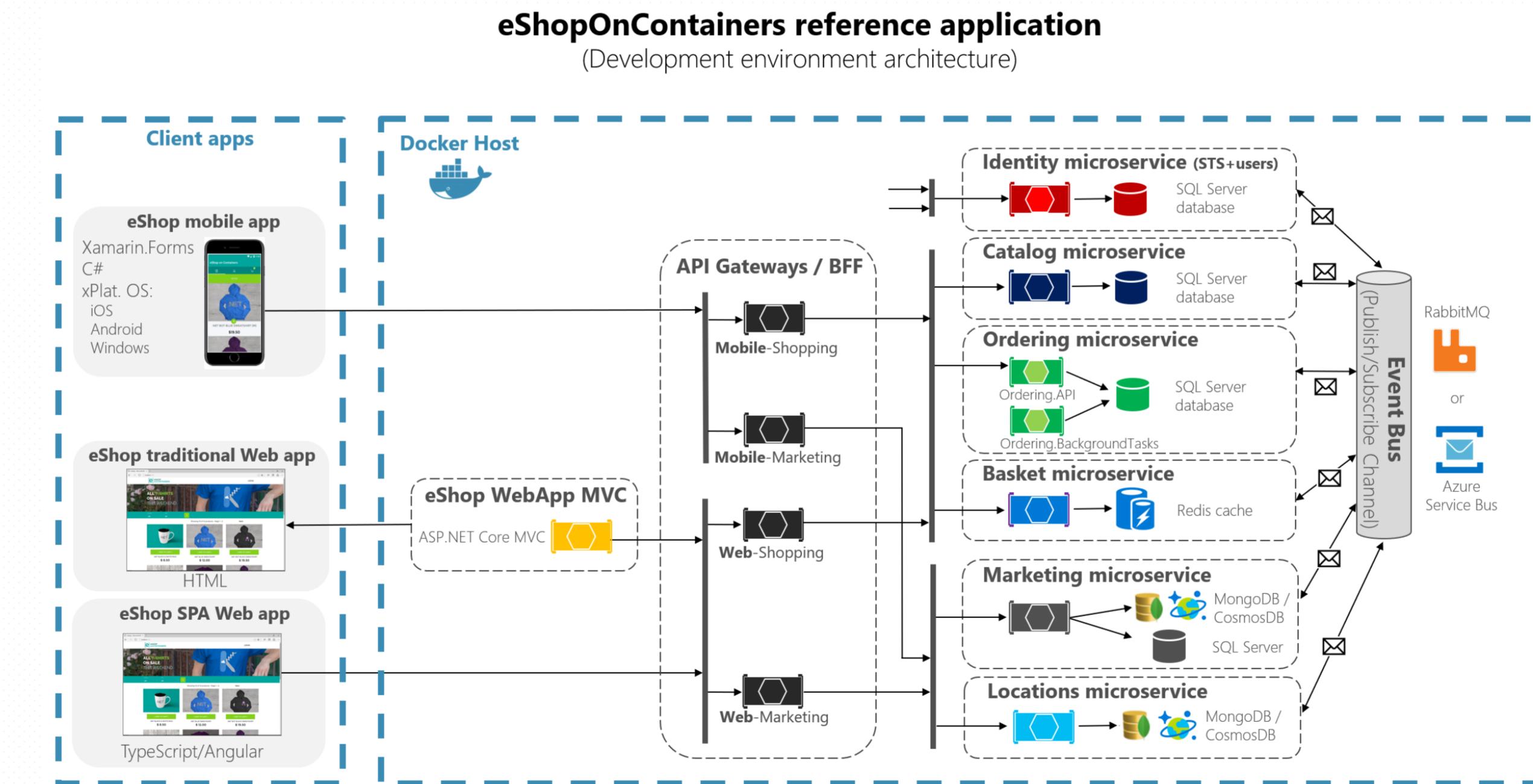
Serverless Deployment

Service Deployment Platform

Mikro hizmetler tarafından oluşturulan bileşik kullanıcı arayüzü



Microservices

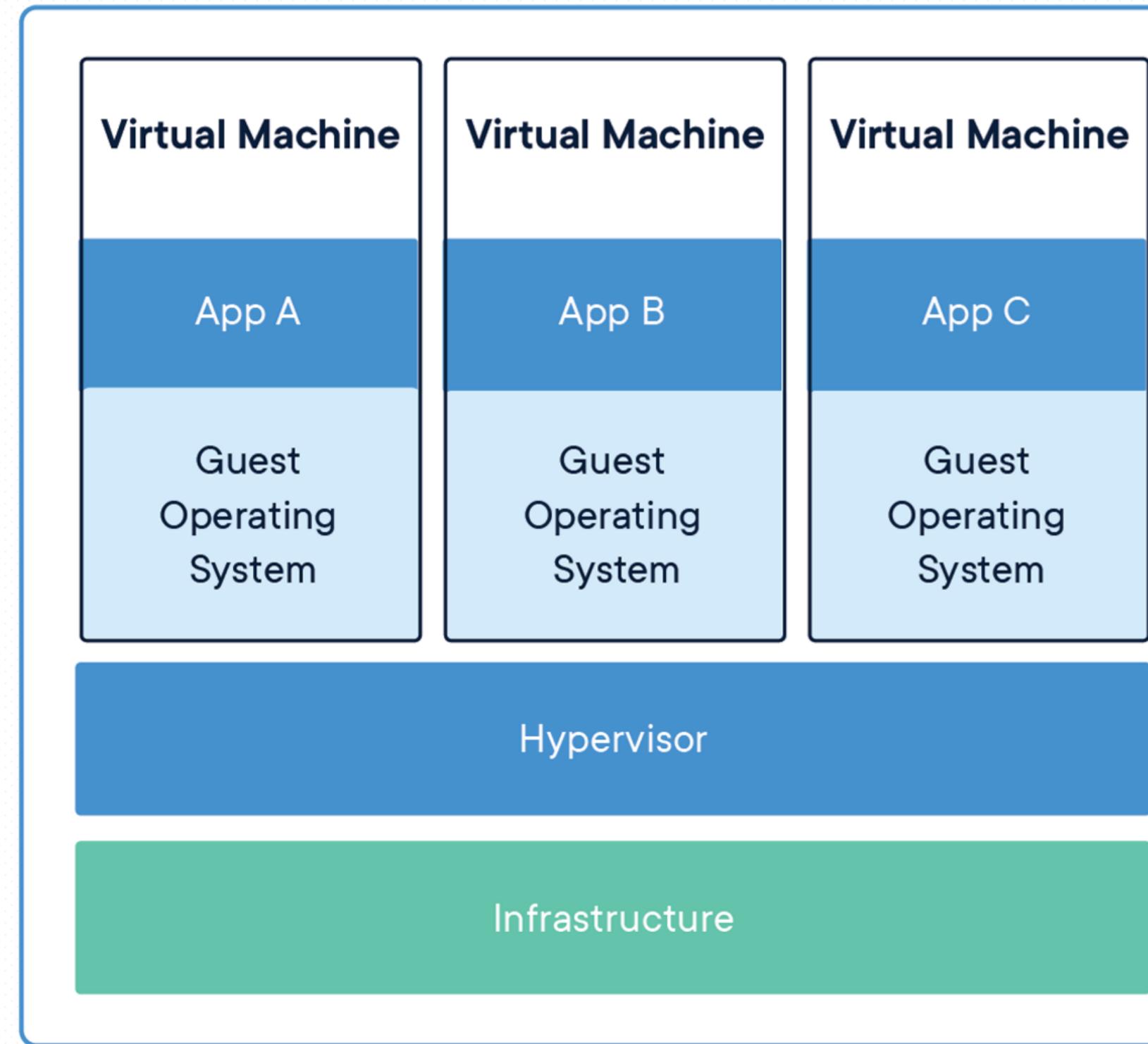


Yazılım Sadece Kod Değildir.

Yazılımların bağımlılıkları vardır. Bu bağımlılıklar paketler, referanslar ve donanım kaynakları ekseninde dikkate alınmalıdır.

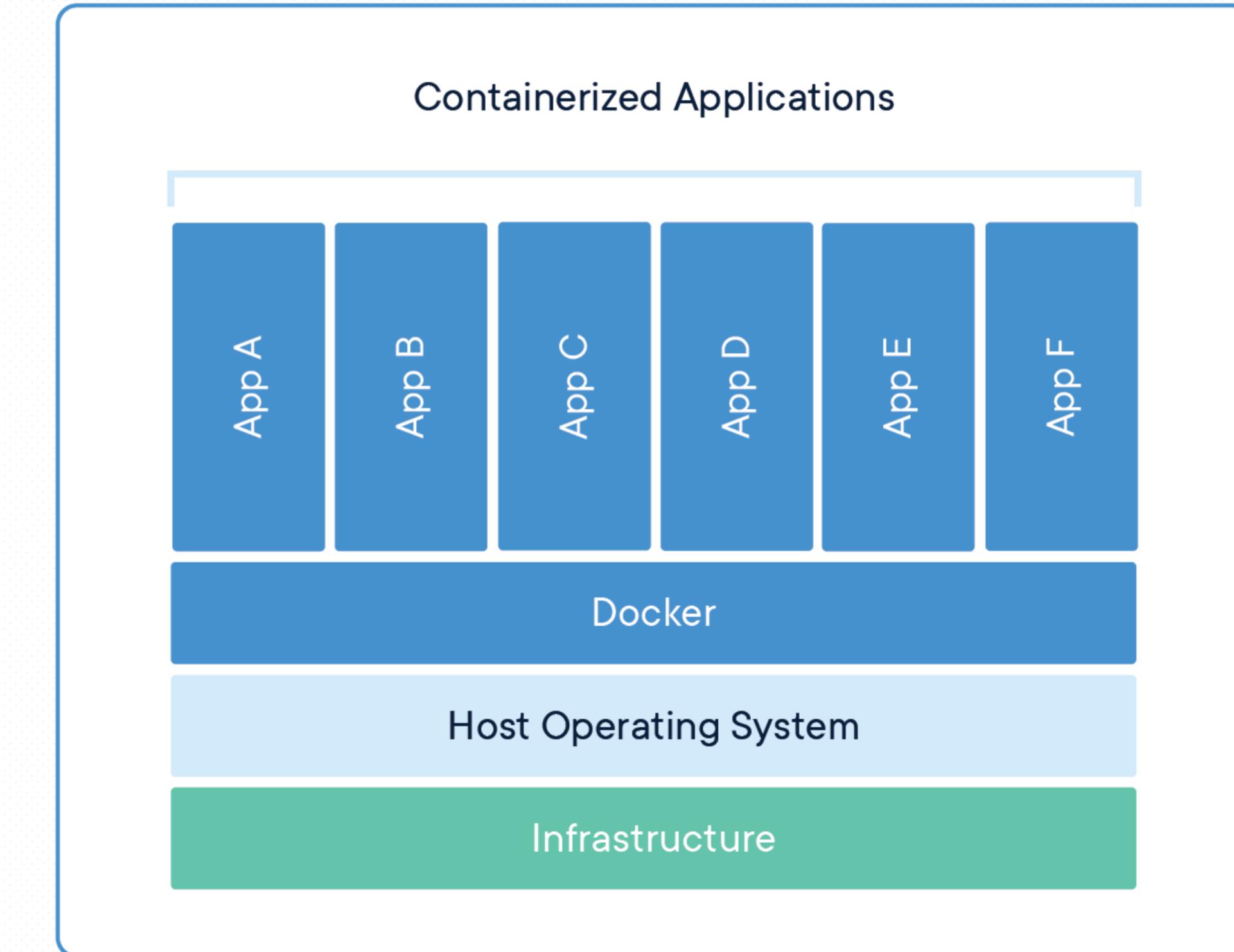
Sanal Makineler ve Konteynerler

Ölçeklenebilirlik



Geleneksel sanal makineler

Taşınabilirlik



Konteynerler

Yoğunluk & İzolasyon Seviyeleri



Donanım	Paylaşılmaz	Paylaşılır	Paylaşılır	Paylaşılır
Çekirdek	Paylaşılmaz	Paylaşılmaz	Paylaşılır*	Paylaşılır
Sistem Kaynakları (Ör: Dosya sistemi)	Paylaşılmaz	Paylaşılmaz	Paylaşılmaz	Paylaşılır

* Windows Hyper-V kapsayıcıları çekirdeği paylaşmaz.

Neler Öğrendik?

- Monolitik Mimari
- Çok Katmanlı (n-Tier) Mimari
- Altıgen (Hexagonal) Mimari
- Servis Odaklı Mimari (SOA)
- Olay Tabanlı Mimari (DDD)
- Mikroservis Mimarisi