

第01期-项目总体概况

2019年12月26日 13:31

在微服务架构非常流行的今天，微软也提供了一个基于 .NET Core 平台的微服务参考示例。

eShopOnContainers 基于 .NET Core 平台和 Docker 容器，提供一个简化版的在线商城。

<https://github.com/dotnet-architecture/News>

<https://github.com/dotnet-architecture/eShopOnContainers>

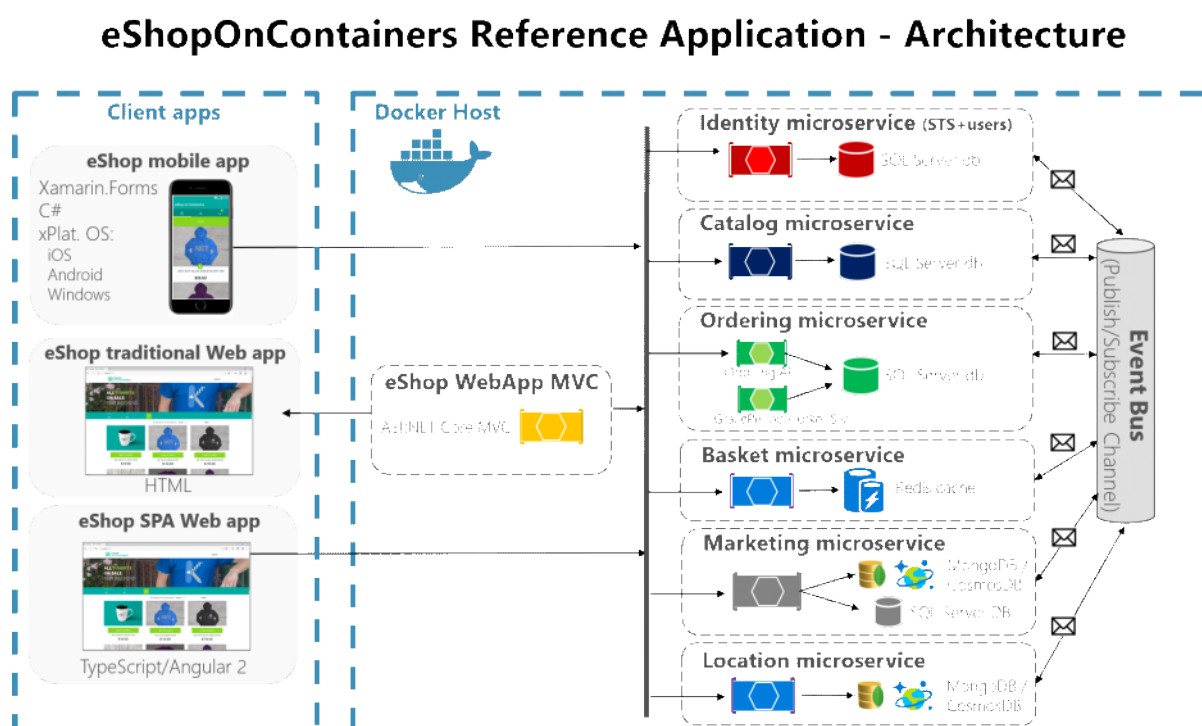
应用技术点

Docker 容器、DDD 领域驱动设计、微服务、设计模式、架构风格、分布式队列，事件队列、身份认证服务、容错故障处理、API 网关、分布式缓存、CQRS 命令查询职责分离、CAP原则(一致性、可用性和分区容错性)、K8S、CI 持续集成 和 CD 持续部署方案等。

eShopOnContainers 作为跨平台的微服务架构，得益于 .NET Core 能够在 Linux 和 Windows 容器中运行。

架构图

客户端应用和Docker主机中运行的服务端应用。



客户端应用

其包含基于浏览器的Web应用、基于Xamarin的Android、IOS、Windows/UWP 移动应用。

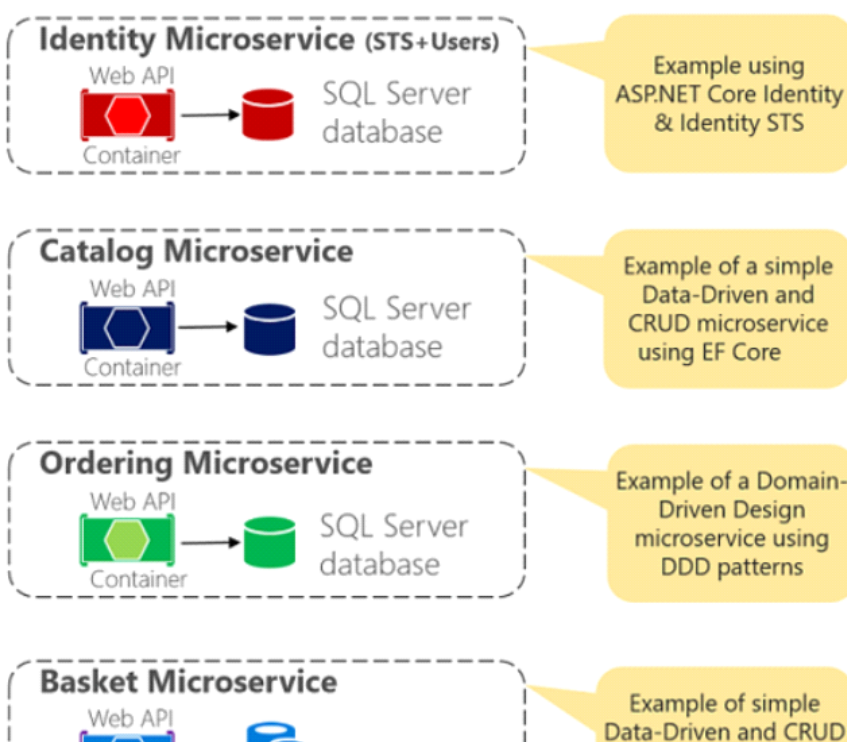
服务端应用

多个自治微服务，每个微服务都有自己的数据库，每个微服务都有不同的实现方式，简单的 CRUD、DDD 和 CQRS 模式，使用 HTTP 作为微服务之间的通信协议，支持异步通信，使用 Integration Events 集成事件 和 Event Bus 事件总线进行数据的更新传播，客户端与微服务通过 API 网关通信。

- Identity Microservice（身份微服务）：用于身份认证和授权。使用SQL Server数据库。
- Catalog microservice（产品目录微服务）：用于产品资料的维护。使用SQL Server数据库。
- Ordering microservice（订单微服务）：用于订单逻辑的处理。使用SQL Server数据库。
- Basket microservice（购物车微服务）：用于购物车逻辑的处理。使用Redis数据库。
- Marketing microservice（市场营销微服务）：用于市场营销逻辑的处理。使用 MongoDB/CosmosDB 和SQL Server数据库。
- Locations microservice（位置微服务）：用于提供位置服务。使用MongoDB 数据库。
- [New] Payment microservice（支付微服务）：用于处理支付逻辑。

微服务架构模式

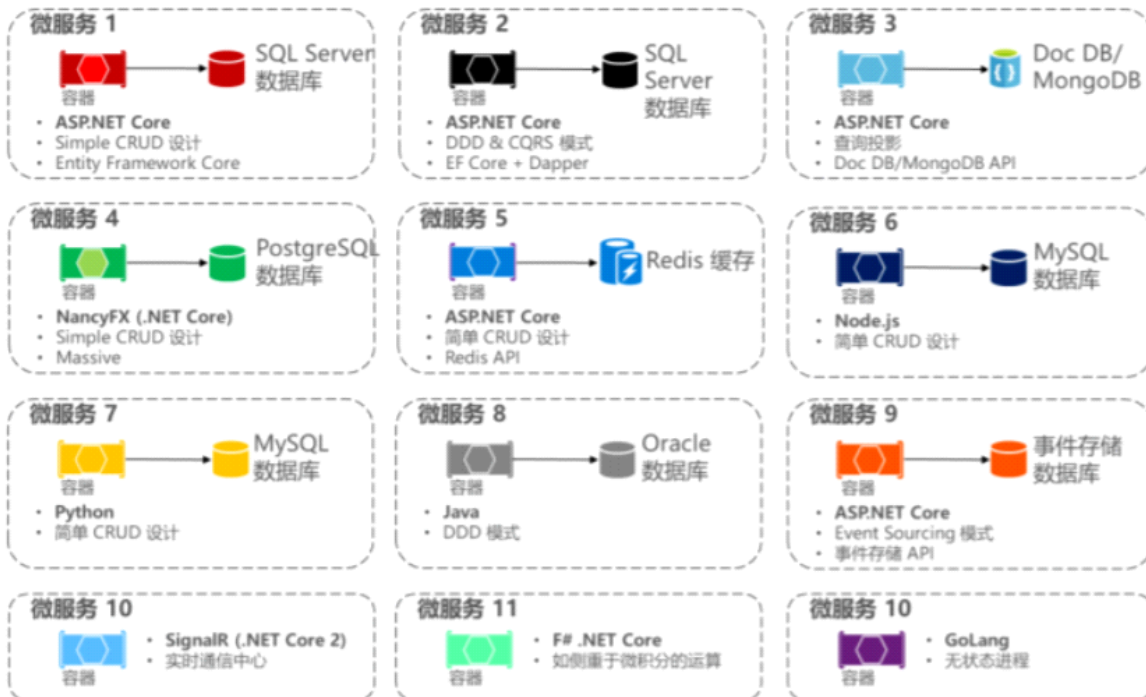
Different types of microservices





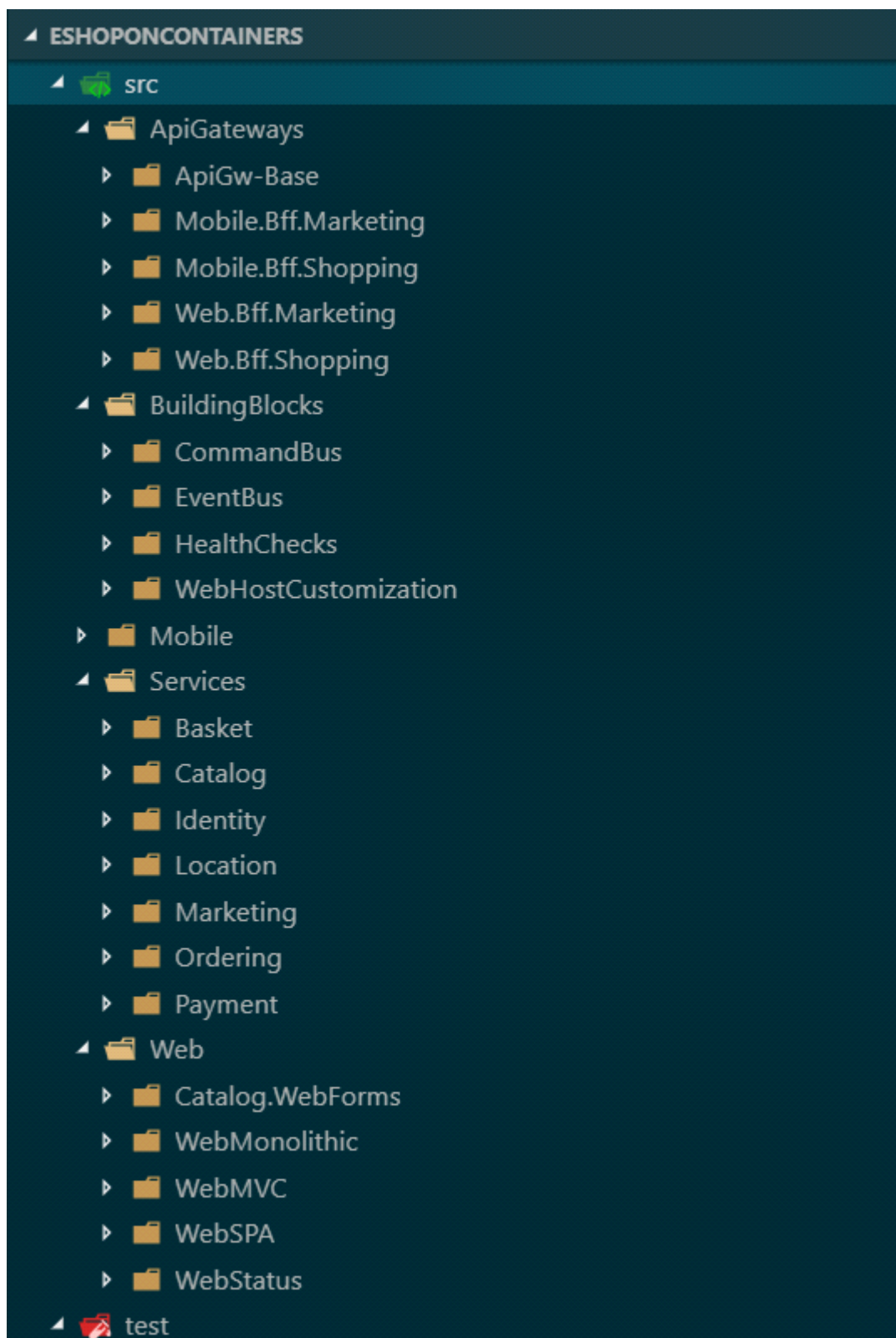
微服务可使用不同的架构模式，单层或者多层。

多种架构模式和多语言的微服务世界

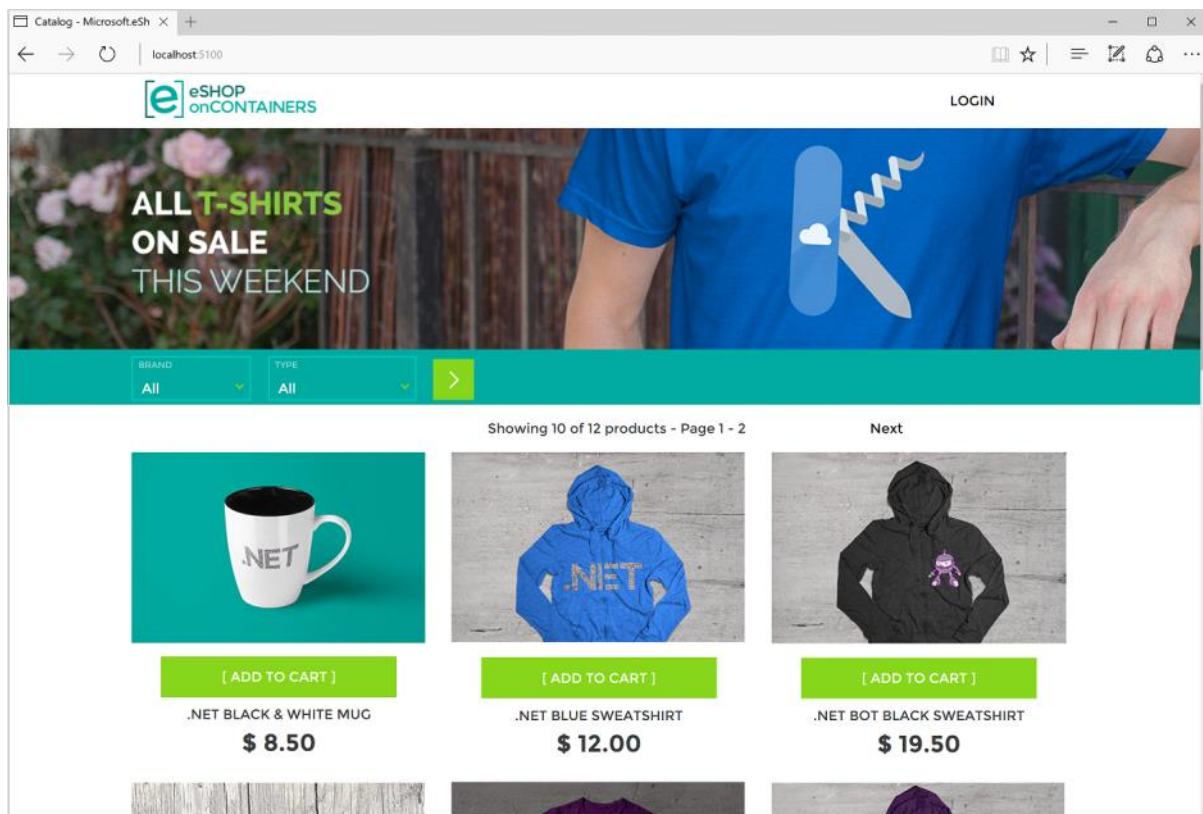


在由多个微服务组成的应用程序中，可以用不同方式实现每个微服务。每个微服务可能具有不同架构模式，并根据应用程序的性质、业务需求和优先级使用不同的语言和数据库。

这也就是微服务的灵活性与复杂性的源头。



运行起来就简单商城



官方参考书

本指南介绍如何使用容器开发基于微服务的应用程序并对其进行管理。 本指南探讨使用 .NET Core 和 Docker 容器的体系结构设计和实现方法。



<https://docs.microsoft.com/zh-cn/dotnet/architecture/microservices>