

ASP.NET Core & Docker & K8s

From 0 to Cloud

张善友

MVP / TVP
geffzhang@weyhd.com



Agenda

- Principles of Microservices
- .NET Core
- What's Docker and why should I care
- ASP.NET Core & Docker
- Let's port a non-trivial project to Docker and AKS

Definition

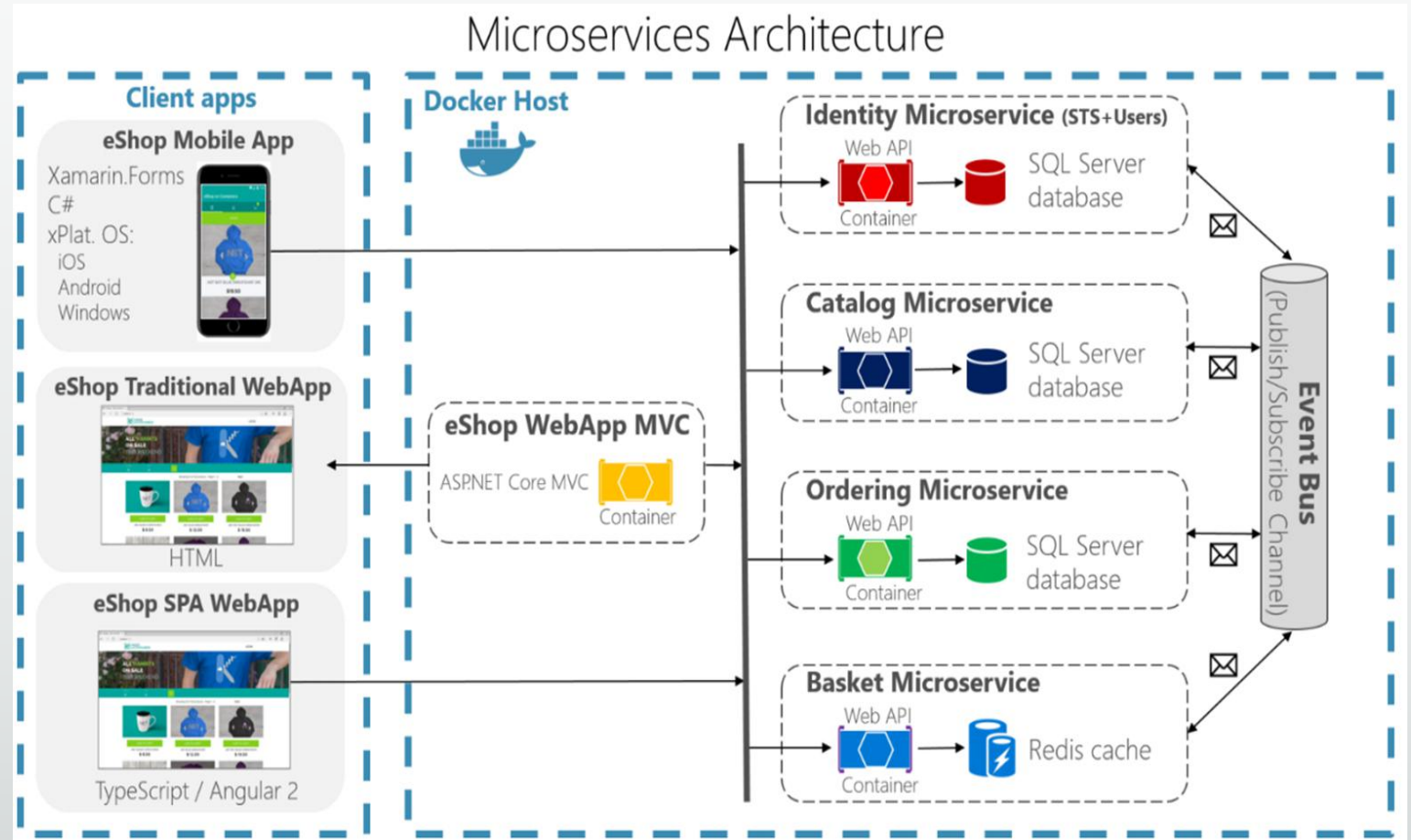
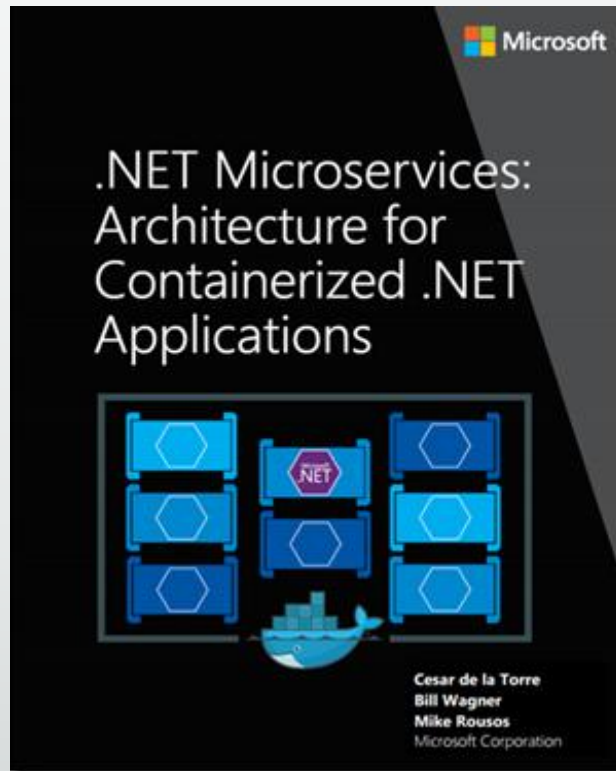
A microservices architecture consists of a collection of small, autonomous services. Each service is self-contained and should implement a single business capability.

Characteristics of a Microservice

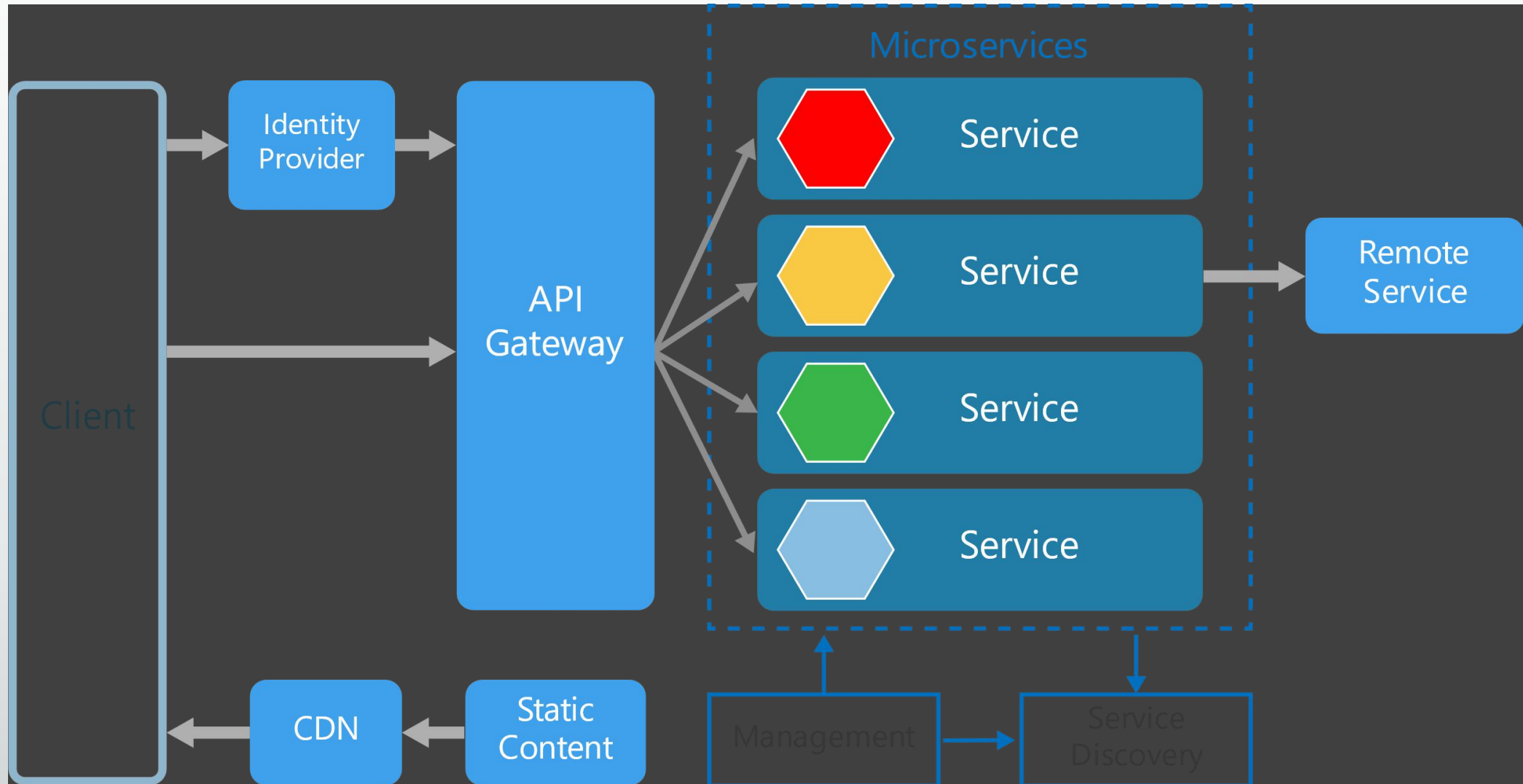
- Services are small, independent, and loosely coupled.
- Each service is a separate codebase.
- Services can be deployed independently.
- Services are responsible for persisting their own data or external state.
- Services communicate with each other by using well-defined APIs.
- Services don't need to share the same technology stack, libraries, or frameworks.

.NET Microservices - Architecture for Containerized .NET Applications

<http://aka.ms/MicroservicesEbook>



Microservices Overview



Other Components in a Typical Microservices Architecture

- **Management**
- **Service Discovery**
- **API Gateway**

.NET Core



Ben Adams @ben_a_adams · 3小时

ASP.NET Core 3.0 has had a +30% performance improvement in the last 2 months alone!

Compared to 2.2:

+42% in throughput and -92% reduction in memory usage!

Always improving #aspnetcore #dotnet #dotnetcore

翻译推文



16



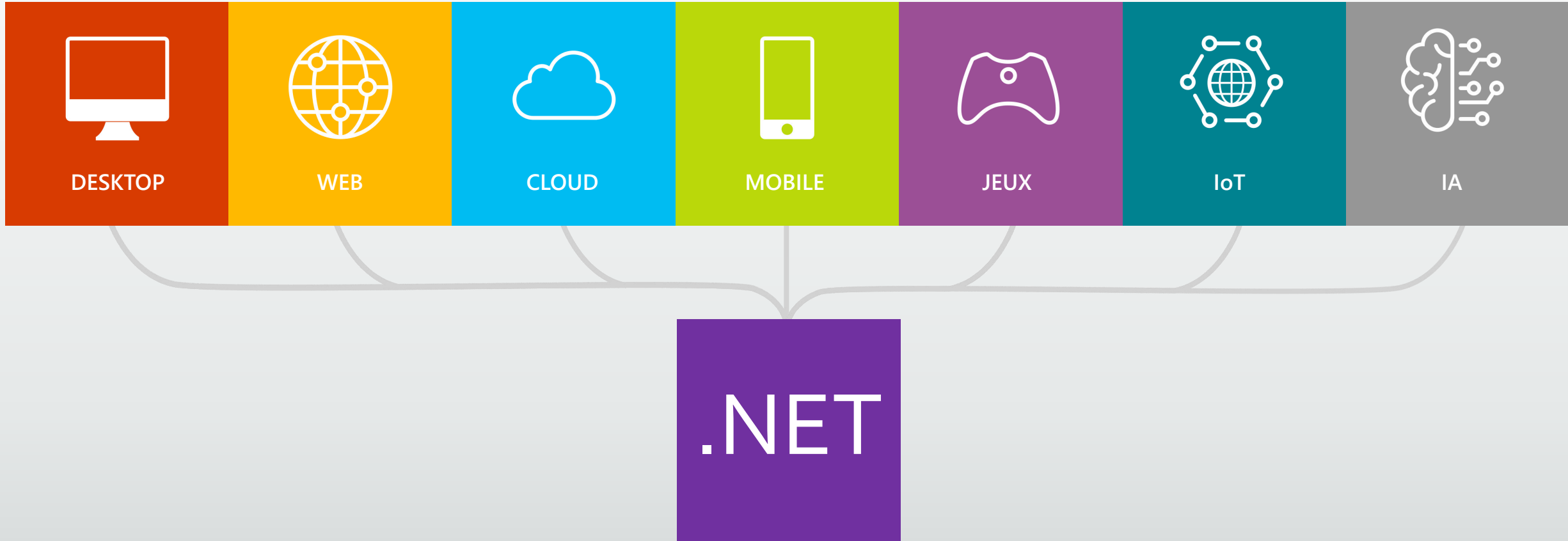
185



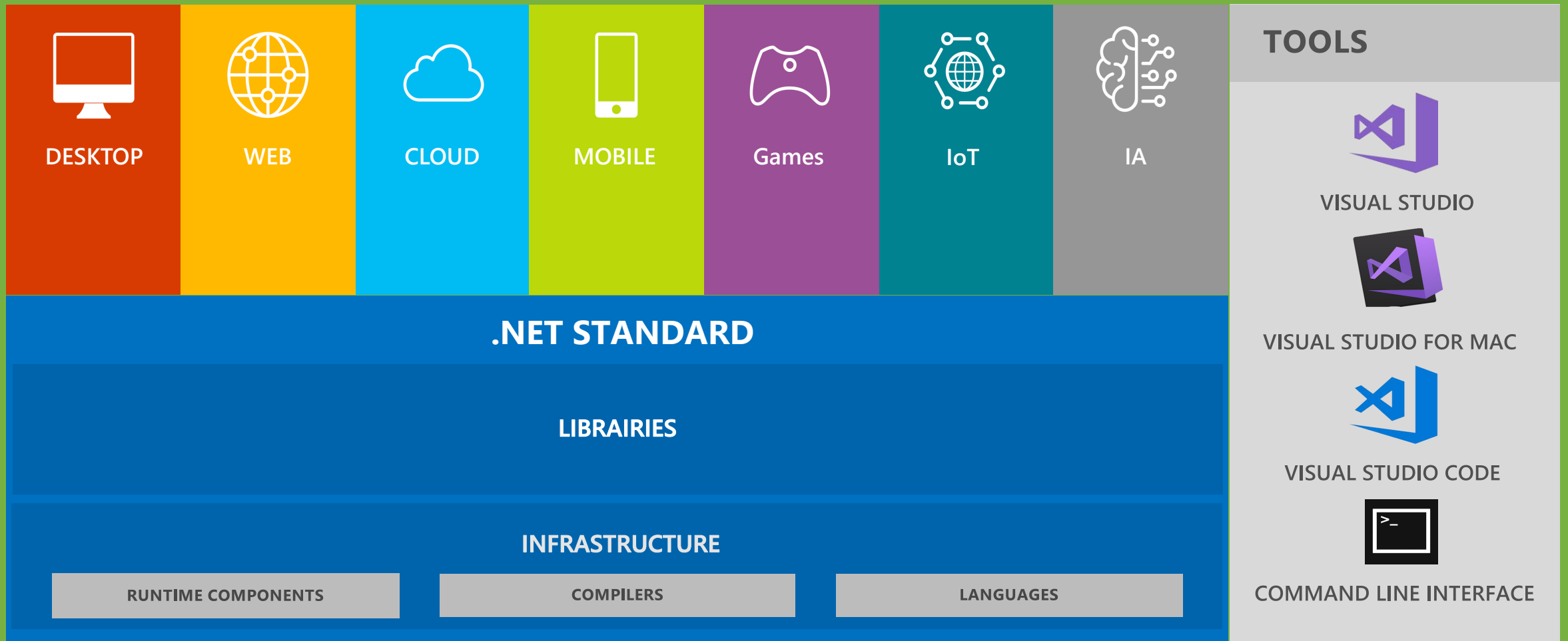
426



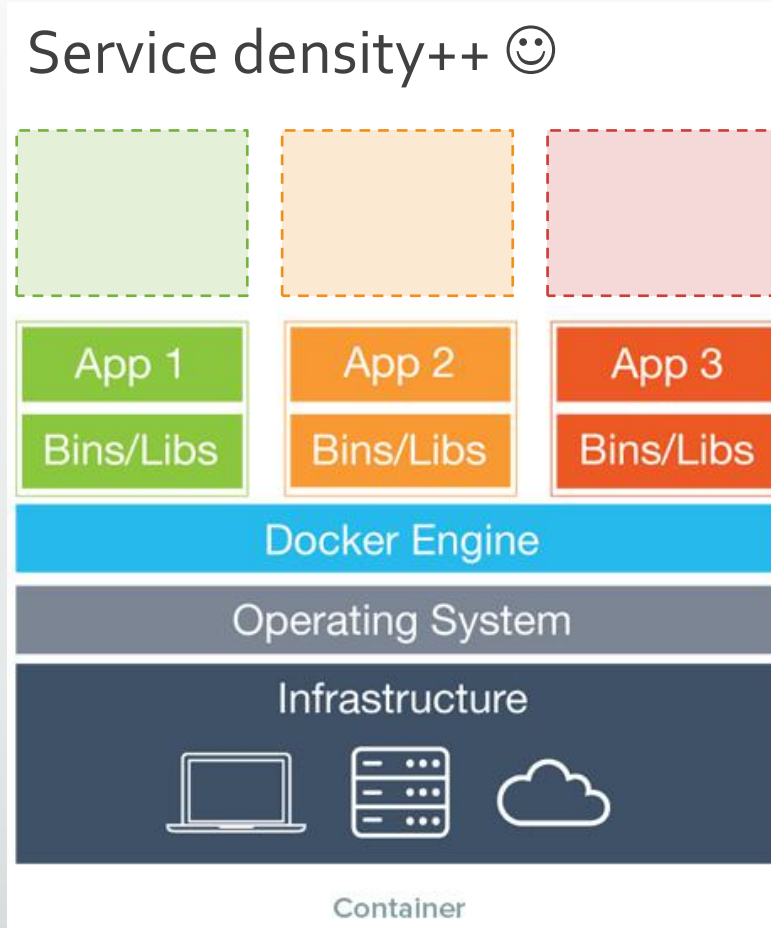
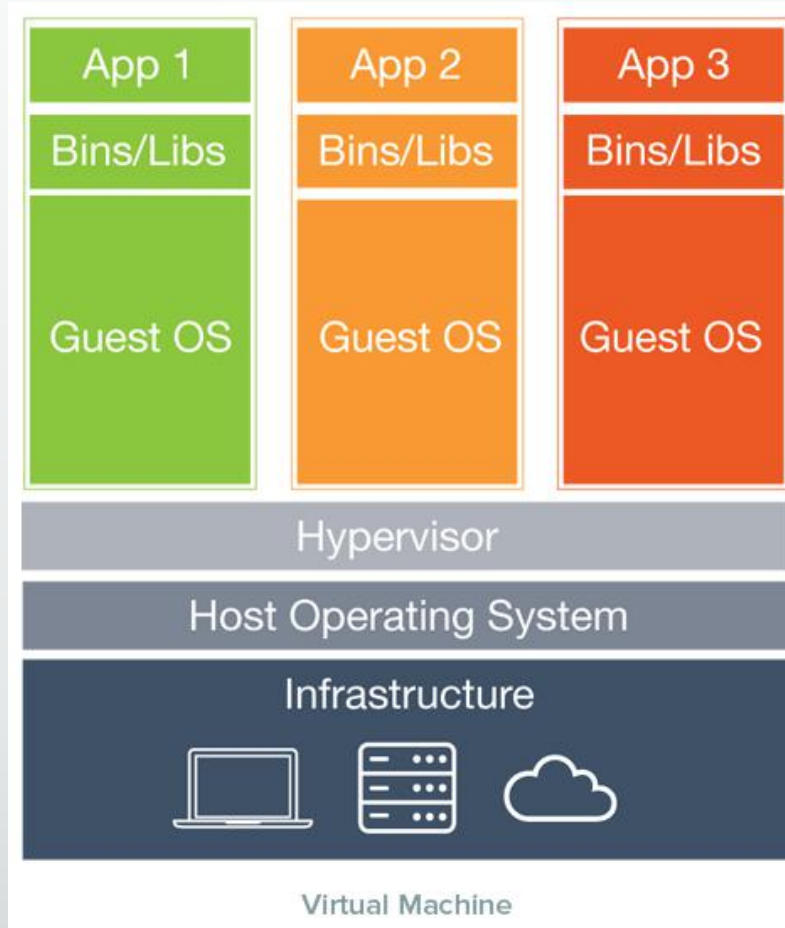
A platform and build everything



.NET - Software development platform



What's Docker

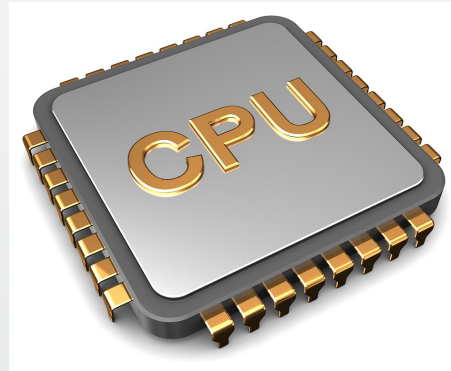


Containers == Virtualized Operating System

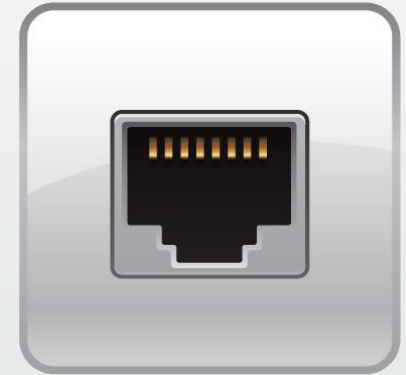
Kernel namespaces (Linux and Windows)



Virtual File System

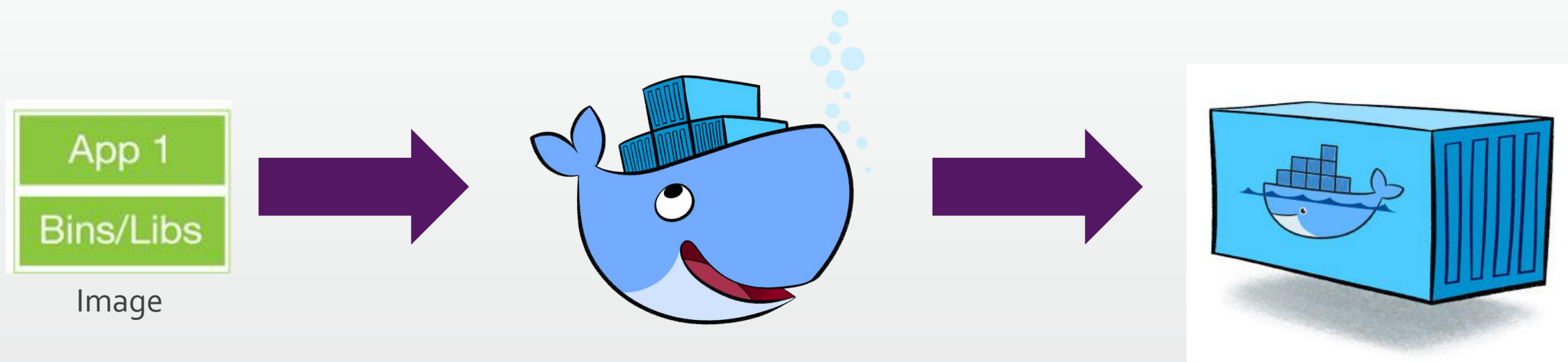


Virtual Process Tree



Virtual Network

Docker is a (the) container engine

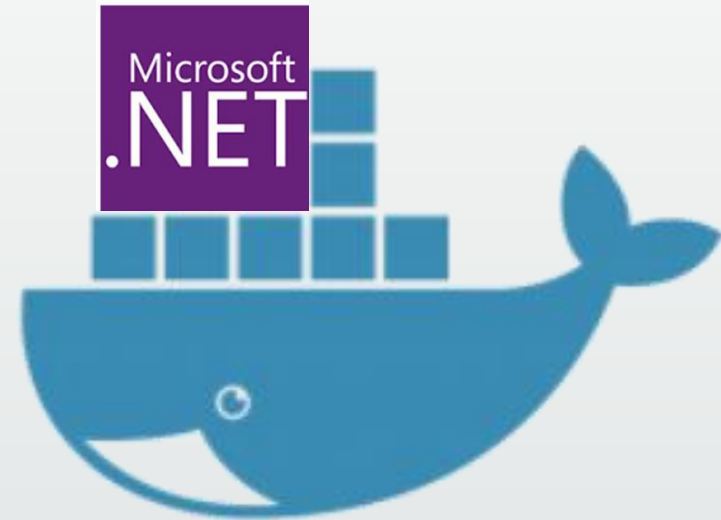
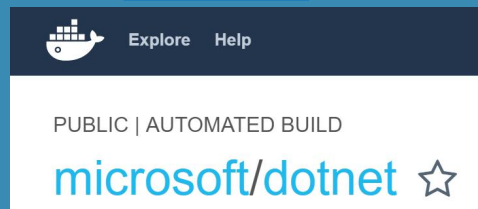


- Docker (aka Moby project) is free and open source, no limitations
- There's an enterprise edition (hosting, support, certification...)
- It's the de-facto standard for container-based applications

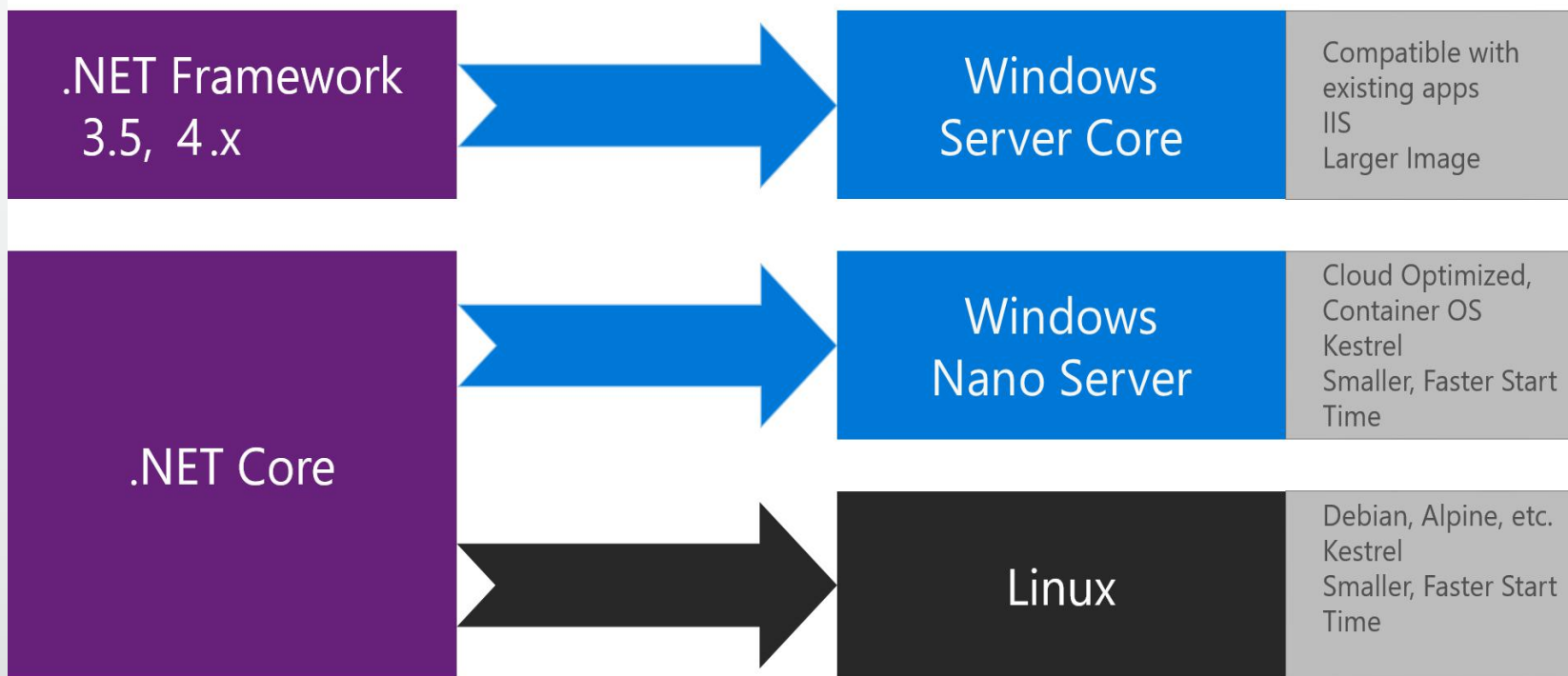
Docker and .NET

- **.NET Core** Docker images
xPlat. (Linux & Windows)
- **.NET Framework** images
Windows only

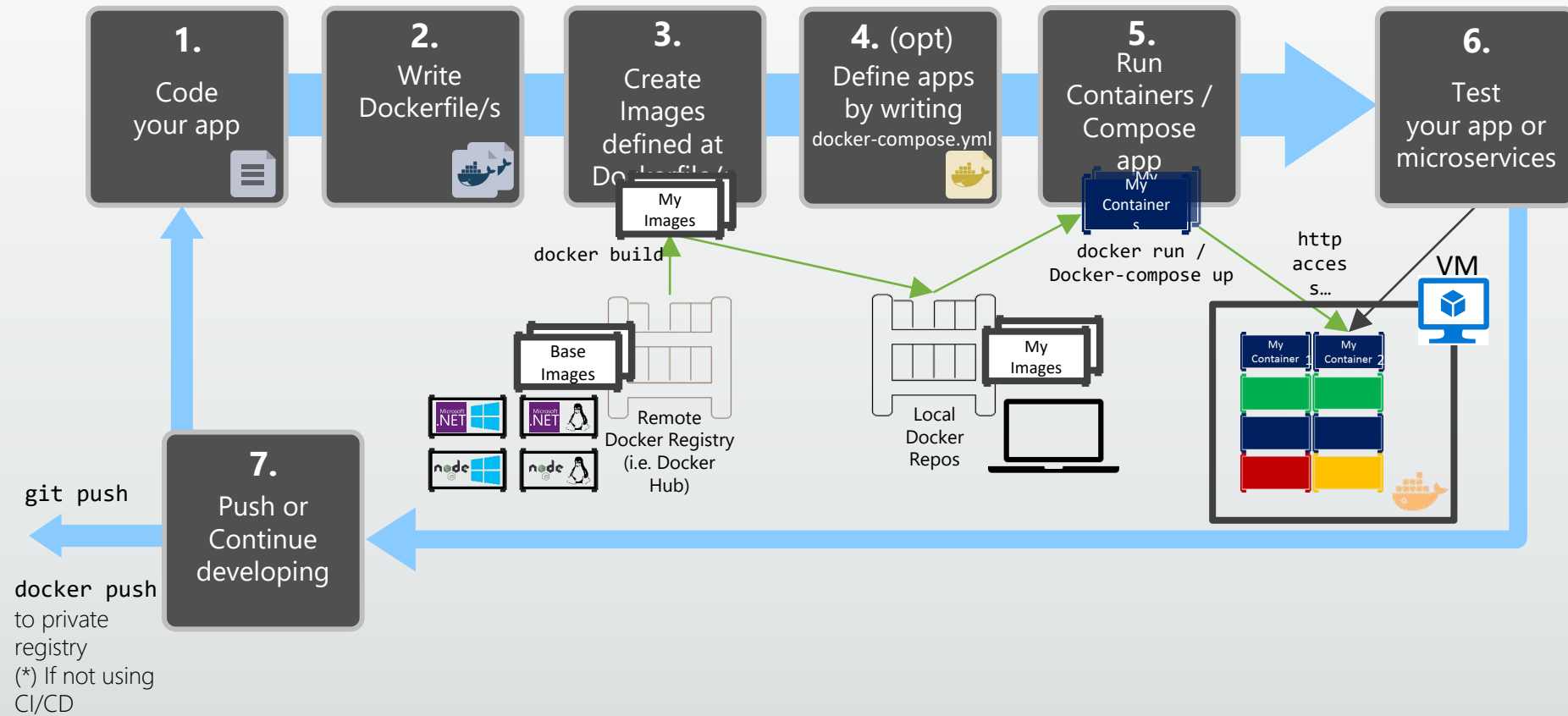
See at [Docker Hub](#)



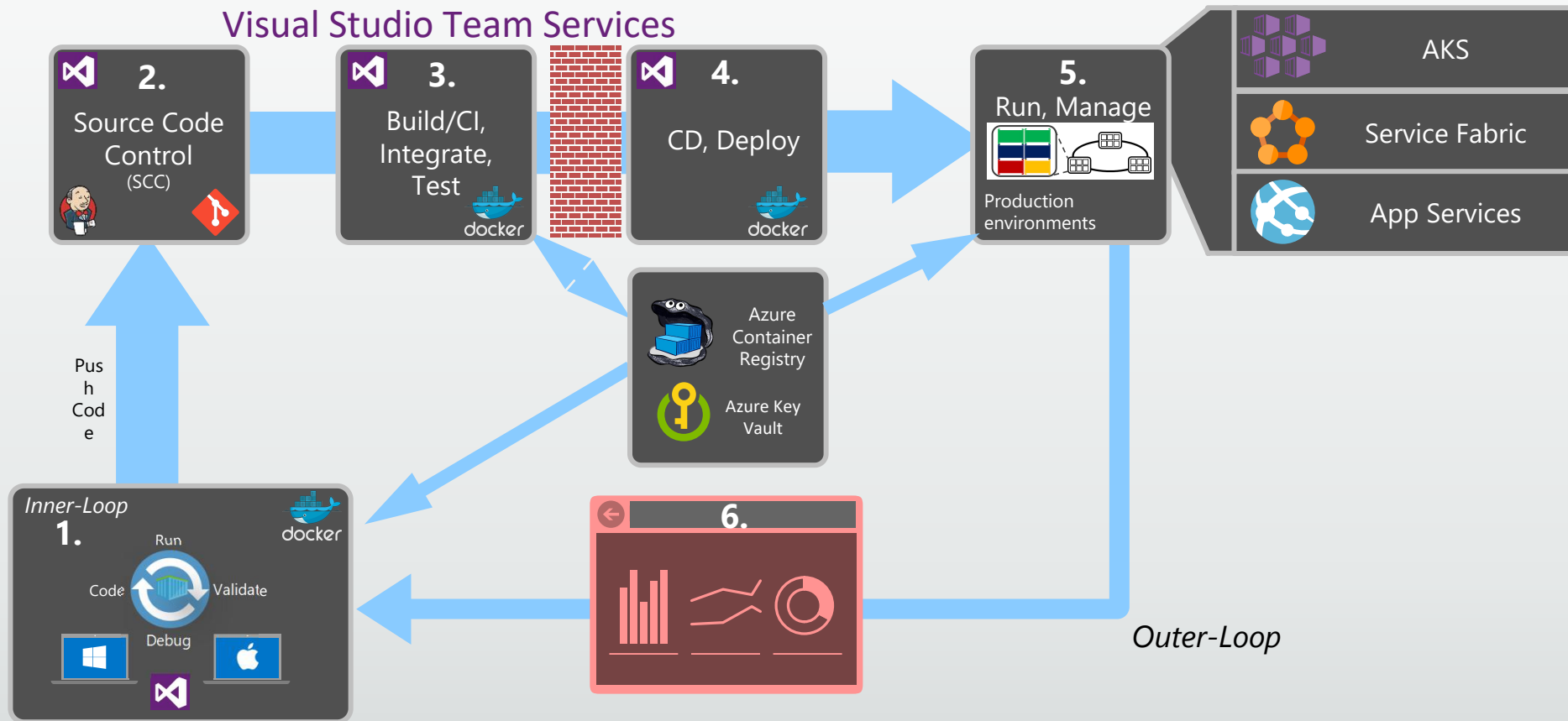
What OS to target with .NET containers



Inner-Loop development workflow for Docker apps



DevOps Workflow using containers



What we are going to build

TechTalksWeb Home About Contact

[Create New](#)

Id	Description	Category			
1	Scaling Docker Containers	Meetup	Edit	Details	Delete
2	Azure Container Services	Free Conference	Edit	Details	Delete
3	Kubernetes	Paid Conference	Edit	Details	Delete
4	Docker and kubernetes	Free Conference	Edit	Details	Delete
6	Modernize Docker and Azure	Paid Conference	Edit	Details	Delete

TechTalksWeb Home About Contact

[Back to List](#)

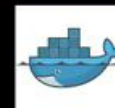
Create New Tech Talk

TechTalkName

CategoryId

LevelId

Create



Dockerfile explained

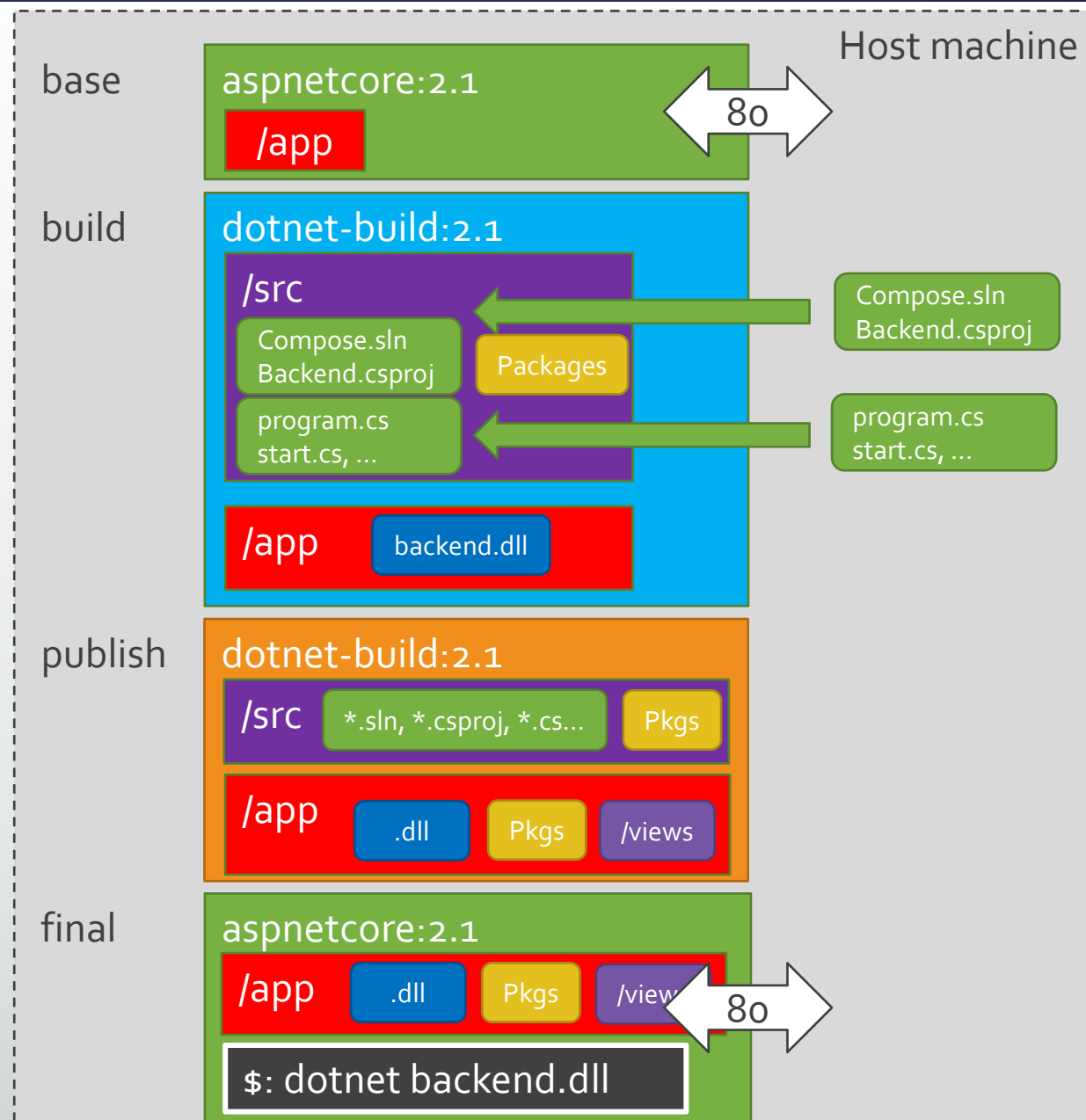
FROM microsoft/dotnet:2.1-aspnetcore-runtime AS base
WORKDIR /app
EXPOSE 80

FROM microsoft/dotnet:2.1-sdk AS build
WORKDIR /src
COPY *.sln ./
COPY backend/backend.csproj backend/
RUN dotnet restore

COPY . .
WORKDIR /src/backend
RUN dotnet build -c Release -o /app

FROM build AS publish
RUN dotnet publish -c Release -o /app

FROM base AS final
WORKDIR /app
COPY --from=publish /app .
ENTRYPOINT ["dotnet", "backend.dll"]



Dockerfile explained

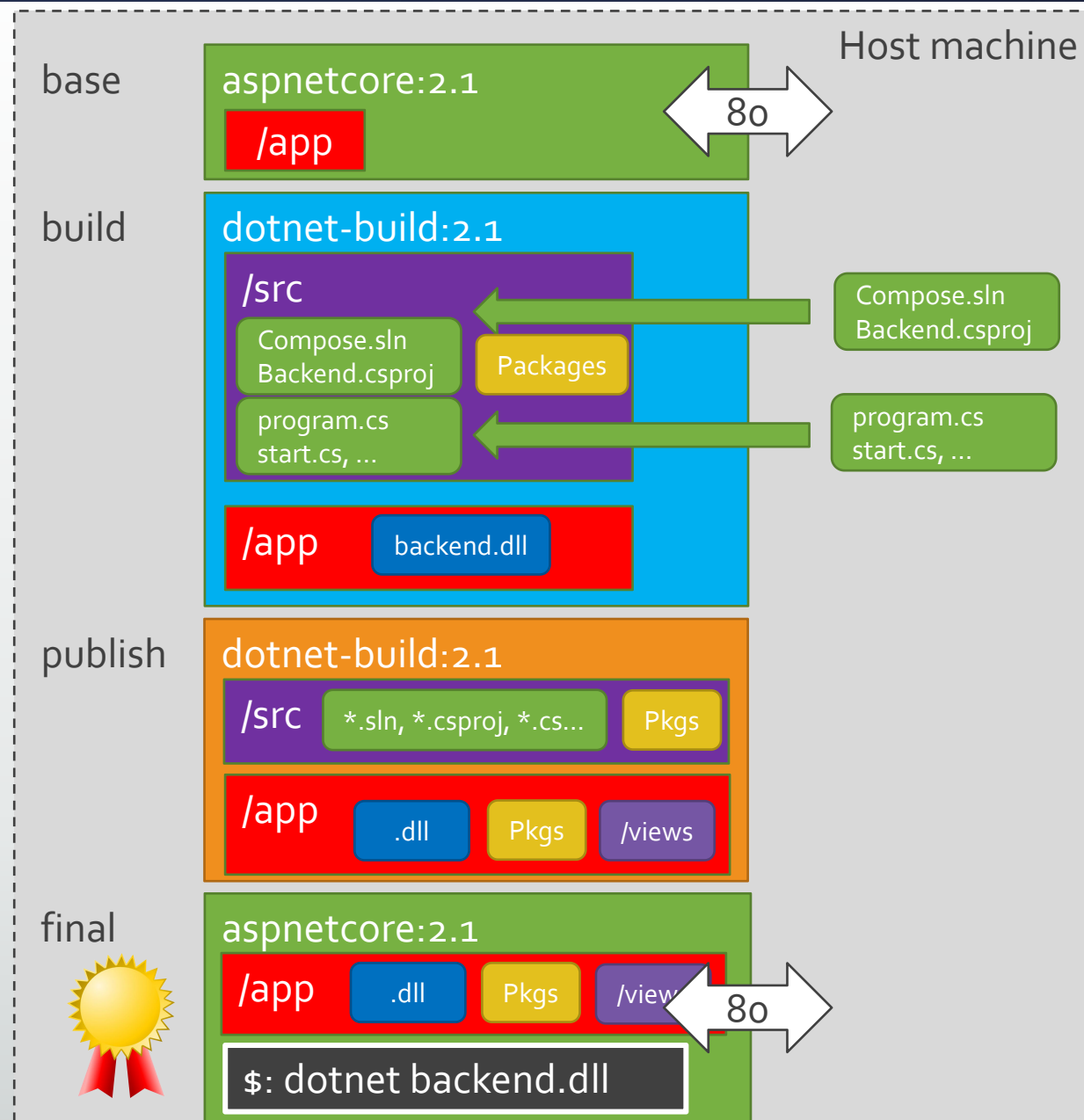
```
FROM microsoft/dotnet:2.1-aspnetcore-runtime AS base
WORKDIR /app
EXPOSE 80
```

```
FROM microsoft/dotnet:2.1-sdk AS build
WORKDIR /src
COPY *.sln ./
COPY backend/backend.csproj backend/
RUN dotnet restore
```

```
COPY . .
WORKDIR /src/backend
RUN dotnet build -c Release -o /app
```

```
FROM build AS publish
RUN dotnet publish -c Release -o /app
```

```
FROM base AS final
WORKDIR /app
COPY --from=publish /app .
ENTRYPOINT ["dotnet", "backend.dll"]
```





We need an orchestrator: enter Kubernetes and AKS

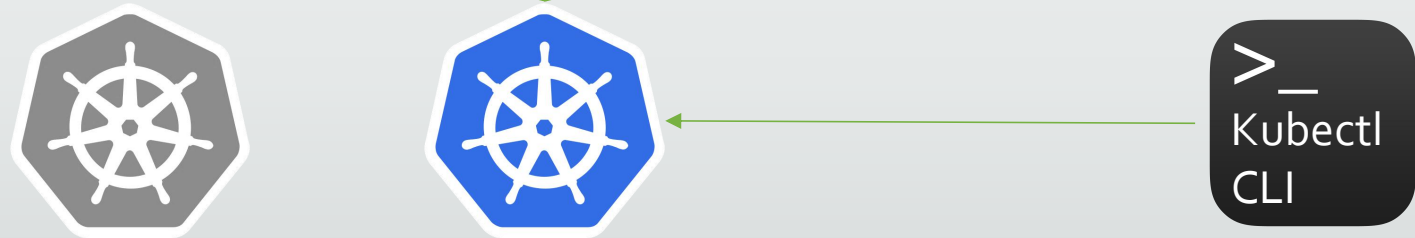
Running Pods
(~containers)



Agent
nodes



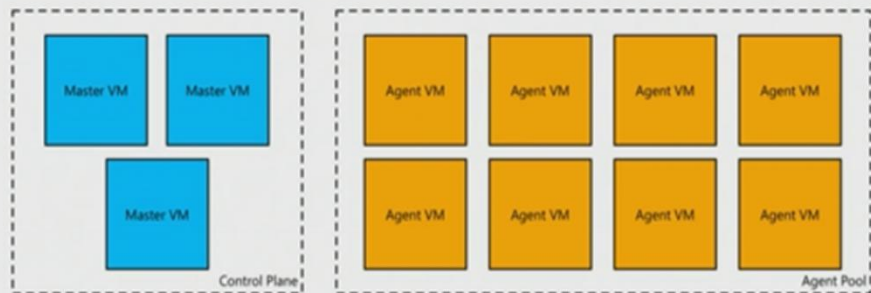
Master
nodes



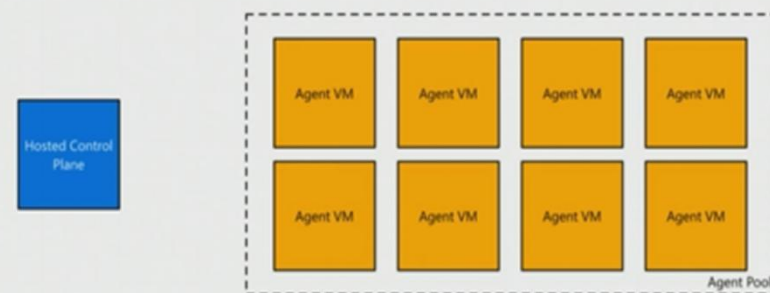
AKS: Managed Kubernetes

- Azure-hosted control plane
 - No master nodes to manage or pay for
- Automated upgrades and patching
 - Easily upgrade control plane and worker nodes to new versions of Kubernetes
- Scale agent pool to increase or decrease capacity

Kubernetes without AKS

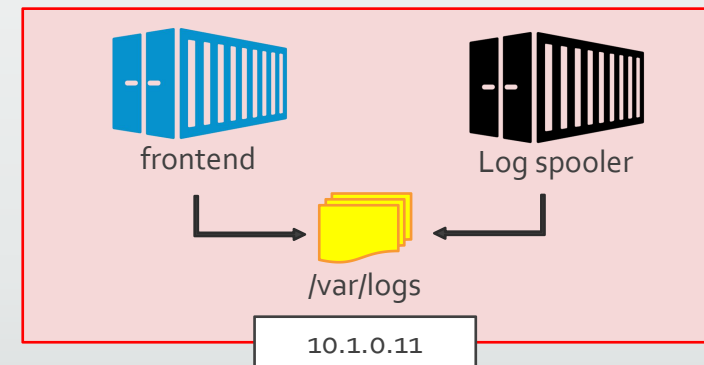
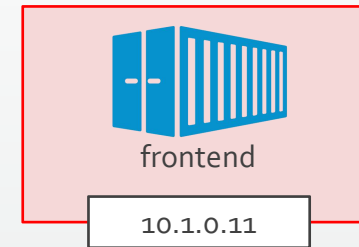


Kubernetes with AKS



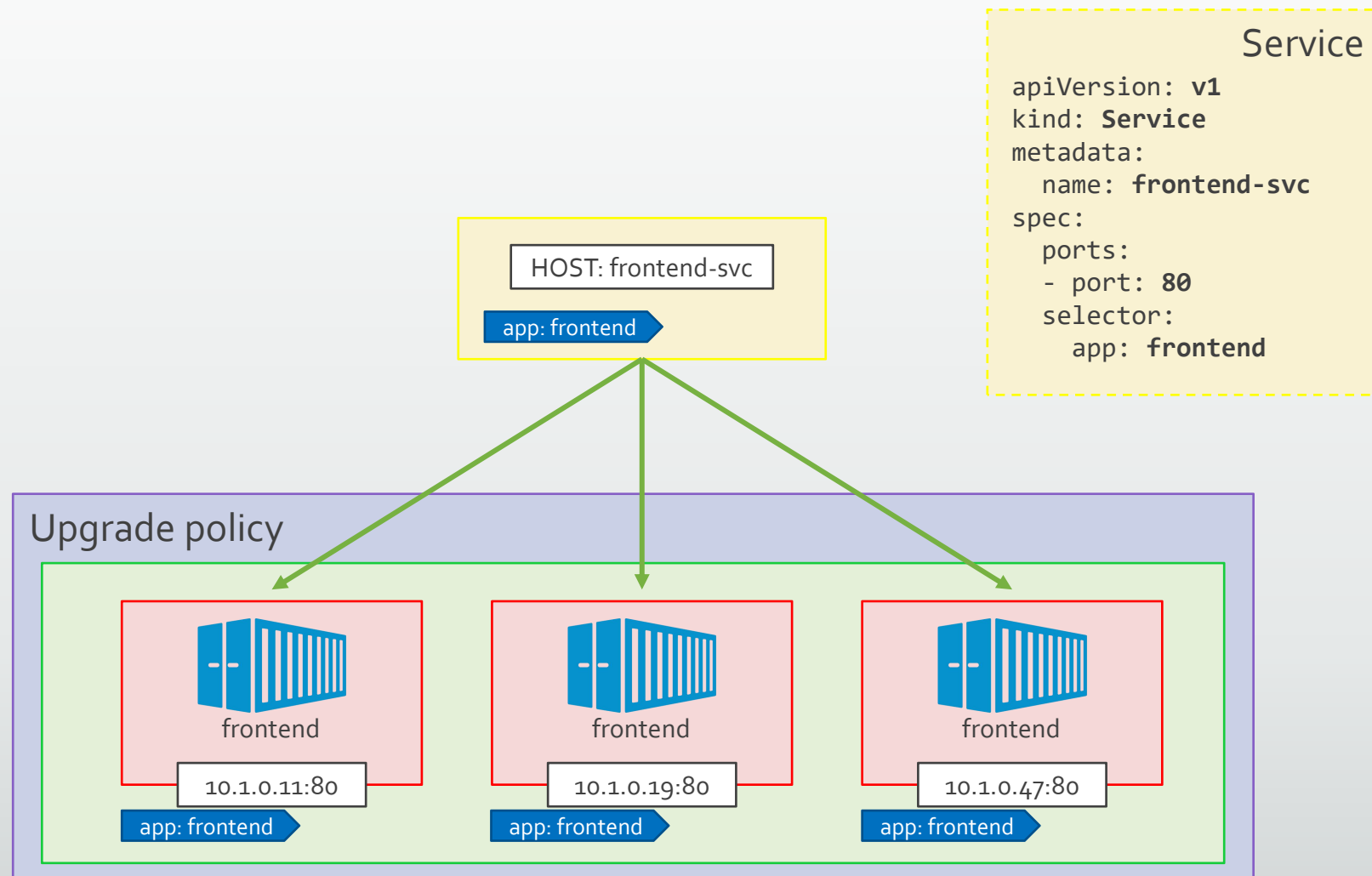
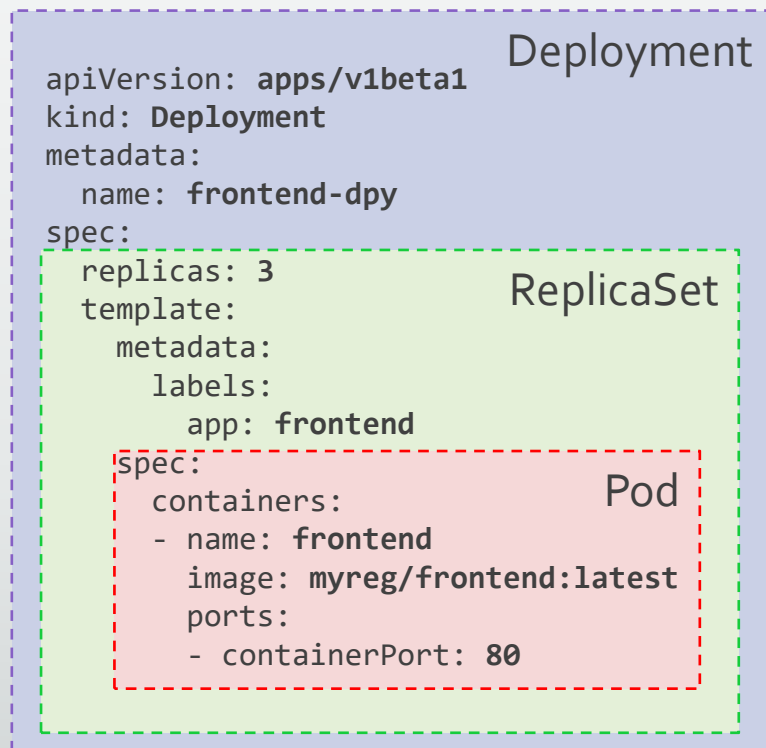
Ok... What's a "Pod" now?!

- Atomic unit of deployment in Kubernetes
- It has its own IP Address
- It's short lived and replaceable
 - Don't get too attached to it 😊
- It runs one or more containers
 - All the containers share storage and network
- We almost never manage pods directly
 - There are higher level objects

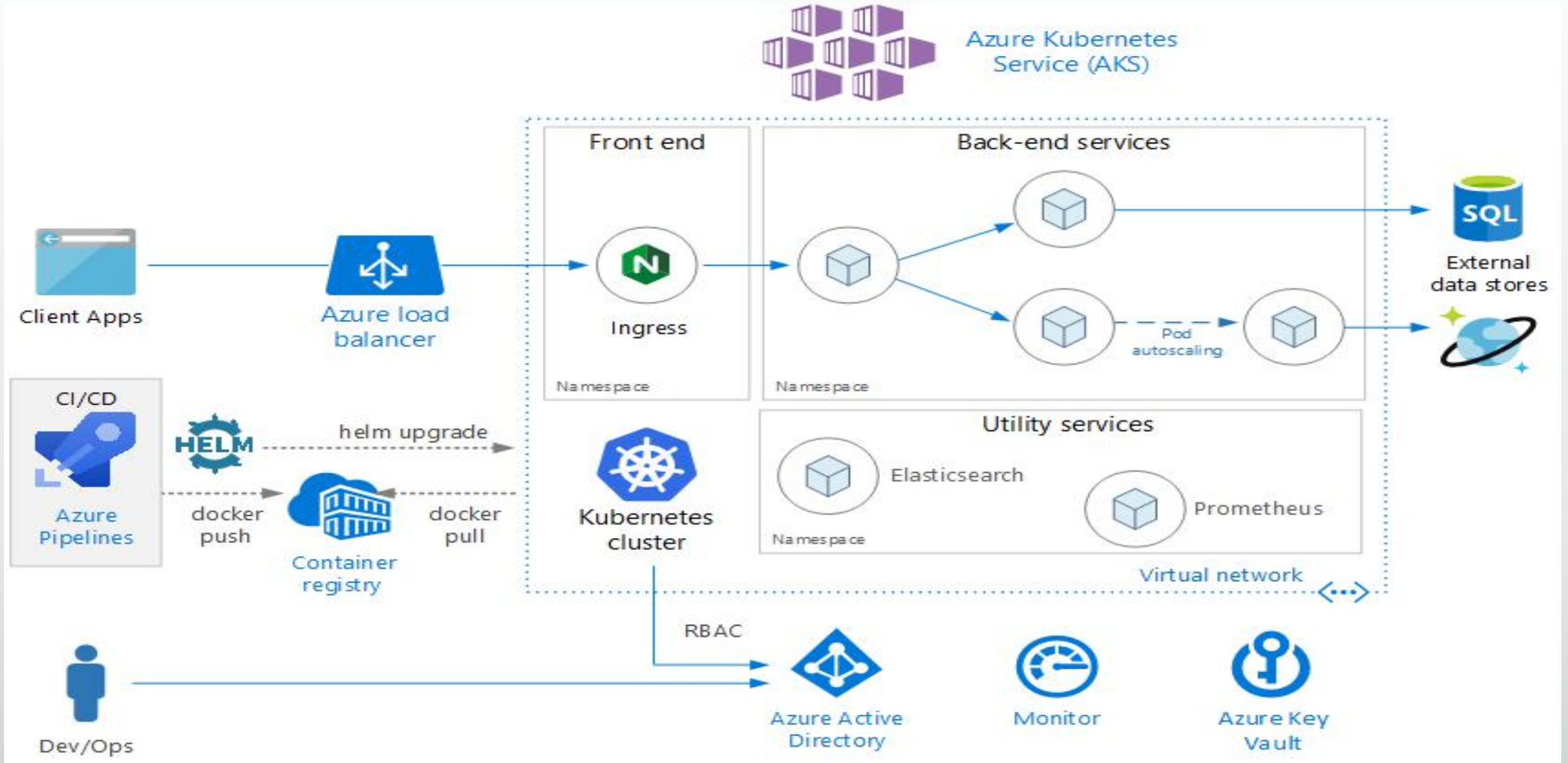




Kubernetes objects are created through its REST API



Microservice architecture on AKS



Recap

- Run **Mongo & Redis** as Docker containers
- Dockerised **frontend** and **backend**
- Added **nginx**
- Described the whole system on **docker-compose**
- Run all of it on our laptop
- Saved the images on **Azure Container Registry**
- Configured a CI/CD pipeline for them
- Deployed on a Kubernetes cluster in **Azure Kubernetes Service (AKS)**
- Set up a dashboard and an alert in **Log Analytics**

Recap - Commands

<code>docker run -p 8080:80 wordpress</code>	Starts a container from an image, exposing it on 8080
<code>docker ps</code>	Lists of all running containers (-a includes stopped ones)
<code>docker start containerName</code>	Starts (and stops) a container
<code>docker rm -f \$(docker ps -q)</code>	Removes all running containers
<code>docker images</code>	Lists all images
<code>docker rmi imageName</code>	Removes an image
<code>docker login myregistry.azurecr.io</code>	Logs in Azure Container Registry
<code>docker tag frontend myregistry.azurecr.io/frontend</code>	Tags an image for Azure Container Registry
<code>docker push myregistry.azurecr.io/frontend</code>	Pushes an image to Azure Container Registry
<code>kubectl apply -f filename</code>	Creates the kubectl objects specified in the file
<code>kubectl get deployments/services/pods</code>	Lists all the deployments/services/pods in the cluster
<code>kubectl proxy --address="0.0.0.0"</code>	Tunnels the cluster's dashboard to localhost

Recap – On Azure

- **Azure Container Registry**
 - Our private repository where we can store Docker images
 - `docker login myregistry.azurecr.io`
- **Azure Web App for Containers**
 - They can run only one container at a time
 - Sort of CD supported via Azure Container Registry and web hooks
- **Azure Container Service (AKS)**
 - Fully managed orchestrator
 - Based on Kubernetes
 - <https://docs.microsoft.com/en-us/azure/aks/>
- **Container monitoring solution**
 - Based on Azure LogAnalytics
 - Uses a DaemonSet to track cluster events
 - <https://docs.microsoft.com/en-us/azure/log-analytics/log-analytics-containers>

Thank you!

@geffzhang

geffzhang@weyhd.com

Get the code at

<https://github.com/geffzhang/DockerStarted>