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

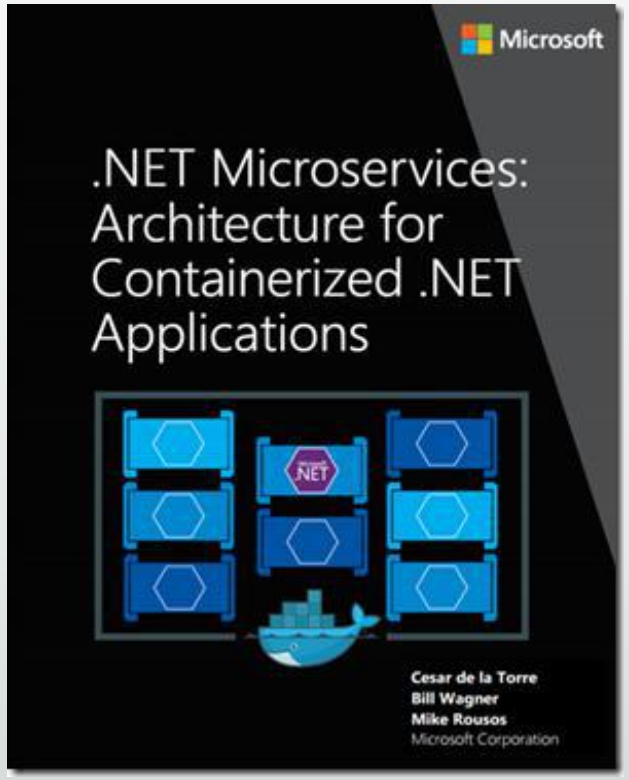
A microservices architecture consists of a collection of small, autonomous services. Each service is self-contained and should implement a single business capability.

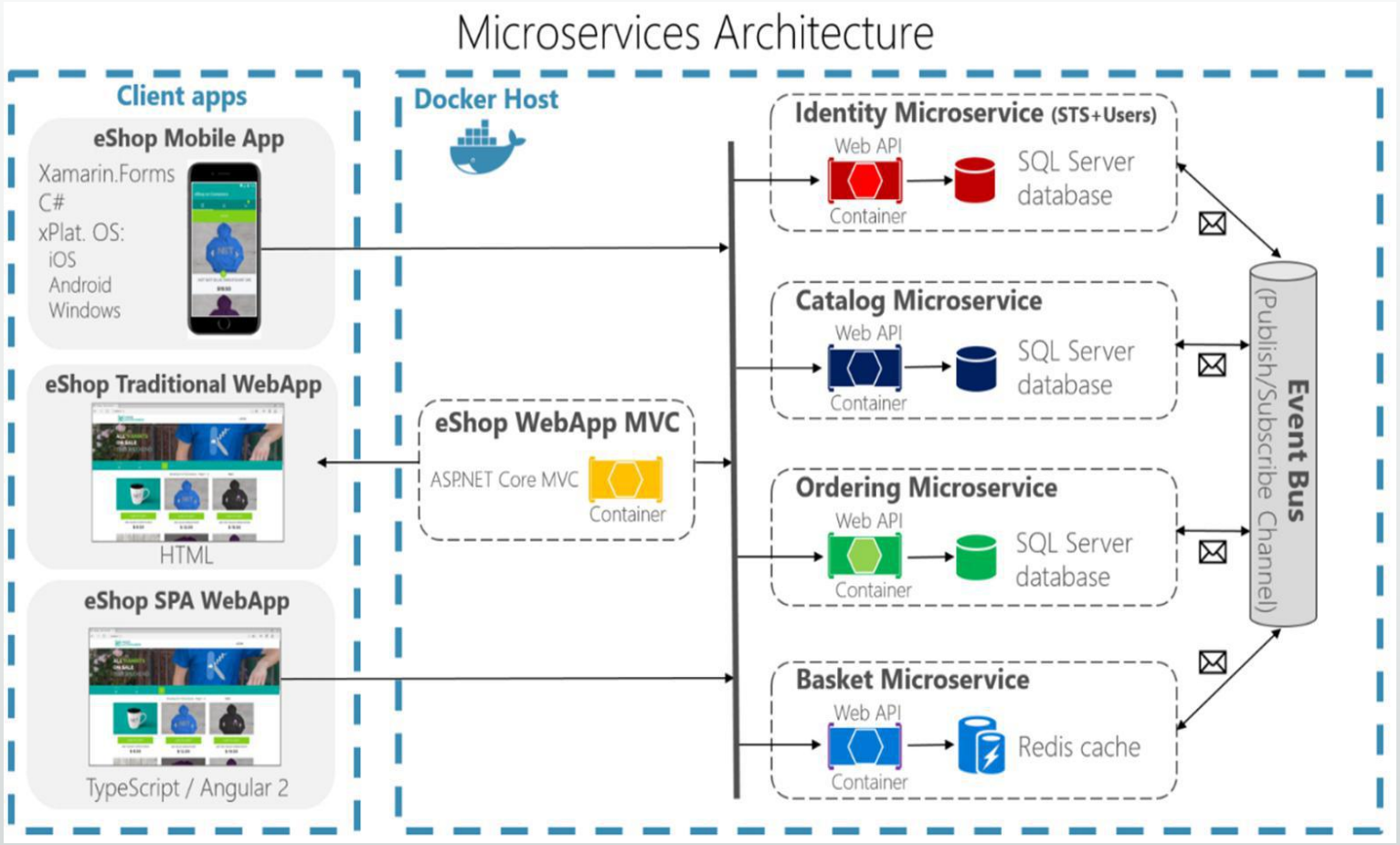
Characteristics of a Microservice

- 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.

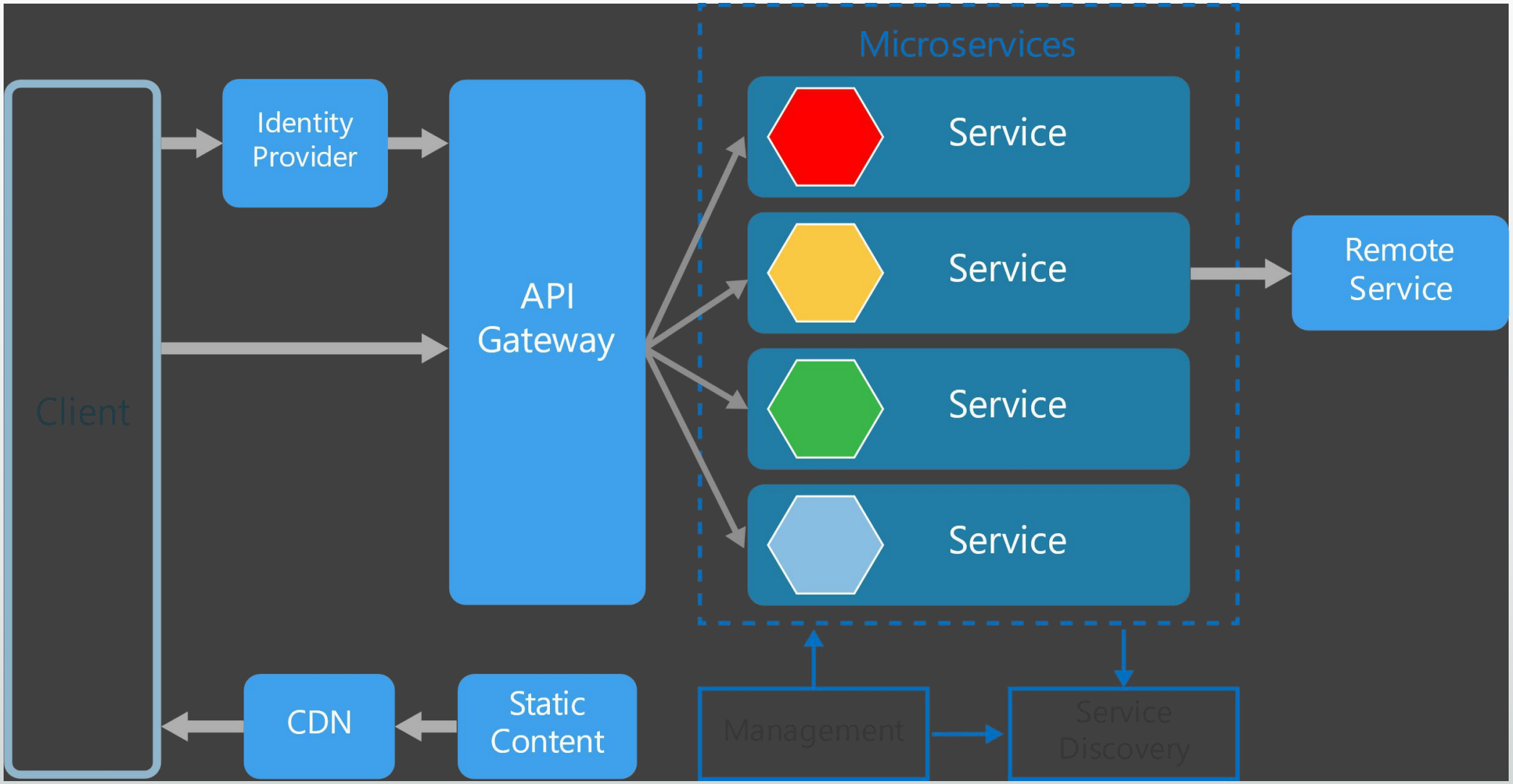
.NET Microservices - Architecture for Containerized .NET Applications

http://aka.ms/MicroservicesEbook





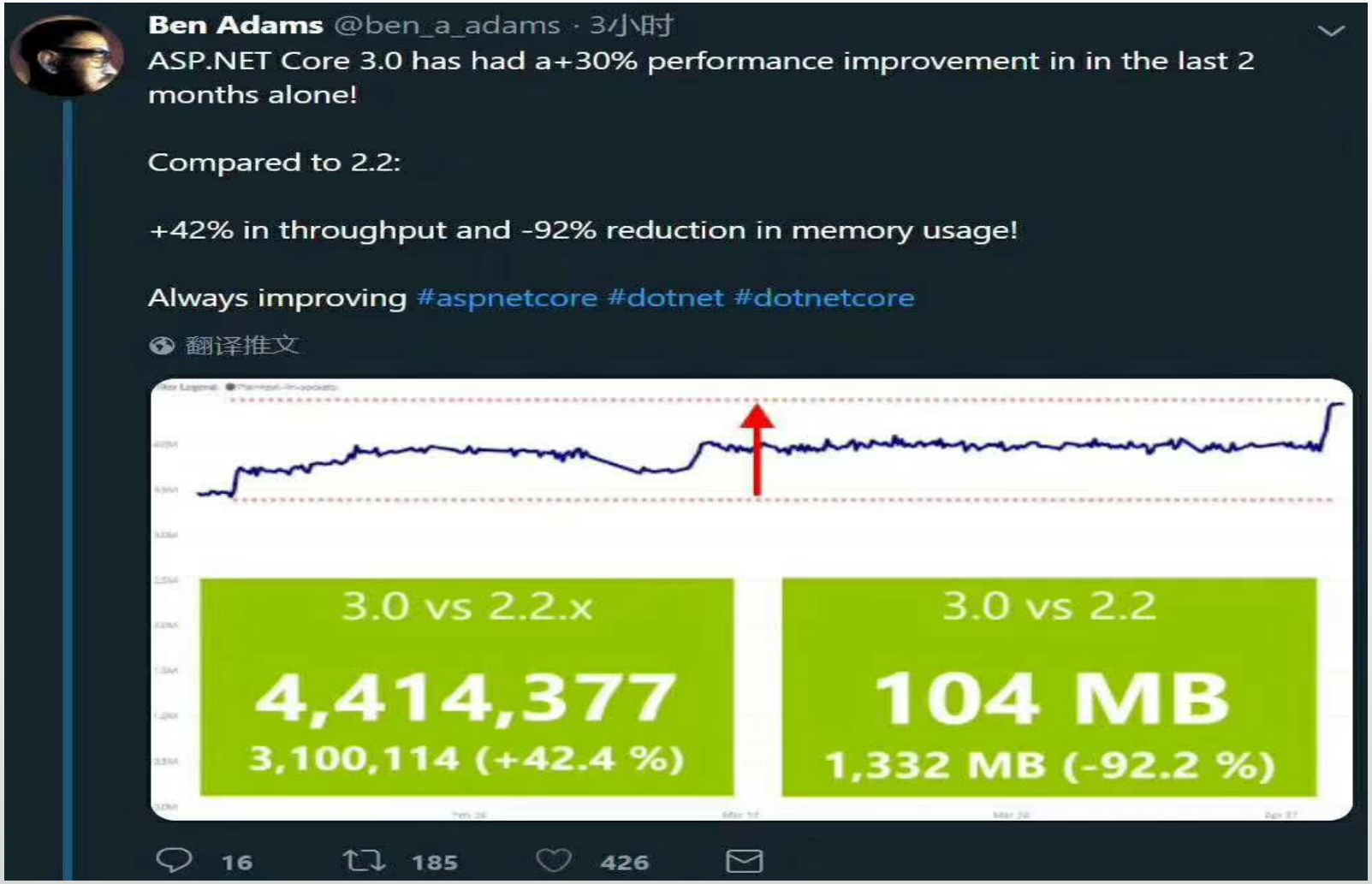
Microservices Overview



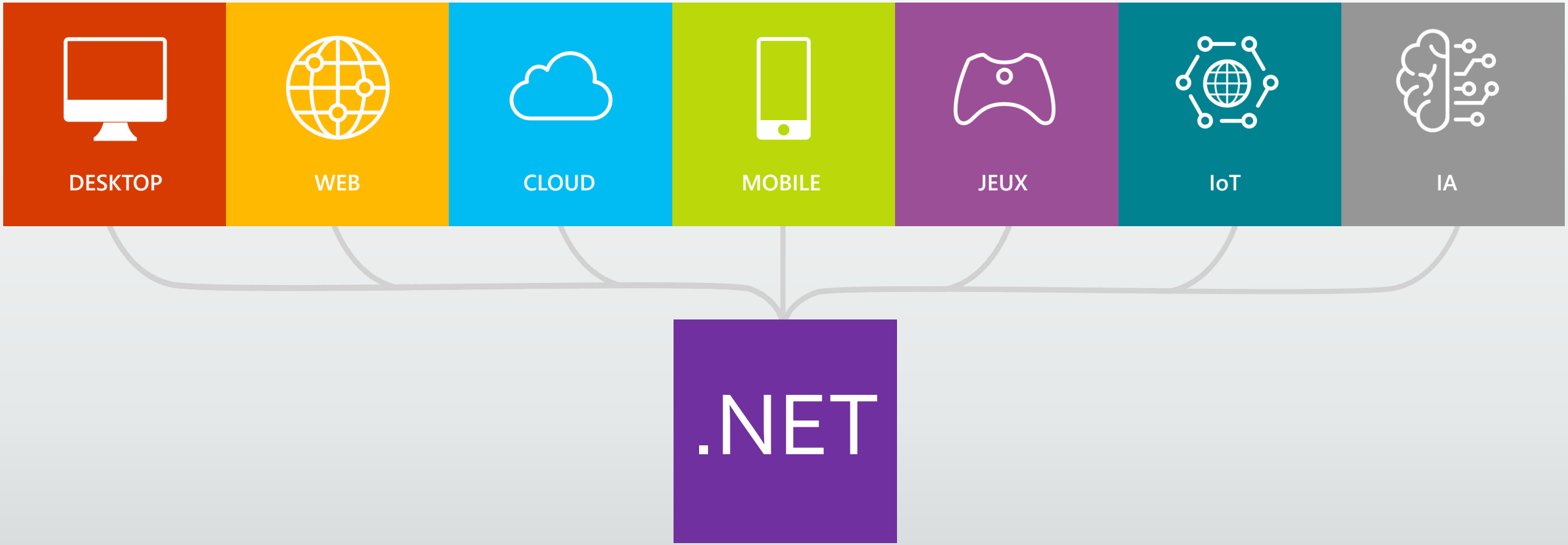
Other Components in a Typical Microservices Architecture

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

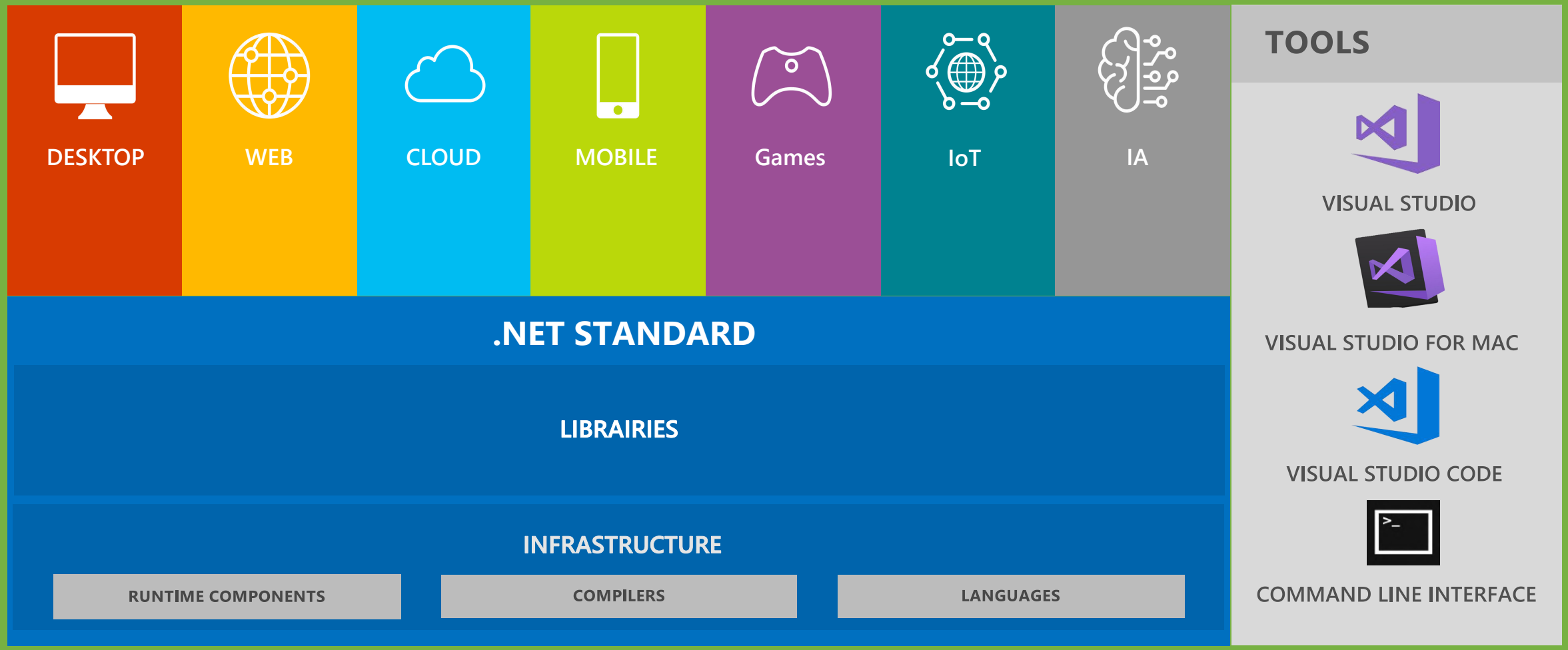
.NET Core


Ben Adams @ben_a_adams · 3小时
ASP.NET Core 3.0 has had a+30% performance improvement in 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
3.0 vs 2.2.x
4,414,377
3,100,114 (+42.4 %)
3.0 vs 2.2
104 MB
1,332 MB (-92.2 %)
16
185
426

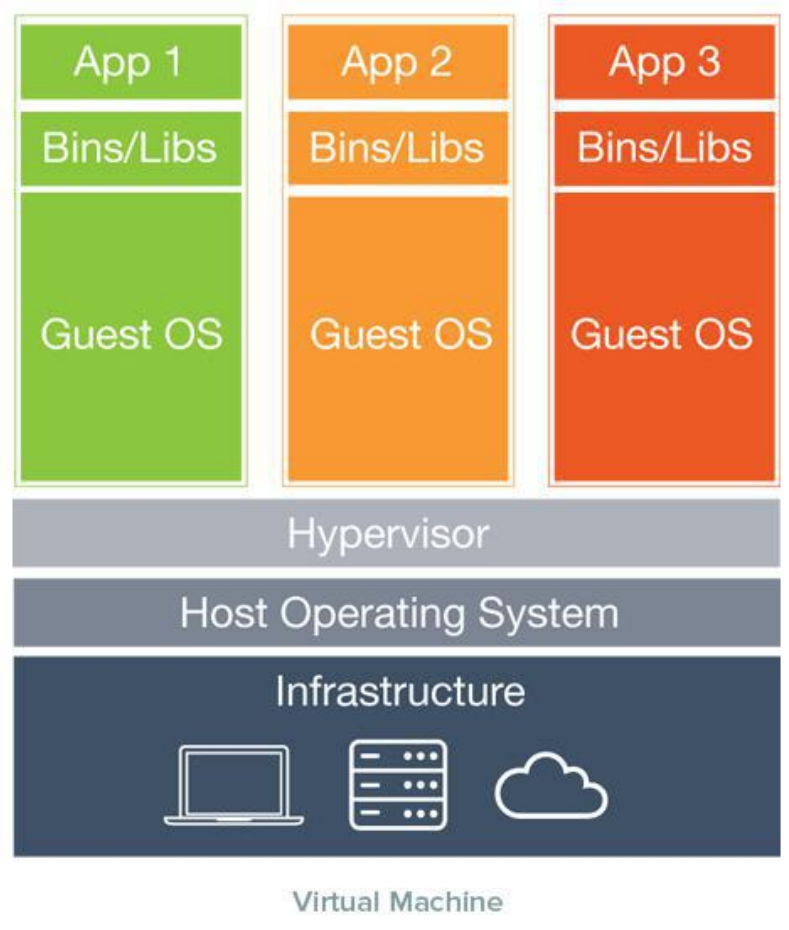
A platform and build everything

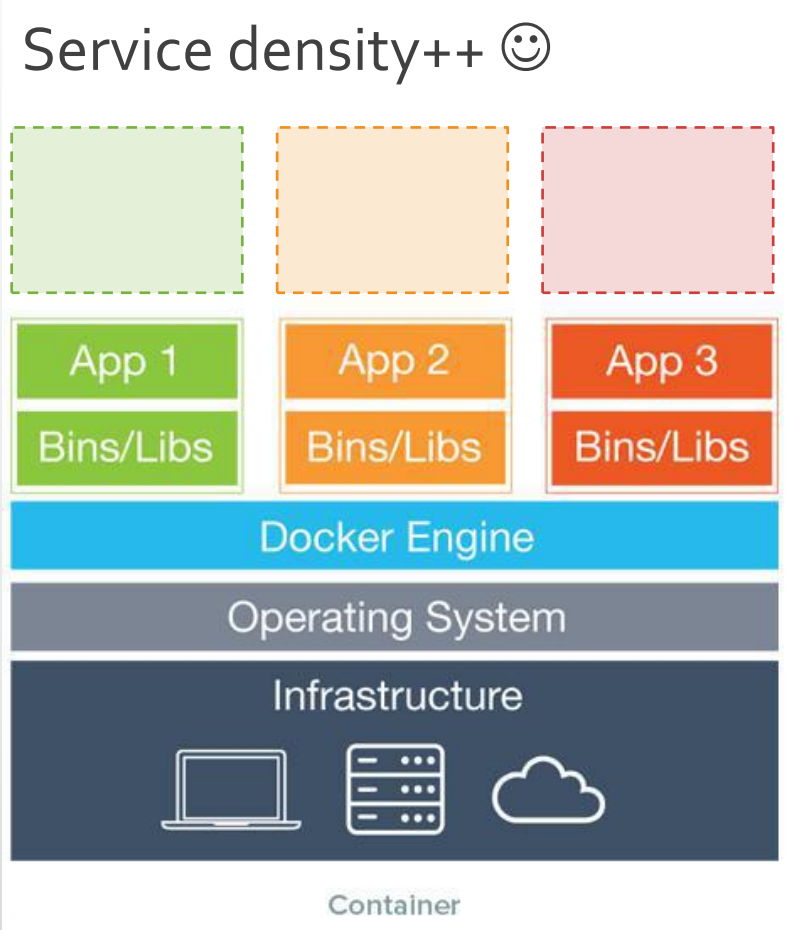


.NET - Software development platform



What's Docker

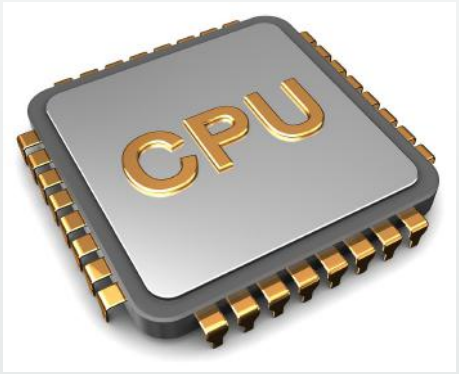




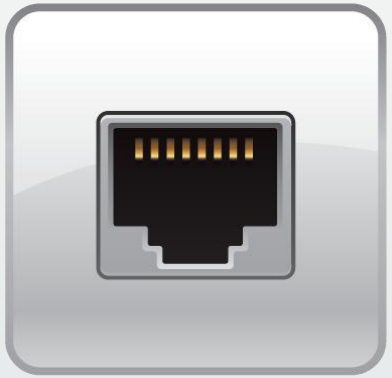
Containers == Virtualized Operating System

Kernel namespaces (Linux and Windows)

Virtual File System

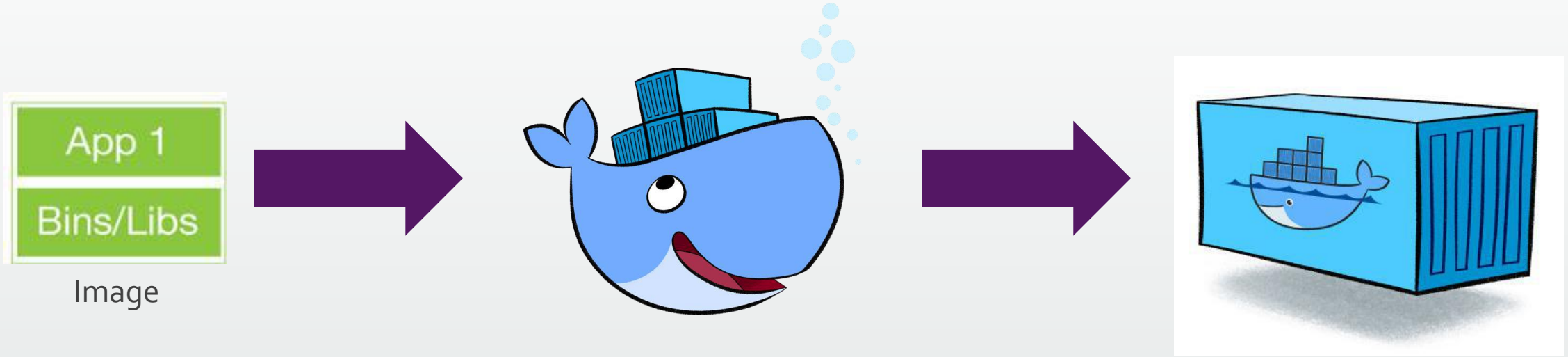


Virtual Process Tree



Virtual Network

Docker is a (the) container engine

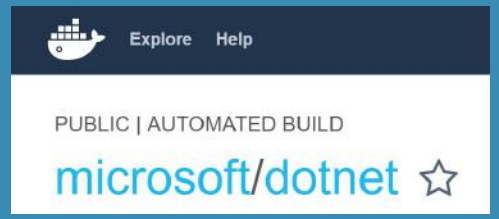


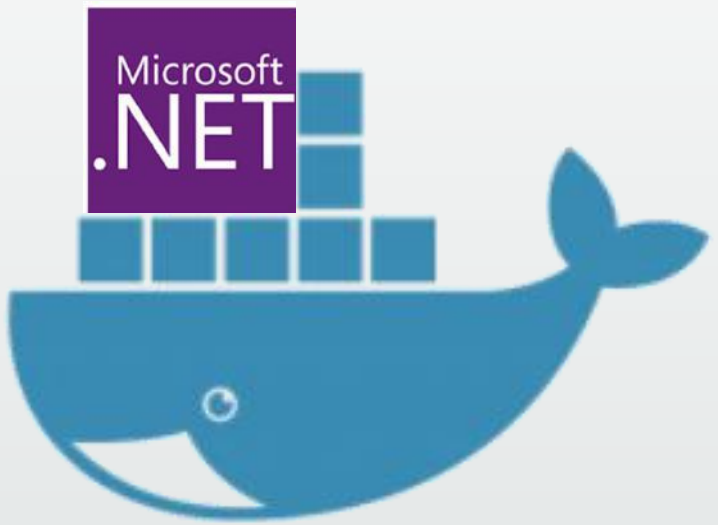
- Docker (aka Moby project) is free and open source, no limitations
- There's an enterprise edition (hosting, support, certification…)
- It's the de-facto standard for container-based applications

Docker and .NET

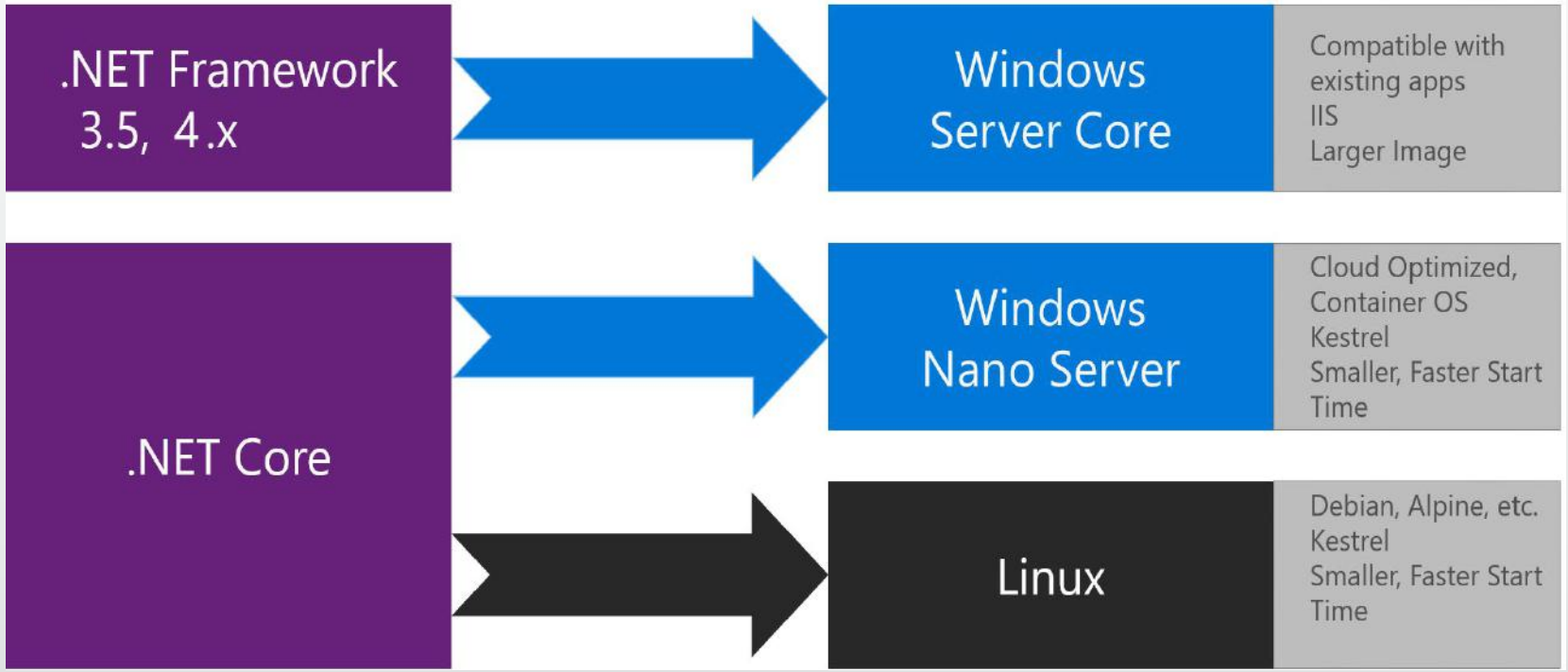
- **.NET Core** Docker images
  xPlat. (Linux & Windows)
- **.NET Framework** images
  Windows only

See at Docker Hub

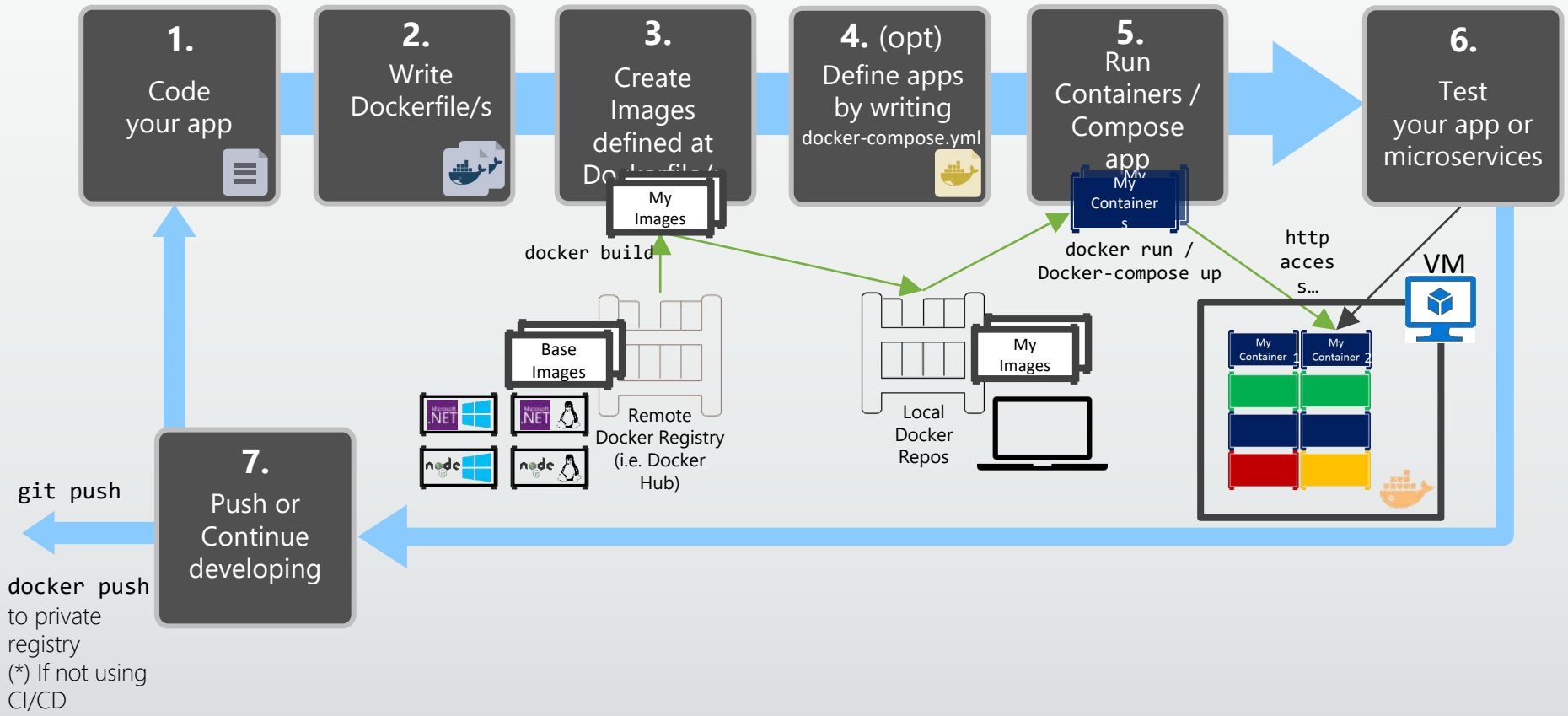




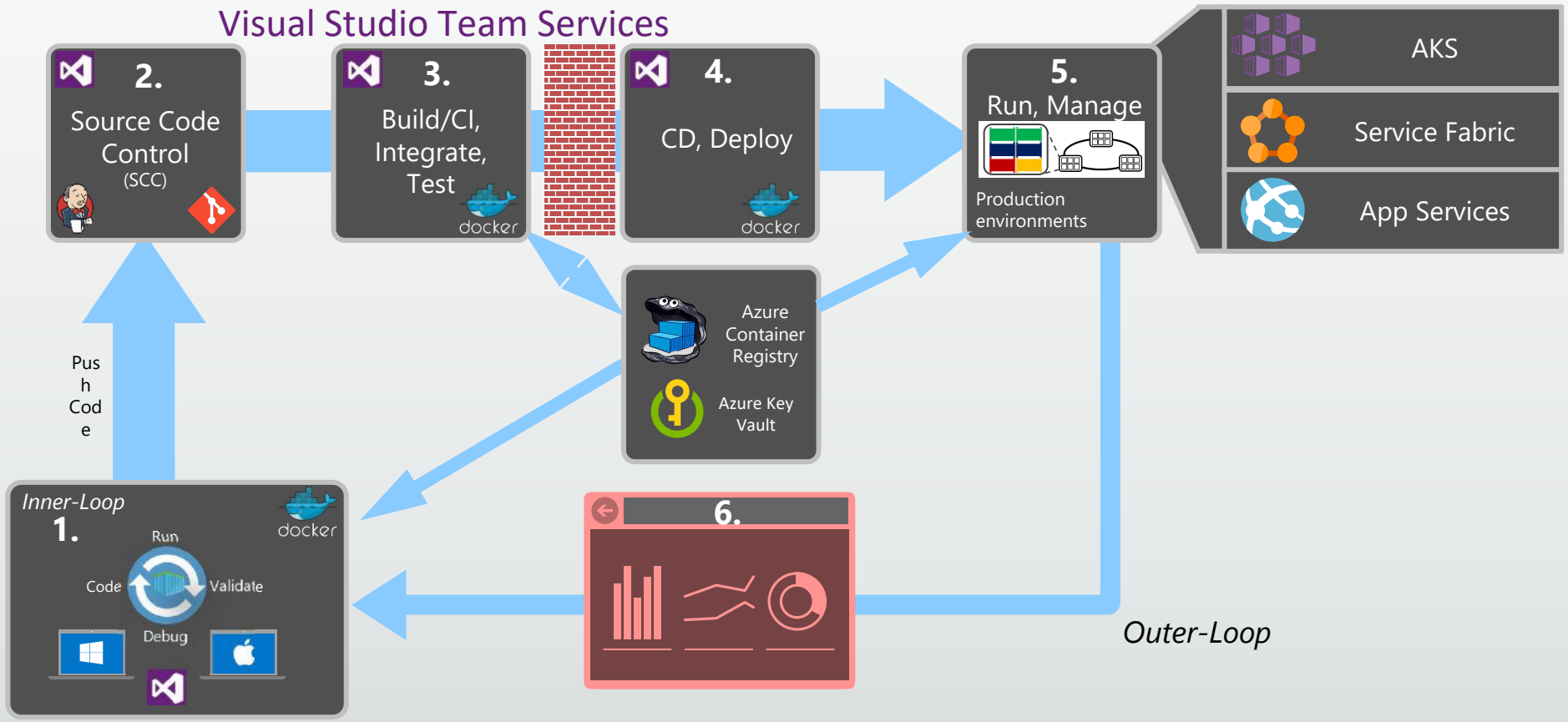
What OS to target with .NET containers



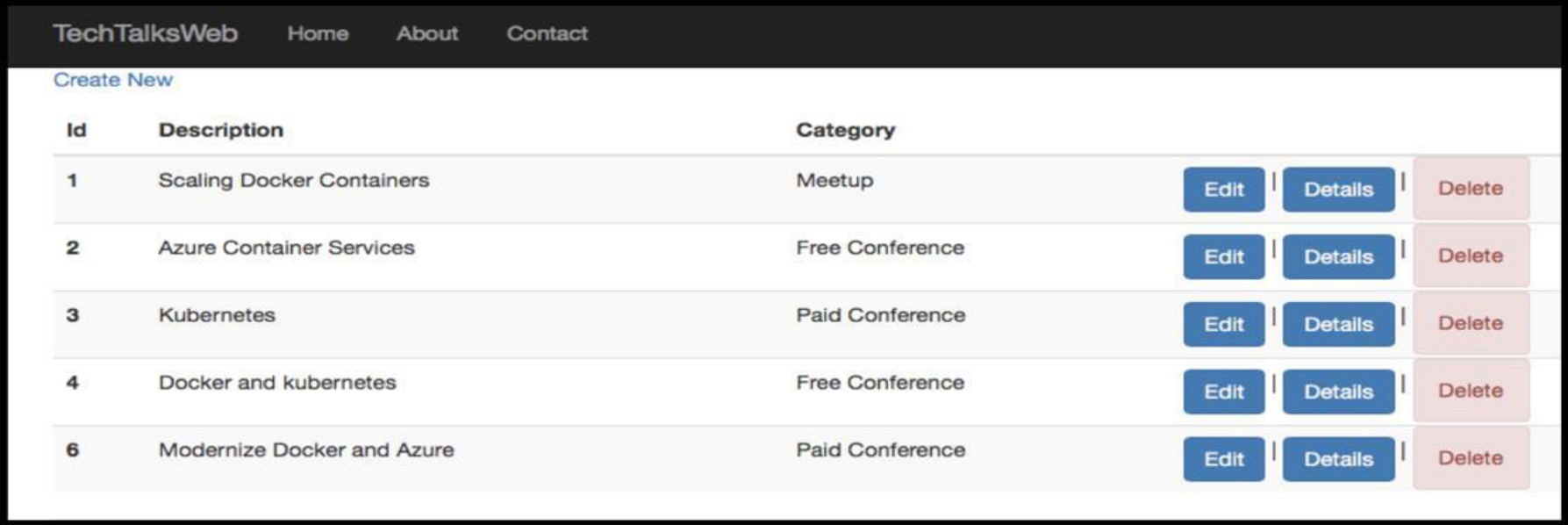
Inner-Loop development workflow for Docker apps

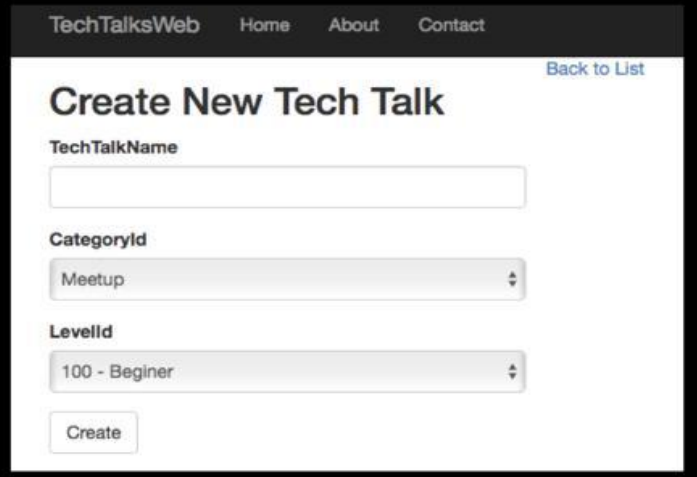


DevOps Workflow using containers



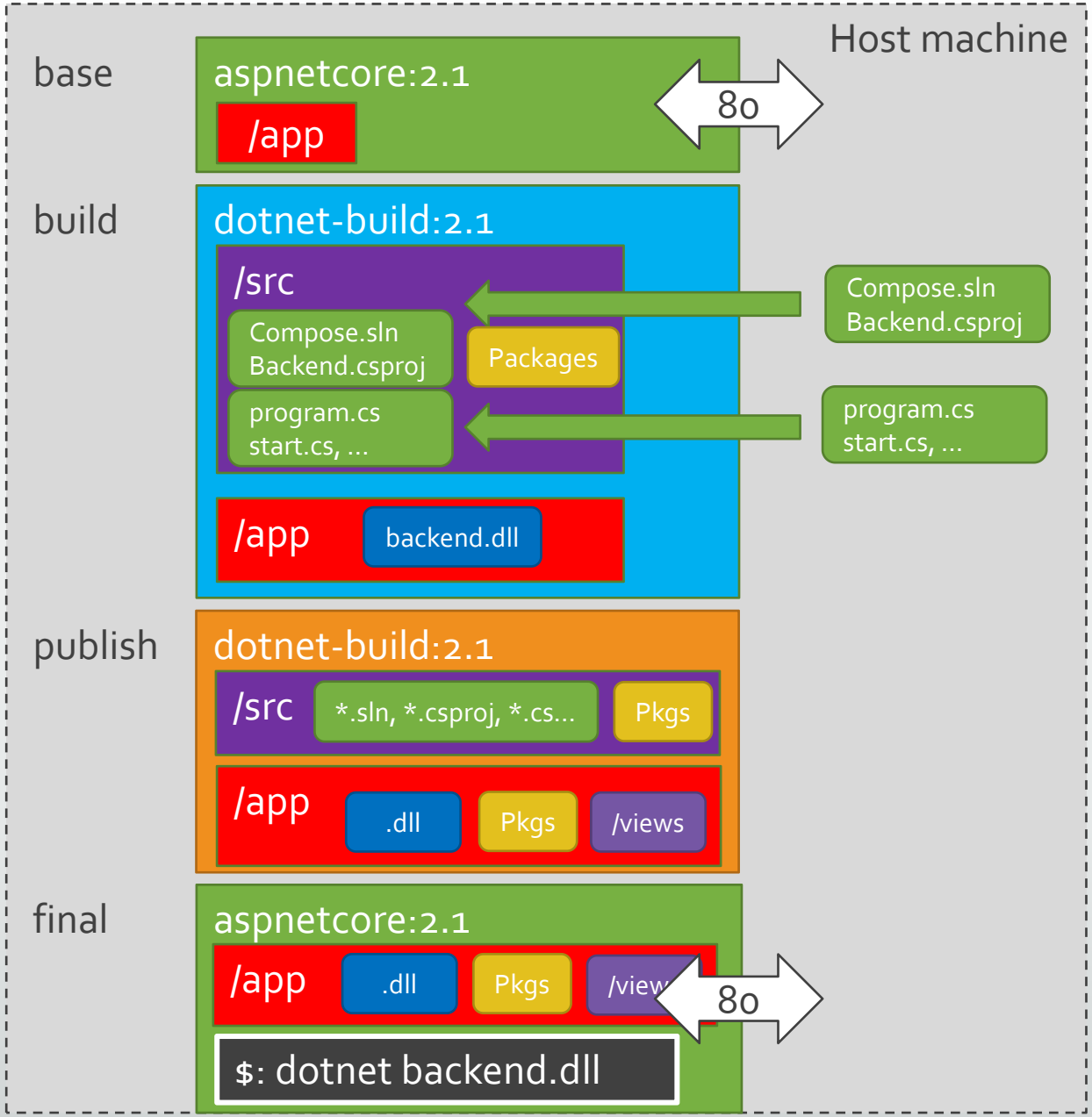
What we are going to build





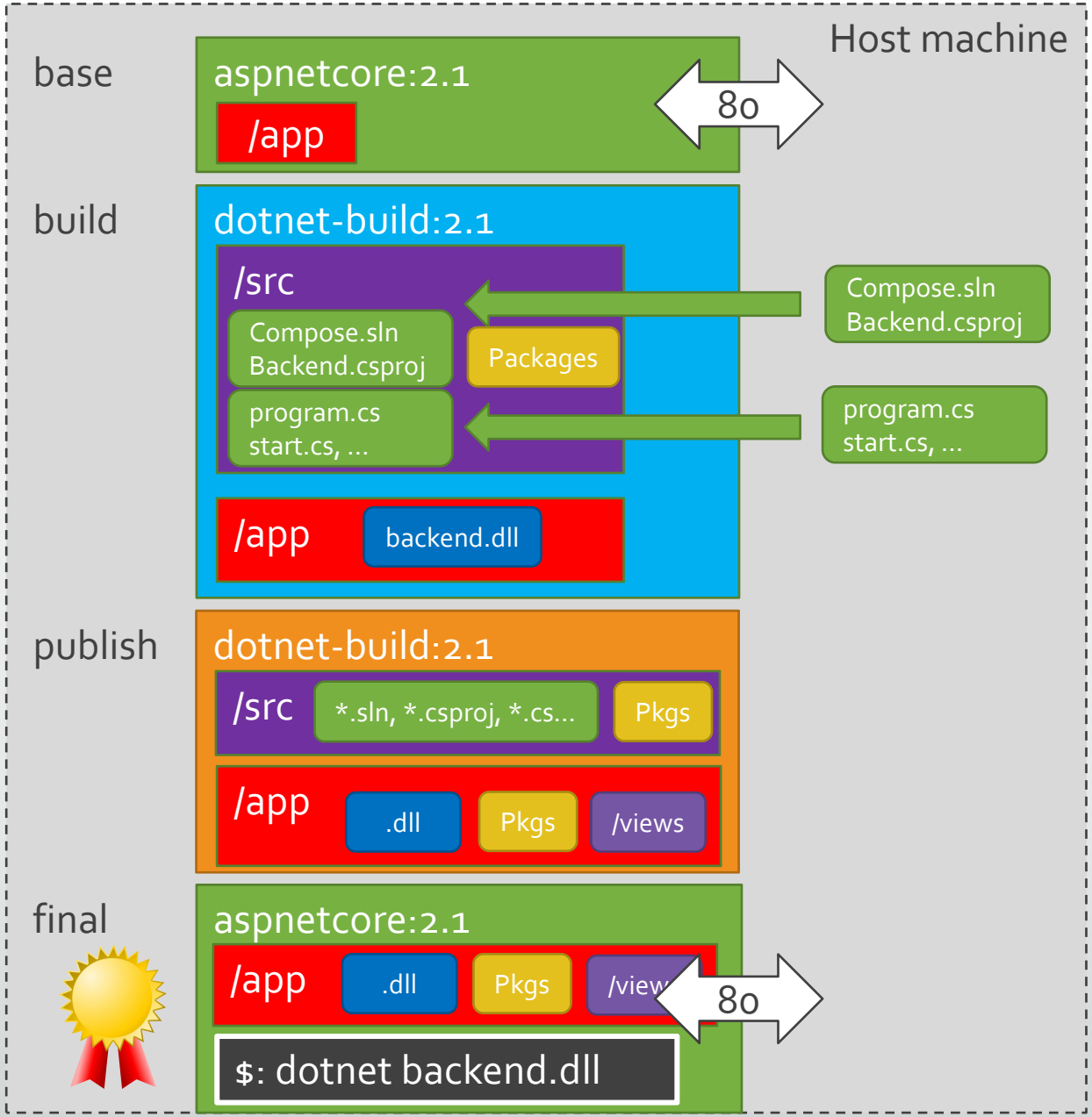
Dockerfile explained

```
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"]
```

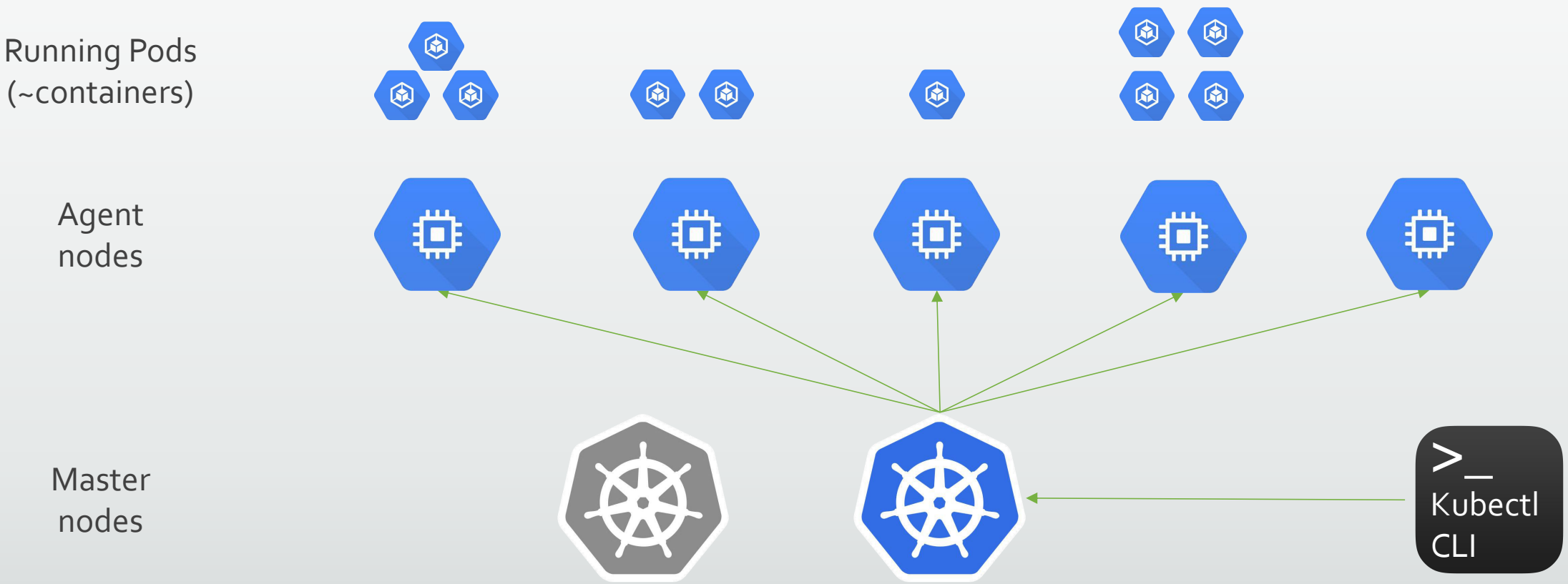


Dockerfile explained

```
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"]
```

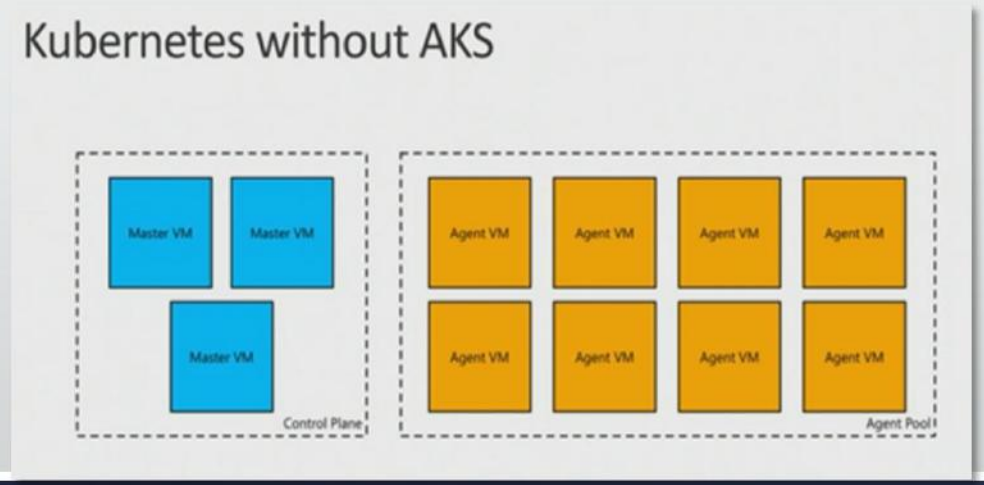


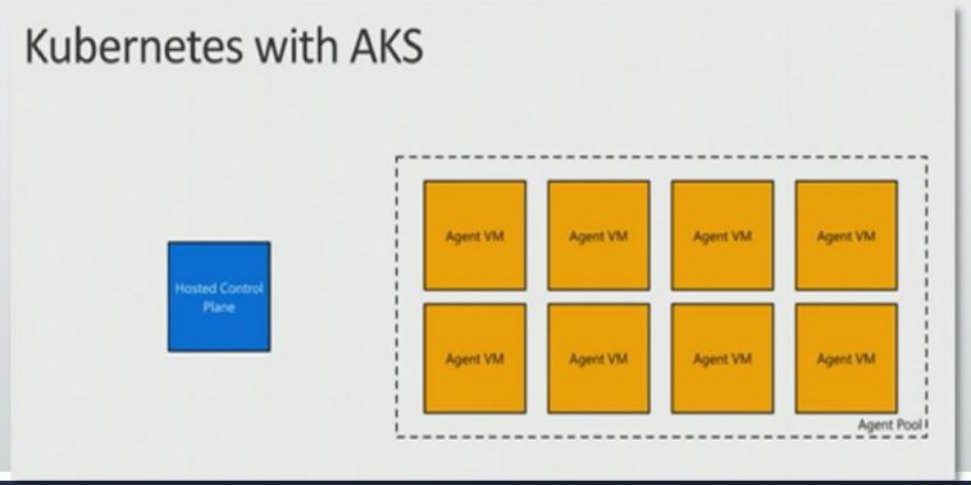
We need an orchestrator: enter Kubernetes and AKS



AKS: Managed Kubernetes

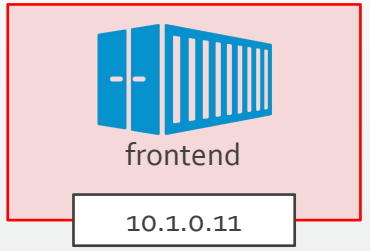
- Azure-hosted control plane
  - No master nodes to manage or pay for
- Automated upgrades and patching
  - Easily upgrade control plane and worker nodes to new versions of Kubernetes
- Scale agent pool to increase or decrease capacity

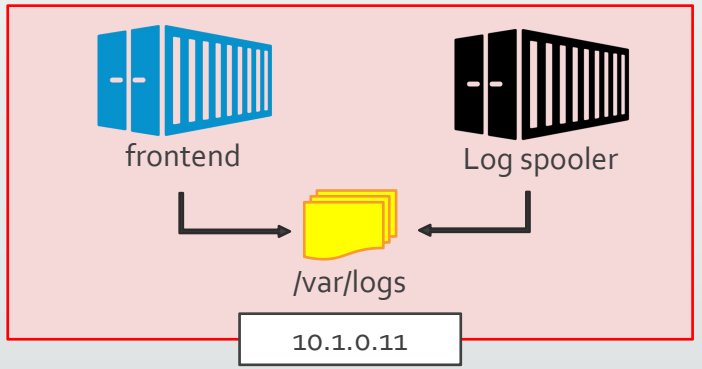




Ok… What's a "Pod" now?!

- 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

Service

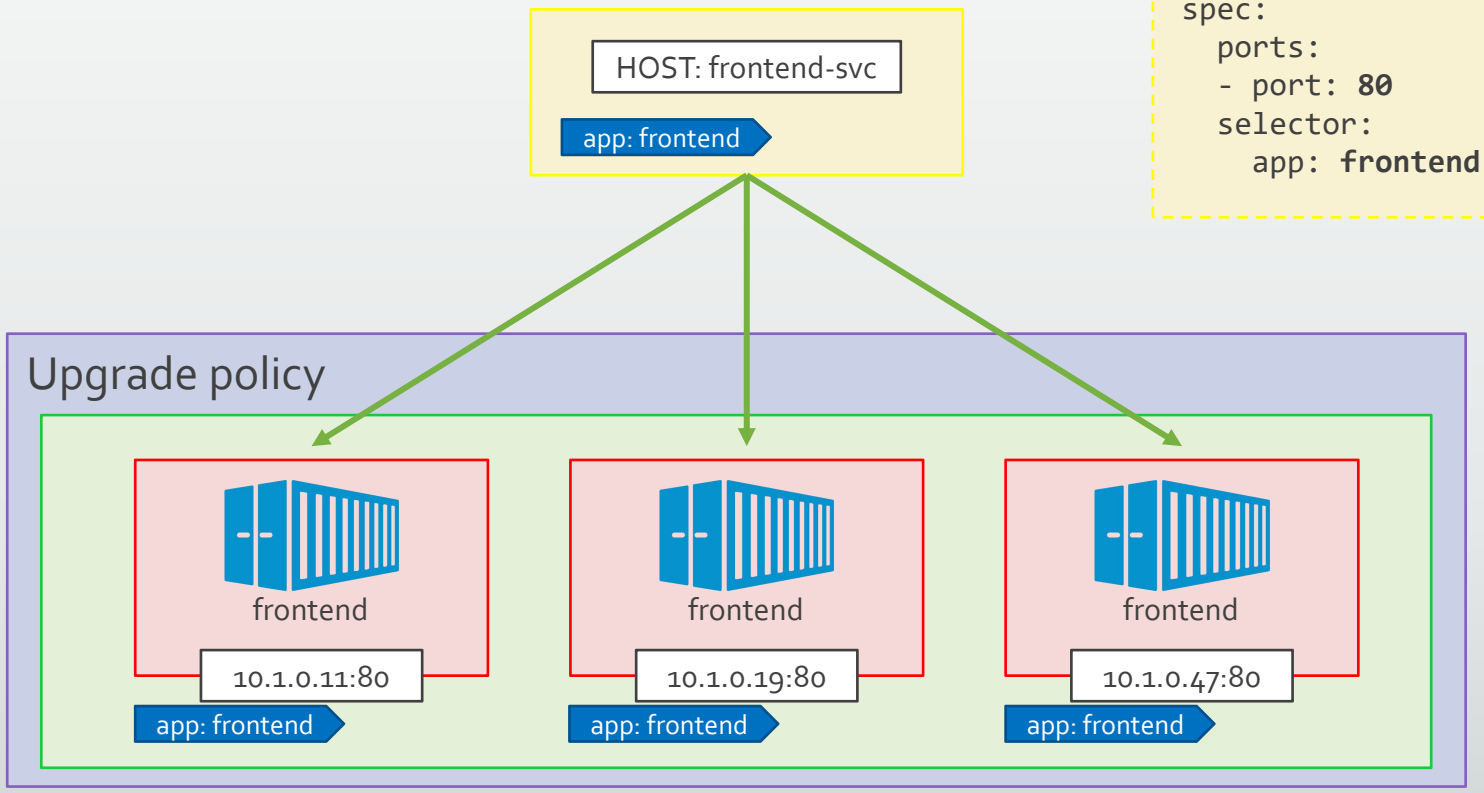
```
apiVersion: v1
kind: Service
metadata:
  name: frontend-svc
spec:
  ports:
  - port: 80
  selector:
    app: frontend
```

Deployment

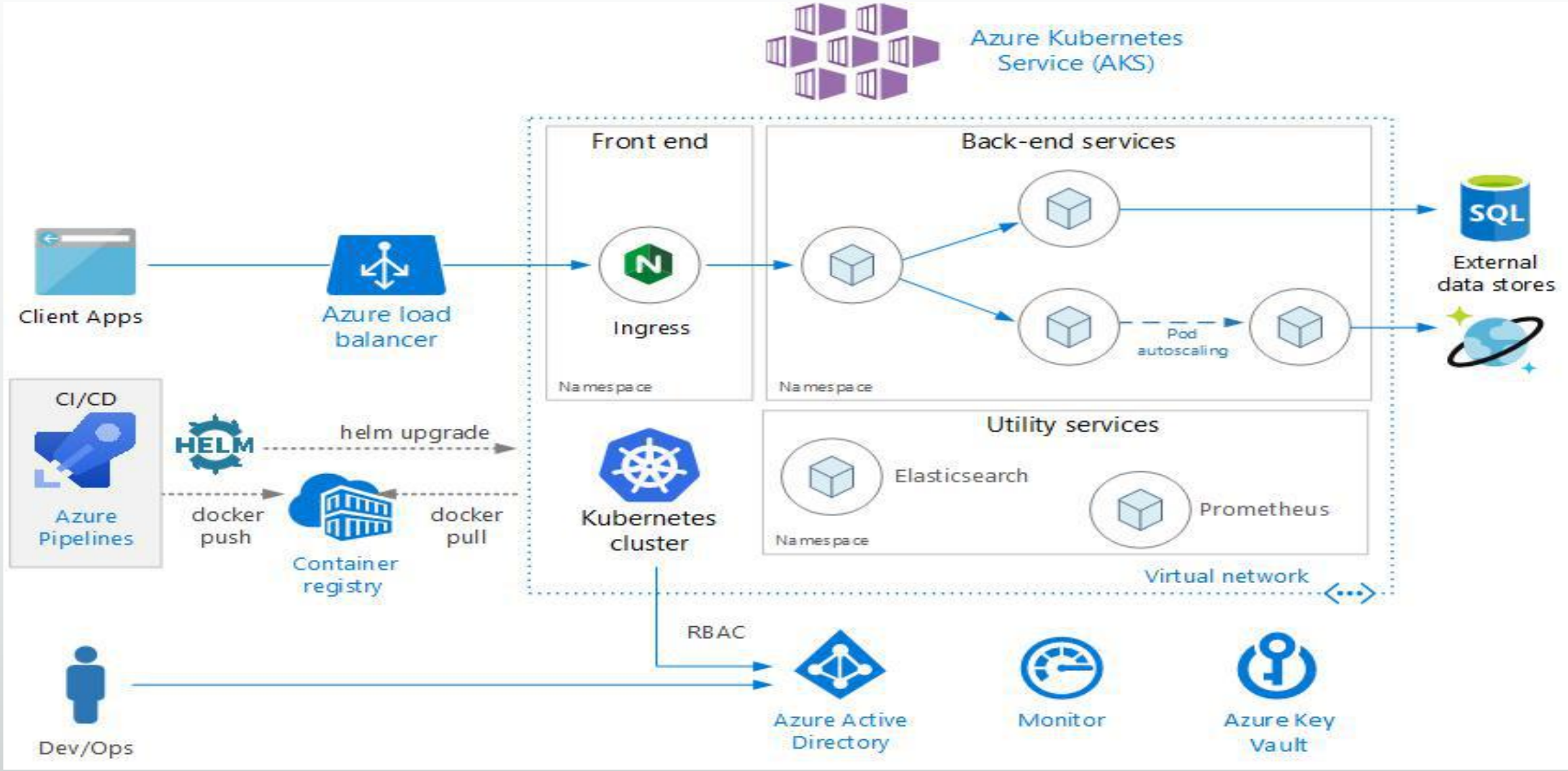
```
apiVersion: apps/v1beta1
kind: Deployment
metadata:
  name: frontend-dpy
spec:
  replicas: 3
  template:
    metadata:
      labels:
        app: frontend
    spec:
      containers:
      - name: frontend
        image: myreg/frontend:latest
        ports:
        - containerPort: 80
```

ReplicaSet

Pod



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

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